WebFOCUS

Using WebFOCUS RStat for Predictive Analytics

Version 2.0

January 05, 2018
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Preface</strong></td>
<td>7</td>
</tr>
<tr>
<td>Documentation Conventions</td>
<td>8</td>
</tr>
<tr>
<td>Related Publications</td>
<td>9</td>
</tr>
<tr>
<td>Customer Support</td>
<td>9</td>
</tr>
<tr>
<td>Information You Should Have</td>
<td>10</td>
</tr>
<tr>
<td>User Feedback</td>
<td>11</td>
</tr>
<tr>
<td>Information Builders Consulting and Training</td>
<td>11</td>
</tr>
<tr>
<td><strong>1. Introducing WebFOCUS RStat</strong></td>
<td>13</td>
</tr>
<tr>
<td>Highlights of WebFOCUS RStat</td>
<td>13</td>
</tr>
<tr>
<td>Data Mining With WebFOCUS RStat</td>
<td>15</td>
</tr>
<tr>
<td>Benefits of a Fully Integrated Environment</td>
<td>15</td>
</tr>
<tr>
<td>RStat Architecture</td>
<td>16</td>
</tr>
<tr>
<td><strong>2. Installing RStat</strong></td>
<td>17</td>
</tr>
<tr>
<td>Installation Prerequisites</td>
<td>17</td>
</tr>
<tr>
<td>RStat Installation</td>
<td>18</td>
</tr>
<tr>
<td>RStat System Requirements</td>
<td>18</td>
</tr>
<tr>
<td>Installing the MinGW Compiler</td>
<td>25</td>
</tr>
<tr>
<td><strong>3. Getting Started With RStat</strong></td>
<td>33</td>
</tr>
<tr>
<td>Starting RStat With Data Loaded</td>
<td>33</td>
</tr>
<tr>
<td>Starting RStat Without Data Loaded in App Studio</td>
<td>36</td>
</tr>
<tr>
<td>Loading Data From Within RStat</td>
<td>36</td>
</tr>
<tr>
<td>RScripts in RStat</td>
<td>43</td>
</tr>
<tr>
<td>Running an RScript</td>
<td>43</td>
</tr>
<tr>
<td>Running an RScript in Order to Create a C File.</td>
<td>46</td>
</tr>
<tr>
<td><strong>4. Data Exploration and Transformations in the WebFOCUS RStat Tool</strong></td>
<td>51</td>
</tr>
<tr>
<td>Variable Definitions and Sampling</td>
<td>51</td>
</tr>
<tr>
<td>Data Source Selection</td>
<td>54</td>
</tr>
<tr>
<td>Data Exploration</td>
<td>57</td>
</tr>
<tr>
<td>Hypothesis Testing</td>
<td>86</td>
</tr>
<tr>
<td>Data Transformation</td>
<td>91</td>
</tr>
<tr>
<td>Clustering</td>
<td>97</td>
</tr>
</tbody>
</table>
5. Creating a Scoring Application ............................................... 101
   Defining the Model Data .............................................................. 101
   Defining Model Sampling.......................................................... 101
   Defining Variable Roles.......................................................... 103
   Setting Variable Roles for Groups............................................. 104
   Setting the Target Type.......................................................... 105
   Executing Data Settings.......................................................... 106

Building a Model ...................................................................... 107
   Model Tab Options...................................................................... 107
      Ada Boost Model................................................................. 109
      ARIMA Model........................................................................ 112
      Bayesian Network Model..................................................... 116
         Plotting Conditional Probability......................................... 121
         Bayes Factor Analysis....................................................... 123
      Decision Tree Model............................................................ 125
      Regression Model................................................................... 128
      Neural Net Model................................................................... 129
      Random Forest Model........................................................... 132
      Survival Model....................................................................... 136
   SVM Model............................................................................... 137

Building the Decision Tree Model........................................... 140
Visualizing the Decision Tree Model.......................................... 141
   Using FancyPlots...................................................................... 145
      Charting With FancyPlots....................................................... 149
      Decision-Tree Snipping........................................................... 149
      Saving a Decision Tree as a PDF File...................................... 153
      Saving a Decision Tree as a C Routine.................................. 153
      Creating FancyPlots.............................................................. 154

Evaluating the Model.............................................................. 163
   Exporting Rpart Rules............................................................... 163
   Evaluating the Decision Tree.................................................... 167
      Error Matrix........................................................................... 169
Preface

This documentation describes how to use WebFOCUS RStat®. It is intended for users who are looking to build predictive models and scoring applications in a WebFOCUS App Studio.

How This Manual Is Organized

This manual includes the following chapters:

<table>
<thead>
<tr>
<th>Chapter/Appendix</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introducing WebFOCUS RStat</td>
<td>Provides information about configuring and accessing WebFOCUS RStat to build predictive and scoring applications in WebFOCUS App Studio.</td>
</tr>
<tr>
<td>2 Installing RStat</td>
<td>Contains instructions for installing RStat.</td>
</tr>
<tr>
<td>3 Getting Started With RStat</td>
<td>Provides instructions about loading data into RStat.</td>
</tr>
<tr>
<td>4 Data Exploration and Transformations in the WebFOCUS RStat Tool</td>
<td>Provides an overview about layout, toolbars, and analytical features for RStat.</td>
</tr>
<tr>
<td>5 Creating a Scoring Application</td>
<td>Describes the process of building a scoring application with RStat.</td>
</tr>
<tr>
<td>6 Creating Statistically Valid Data Samples</td>
<td>Describes advanced sampling techniques, which enable you to generate statistically representative data extracts for data examination and modeling purposes.</td>
</tr>
<tr>
<td>7 Building a Linear Regression Model</td>
<td>Describes some of the basic models used for predictive analytics.</td>
</tr>
<tr>
<td>8 Building a Logistic Model</td>
<td>Describes some of the basic logistic models used to predict outcomes.</td>
</tr>
<tr>
<td>9 Building a Survival Model</td>
<td>This chapter discusses survival models that you can use to model time-to-event data.</td>
</tr>
<tr>
<td>10 Building a Decision Tree Model</td>
<td>Describes the decision tree model and when you should use it.</td>
</tr>
</tbody>
</table>
### Chapter/Appendix

<table>
<thead>
<tr>
<th>Chapter/Appendix</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>11 Building a Market Basket Model</td>
<td>This chapter discusses Market Basket Analysis. You can use Market Basket Analysis to analyze transactions and evaluate what items are frequently purchased together.</td>
</tr>
<tr>
<td>A Deploying RStat Scoring Routines on WebFOCUS Reporting Servers Prior to Release 7.6.8</td>
<td>Provides information on how to deploy RStat scoring routines prior to Version 7.6.8.</td>
</tr>
<tr>
<td>B Glossary</td>
<td>Defines key terms and concepts in this manual.</td>
</tr>
</tbody>
</table>

### Documentation Conventions

The following table describes the documentation conventions that are used in this manual.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>THIS TYPEFACE</strong> or <strong>this typeface</strong></td>
<td>Denotes syntax that you must enter exactly as shown.</td>
</tr>
<tr>
<td><strong>this typeface</strong></td>
<td>Represents a placeholder (or variable) in syntax for a value that you or the system must supply.</td>
</tr>
<tr>
<td><strong>underscore</strong></td>
<td>Indicates a default setting.</td>
</tr>
<tr>
<td><strong>this typeface</strong></td>
<td>Represents a placeholder (or variable), a cross-reference, or an important term. It may also indicate a button, menu item, or dialog box option that you can click or select.</td>
</tr>
<tr>
<td><strong>Key + Key</strong></td>
<td>Indicates keys that you must press simultaneously.</td>
</tr>
<tr>
<td><strong>{  }</strong></td>
<td>Indicates two or three choices. Type one of them, not the braces.</td>
</tr>
<tr>
<td><strong>[  ]</strong></td>
<td>Indicates a group of optional parameters. None are required, but you may select one of them. Type only the parameter in the brackets, not the brackets.</td>
</tr>
<tr>
<td>Convention</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Separates mutually exclusive choices in syntax. Type one of them, not the symbol.</td>
</tr>
<tr>
<td>...</td>
<td>Indicates that you can enter a parameter multiple times. Type only the parameter, not the ellipsis (...).</td>
</tr>
<tr>
<td>. . . .</td>
<td>Indicates that there are (or could be) intervening or additional commands.</td>
</tr>
</tbody>
</table>

**Related Publications**

Visit our Technical Documentation Library at [http://documentation.informationbuilders.com](http://documentation.informationbuilders.com). You can also contact the Publications Order Department at (800) 969-4636.

**Customer Support**

Do you have questions about this product?

Join the Focal Point community. Focal Point is our online developer center and more than a message board. It is an interactive network of more than 3,000 developers from almost every profession and industry, collaborating on solutions and sharing tips and techniques. Access Focal Point at [http://forums.informationbuilders.com/eve/forums](http://forums.informationbuilders.com/eve/forums).

You can also access support services electronically, 24 hours a day, with InfoResponse Online. InfoResponse Online is accessible through our website, [http://www.informationbuilders.com](http://www.informationbuilders.com). It connects you to the tracking system and known-problem database at the Information Builders support center. Registered users can open, update, and view the status of cases in the tracking system and read descriptions of reported software issues. New users can register immediately for this service. The technical support section of [www.informationbuilders.com](http://www.informationbuilders.com) also provides usage techniques, diagnostic tips, and answers to frequently asked questions.

Call Information Builders Customer Support Services (CSS) at (800) 736-6130 or (212) 736-6130. Customer Support Consultants are available Monday through Friday between 8:00 a.m. and 8:00 p.m. EST to address all your questions. Information Builders consultants can also give you general guidance regarding product capabilities. Please be ready to provide your six-digit site code number (xxxx.xx) when you call.
To learn about the full range of available support services, ask your Information Builders representative about InfoResponse Online, or call (800) 969-INFO.

**Information You Should Have**

To help our consultants answer your questions effectively, be prepared to provide the following information when you call:

- Your six-digit site code (xxxx.xx).
- Your WebFOCUS configuration:
  - The front-end software you are using, including vendor and release.
  - The communications protocol (for example, TCP/IP or HLLAPI), including vendor and release.
  - The software release.
  - Your server version and release. You can find this information using the Version option in the Web Console.
- The stored procedure (preferably with line numbers) or SQL statements being used in server access.
- The Master File and Access File.
- The exact nature of the problem:
  - Are the results or the format incorrect? Are the text or calculations missing or misplaced?
  - Provide the error message and return code, if applicable.
  - Is this related to any other problem?
- Has the procedure or query ever worked in its present form? Has it been changed recently? How often does the problem occur?
- What release of the operating system are you using? Has it, your security system, communications protocol, or front-end software changed?
- Is this problem reproducible? If so, how?
- Have you tried to reproduce your problem in the simplest form possible? For example, if you are having problems joining two data sources, have you tried executing a query containing just the code to access the data source?
Do you have a trace file?

How is the problem affecting your business? Is it halting development or production? Do you just have questions about functionality or documentation?

User Feedback

In an effort to produce effective documentation, the Technical Content Management staff welcomes your opinions regarding this document. Please use the Reader Comments form at the end of this document to communicate your feedback to us or to suggest changes that will support improvements to our documentation. You can also contact us through our website http://documentation.informationbuilders.com/connections.asp.

Thank you, in advance, for your comments.

Information Builders Consulting and Training

Interested in training? Information Builders Education Department offers a wide variety of training courses for this and other Information Builders products.

For information on course descriptions, locations, and dates, or to register for classes, visit our website (http://education.informationbuilders.com) or call (800) 969-INFO to speak to an Education Representative.
Introducing WebFOCUS RStat

WebFOCUS RStat is a statistical modeling workbench embedded in a WebFOCUS desktop product, such as App Studio. It allows you to perform common statistical and data mining tasks, and develop models that can be deployed as scoring applications on every platform. RStat enables data miners and Business Intelligence developers to collaborate with the same tools used to access, manipulate, or transform data, develop predictive models, and create and deploy scoring applications along with associated reports to any worker within their organization.

In this chapter:

- Highlights of WebFOCUS RStat
- Data Mining With WebFOCUS RStat
- RStat Architecture

### Highlights of WebFOCUS RStat

WebFOCUS RStat includes:

- An intuitive user interface in a WebFOCUS desktop product to access the most widely used statistical and data mining models: Decision Trees, Neural Networks, Linear and Logistic Regressions, Random Forests, Support Vector Machine, Boosting, Association Rules, and KMeans, Ewkm, Hierarchical, and BiCluster clustering.

- Ability to prepare the data, for example, extract it from any data source, manipulate and transform fields to prepare data for analysis, and perform the modeling in a WebFOCUS desktop product.

- Access to a simple web form so that you can, with the click of a button, generate scores and predictions without having to know anything about data mining and modeling.

- Ability to perform common data exploration tasks and various distribution tests in order to determine the most appropriate modeling technique.

- Ability to explore the data visually with a robust set of charts and interactive visualizations.

- Incorporate predicted values and scores into any WebFOCUS report so that users can benchmark their intuitions and expert opinions to ensure that they have taken all factors into consideration when making decisions.
- RStat is extensible through scripts and open source to leverage the numerous packages developed by industry experts and scientists to perform highly complex analyses and rapidly deploy applications.

- A 64-bit trial version of RStat is offered.

- RStat is for use with full installations of App Studio on local machines. Network installations of App Studio do not support RStat.

- RStat 2.0 requires and is installed on the same computer as App Studio 82xx and higher, running on Microsoft Windows 64-bit. Supported operating systems include Windows 10, Windows 8.1, Windows 7 Enterprise, Professional, or Ultimate editions, and Windows 2012, 2012 R2, 2008, or 2008 R2 Server editions. For information on software support, see the WebFOCUS App Studio Installation and Configuration Guide.

Models are deployed to the WebFOCUS Reporting Server, 64-bit, running on Microsoft Windows, Linux, UNIX, and other operating systems.

---

**Note:** In most cases, each screen has been cropped to remove the Windows title. This is due to varying appearance because of different operating systems.
Data Mining With WebFOCUS RStat

Until recently, data mining was a branch of Business Intelligence (BI) used only by expert statisticians. Few people understood their mathematical methods. Thus, the statistical results were communicated only to upper management. But, increasingly, operational users are required to make decisions and take actions based on their expectations about the future.

Most reporting applications do a good job of recording what has happened. But that is just a rear-facing view of the business. They do not provide guidance about future actions. To compete effectively in the business world today, decision makers at every level of an organization need access to predictive modeling applications. Police officers need to determine where crimes are likely to occur so patrol cars can be in areas where they are most needed. Marketing managers need to predict who is most likely to respond to an email blast or ad campaign. Auto insurance personnel need to create risk profiles based on the likelihood of certain individuals to file claims.

RStat bridges the gap between the rear and forward-facing views of business operations. It offers the first fully integrated BI and data mining environment for developing predictive models and distributing scoring applications so operational users can make decisions with confidence instead of relying on their instincts.

Benefits of a Fully Integrated Environment

Data mining is the technique of identifying patterns and relationships within large databases through the use of advanced statistical methods. It extracts historical data and then applies statistical techniques to build a model. Traditionally, highly trained analysts and statisticians built these models. But unless their results are widely deployed, they end up as isolated research products, doing little good for the business.

A scoring application deploys analytic models for repeated use on new data sets by non-technical users to support decision-making. For example, a marketing analyst would use a scoring application to score new mailing lists in order to screen for the best possible respondents. In simple terms, the scoring application labels a prospect as either good or bad.

Statisticians spend much of their time extracting and querying data. But, by working in the same BI environment, developers can create queries that statisticians reuse to create models. The statisticians can compile their models as standard WebFOCUS functions that BI developers turn into WebFOCUS scoring applications, deployable on any platform. There is no need to work with multiple tools or pay for extra licenses. By unifying BI and data mining environments, RStat reduces licensing costs by consolidating software tools. This has the corollary effect of simplifying maintenance and making optimum use of IT resources.
RStat Architecture

RStat is built on the open source R engine. R is well known as the most powerful and flexible statistical programming language available. It is used by over one million analysts worldwide, is taught in countless universities, and has more than one thousand packaged extensions for various types of analysis exercises.

R is a powerful scripting environment designed for technical users. RStat integrates R within App Studio. This user interface provides an easy and intuitive workflow. It incorporates the top most commonly used data mining routines, including regression decision trees, neural networks, clustering, association, support vector machine, and other algorithms that are familiar to business and engineering students. A wide variety of users, including statisticians, business analysts, and other professionals, can easily use RStat to develop models.

The unique capability to compile models as native WebFOCUS functions directly in RStat enables an organization to spread data mining benefits to operational users. Those users will find the scoring application familiar and accessible. Clicking on a web-based form, users can generate scores and predictions to support decision-making without having to know anything about data mining and modeling and without the need to install R or other scoring engines in the operational environment. This ease of use and scalability will drive organizations to adopt more and more users in the predictive modeling environment.

To develop data mining applications, you install RStat on top of WebFOCUS App Studio. The installation program installs a full R environment. Once RStat is installed, you can use App Studio to access data through any Reporting Server. Once a model is built, the model can be compiled and deployed on any WebFOCUS Server. Users do not need to have R or RStat installed on the production environment in order to run scoring applications.
This chapter reviews the installation prerequisites, as well as the system requirements and instructions for installing RStat. The installation walks you through the installation of a full R environment and the additional required GNU Open Source Gtk+ Runtime Environment and GGobi packages.

R is a language and environment for statistical computing and graphics. RStat is a package extension that resides within the R environment and takes advantage of many other R packages.

All of these packages are GNU Open Source free software licensed under the GNU General Public License (GPL).

As required by the GNU General Public License (GPL), a copy of the RStat source code is available upon request. You may contact Information Builders Customer Support to request a copy of the RStat source code. Information Builders will not support modifications to the source code provided.

In this chapter:

- Installation Prerequisites
- RStat Installation

Installation Prerequisites

RStat deployment on a Windows machine requires the installation of a C compiler.

The open source compilers made available on the IBI downloads site (MinGW) can be installed on the Windows machine. The RStat installation package can launch the installation of MinGWxx.exe, where xx is the bit version (that is, 64). If you prefer the RStat install to kick off the MinGW installation, then you need to download the MinGW installation package. For more information, see *How to Download the Mingw64.exe File* on page 30. The MinGW install can also be manually launched separately after the RStat install completes.

**Note:** The MinGW install needs to be installed on the machine where the WebFOCUS Reporting Server is installed. Depending on the architecture, it does not necessarily need to be on the same machine as RStat.
RStat Installation

The following section lists the requirements needed to run RStat and describes how to install the software.

RStat System Requirements

In addition to the core App Studio product, RStat requires:

- At least 400 megabytes of disk space.
- 4 gigabytes of RAM recommended.

Note: RStat uses in-memory computing. Therefore, the more RAM that you have available, your system performance will be improved. For example, you will be able to work with larger data sets.

The amount of memory required by RStat varies depending on your operating system. If the operating system is 32-bit, then 64-bit R is 2 GB. If the operating system is 64-bit, it is 8 GB for 64-bit R.

Note:

- For non-Windows platform deployments, the MinGW compiler, which can be downloaded from the Information Builders Technical Support site, is not required.
  - UNIX/Linux systems have a native compiler.
  - For AS400/iSeries systems, there is a special compiler option that needs to be enabled. For more information, see your system administrator.

- The Readme regarding the compiler can be downloaded with the compilers.
  - In the Licensed RStat section, you can download three files (64-bit compiler, compiler source, and ReadMe).

Procedure: How to Install RStat

You can obtain the RStat installation from the Information Builders website (http://techsupport.informationbuilders.com) or you can receive it on CD (which is available upon request).

1. Execute the installation program:

   - For a CD installation, insert the installation CD in the CD drive and follow the instructions on the screen. If the installation program does not start automatically, run the setup.exe application in the root directory of the CD.
For a downloaded installation, run the downloaded file and respond when prompted to unpack files needed during the installation. When the files are unloaded, the actual installation begins.

The InstallAnywhere dialog box opens, as shown in the following image.

![InstallAnywhere dialog box](image)

The Introduction page displays, as shown in the following image.

![Introduction page](image)

2. Click Next to continue.
3. If a previous version of RStat is detected, a dialog box appears, asking if you want to overwrite the existing version.
   
a. Click **Yes, Overwrite** to overwrite the RStat version that is already installed and continue with the new installation.

b. If this dialog box does not display, proceed to step 4.

The License Agreement displays.

4. Click **I accept the terms of the License Agreement**, and then click **Next**.
5. On the Choose Install Folder page, accept the default location (C:\ibi\Rstat20) as the destination folder or click Browse to change the destination, as shown in the following image.

Note: It is recommended that you install RStat within your existing ibi directory structure.

6. If you previously downloaded the MinGW compiler, select or type the full path in the MinGW Compiler Install field or browse to the location.

For information on downloading the MinGW compiler, see Installation Prerequisites on page 17.

Note:

- Uncheck the MinGW Compiler Install check box if it is already installed. In addition, this installation assumes that you have not installed the MinGW Compiler. For more information, see Installing the MinGW Compiler on page 25.

- When performing an upgrade installation and searching for the mingw64.exe file, you can exit the Browse window and click Browse again to bring you to the drive and directory where mingw64.exe is located.
As of WebFOCUS Version 7.7.03, the WebFOCUS Reporting Server is no longer packaged with the MinGW Open Source compiler. To facilitate the process for the compiling, Information Builders has created an installation package for the 64-bit compiler. The download facility allows you to also download the source code for the compilers. Note that the source code is the same for both compilers.

**Note:** The source code for the compiler is not required to use RStat. It is only included for customers who may wish to rebuild the compiler.

<table>
<thead>
<tr>
<th>Compiler Information</th>
<th>Executable</th>
</tr>
</thead>
<tbody>
<tr>
<td>64-bit compiler for 64-bit <strong>WebFOCUS</strong></td>
<td>Mingw64.exe</td>
</tr>
<tr>
<td>Reporting Server</td>
<td></td>
</tr>
<tr>
<td>Compiler source code for 64-bit compilers</td>
<td>Mingw-w64-v1.0-snapshot-20110809.tar.bz2</td>
</tr>
</tbody>
</table>

The process to install the compiler is as follows:

- Download the compiler: 64-bit compiler for the 64-bit WebFOCUS Reporting Server. For more information, see *How to Download the Mingw64.exe File* on page 30.

- Double-click the Mingw64.exe file. For more information, see *Installing the MinGW Compiler* on page 25.

- Follow the prompts and accept the defaults.

**Note:** When installing the compiler during the licensed RStat installation, if a 64-bit compiler is required, that step can be completed separately after RStat is installed. In this case, the check box for MinGW Compiler Install should not be selected.

7. Click Next.
8. Click **Install**.

The installation automatically installs a full R environment and the additional required Gtk+ Runtime Environment and GGobi packages. The Gtk+ Runtime Environment is the toolkit for creating graphical user interfaces in which the RStat user interface has been developed. GGobi is an open source visualization program for exploring high-dimensional data. It provides highly dynamic and interactive graphics, as well as graphics, such as the scatter plot, bar chart, and parallel coordinates plots.
The installation progresses, as shown in the following image.
When the installation is complete, the Install Complete page displays, as shown in the following image.

9. Click Done to complete the RStat installation.

**Installing the MinGW Compiler**

During the installation, you have the option to launch the MinGW compiler installation. If you have downloaded the Mingw64.exe, you can specify its path during the installation. Then, at the end of the RStat installation, you will encounter the following screens to proceed with the MinGW Compiler installation.
Procedure: How to Install the MinGW Compiler

Note: This information can be used when installing the MinGW compiler on the machine where the WebFOCUS Reporting Server is installed.

1. From within the Installation package, click Next on the Installation Wizard for MinGW screen, as shown in the following image.

2. The Choose Destination Location screen displays, as shown in the following image.
Accept the default path (C:\ibi) or specify a new path to which to install the program.

3. Click Next.
4. The files begin to copy, as shown in the following image.

5. Click Next.

The compiler installs.
6. Once you reach the InstallShield Wizard Complete screen, click *Finish*, as shown in the following image.
Procedure: How to Download the Mingw64.exe File

1. Log on to https://techsupport.informationbuilders.com, as shown in the following image.

2. In the Quick Links section, click My Downloads and Shipments.
3. Log in with your Customer ID.
4. For Customer Downloads/Shipments, enter a Customer Site Code or Serial Number then click **Customer Orders**, as shown in the following image.

![Customer Downloads/Shipments](image1)

5. On the Software Downloads/Shipments page, scroll down to RStat License, as shown in the following image.

![RStat License](image2)
Note:

- The downloads available to you may differ depending on what products are associated with your Customer ID.
- The RStat trial is valid for 60 days.

6. Expand the RStat category.

7. Scroll down to Windows Compilers and select Download.
   The Download Registration Form displays.

8. On the Software Download Agreement page, select I agree to the Terms of the Agreement and click Continue.

9. Select the MinGW compiler that is compatible with your WebFOCUS Reporting Server, as shown in the following image.

<table>
<thead>
<tr>
<th>File Name</th>
<th>File Size (bytes)</th>
<th>Download</th>
</tr>
</thead>
<tbody>
<tr>
<td>mingw-w64-v1.0-snapshot-20110809.tar.bz2</td>
<td>4,617,950</td>
<td>FTP HTTP</td>
</tr>
<tr>
<td>mingw32.exe</td>
<td>45,945,494</td>
<td>FTP HTTP</td>
</tr>
<tr>
<td>mingw64.exe</td>
<td>57,655,059</td>
<td>FTP HTTP</td>
</tr>
<tr>
<td>ReadMe.doc</td>
<td>28,160</td>
<td>FTP HTTP</td>
</tr>
</tbody>
</table>

Note: If your WebFOCUS Reporting Server is 64-bit, download the 64-bit MinGW compiler.

10. Click FTP or HTTP to indicate how you would like to download the compiler.

11. When the File Download dialog box displays, click Save to save the file.
   Once the file is saved to your local drive, make a note of its location as you will navigate to and select it during the RStat installation.
Getting Started With RStat

Now that you are ready to start RStat, there are two approaches you can take. You can open RStat and load an existing data source or call a procedure (FEX) to extract a new sample, or you can Run to RStat within the Report canvas and open RStat with your current report output loaded.

Note: Depending on data, variables, and other options selected, results may vary from the examples provided throughout this manual.

In this chapter:

- Starting RStat With Data Loaded
- Starting RStat Without Data Loaded in App Studio
- Loading Data From Within RStat
- RScripts in RStat

Starting RStat With Data Loaded

You can pass any data generated by the Report canvas directly into RStat. The Report canvas is accessible in App Studio. You can develop your data, join various data sources, build virtual fields, and create data sets to pass to RStat for modeling activities. Run RStat extracts the data defined within your report and loads it directly into RStat. Within each report, you can customize the name of the HOLD file to be created and the path where it should be placed. You can customize this for every FEX or run with the defaults.
There are several RStat specific options available from the Modeling tab in the Report canvas in App Studio, such as Launch RStat, Parameters (RStat Model Configuration), and Sampling (RStat Data Sampling). The following image shows the Report canvas in App Studio with the Modeling tab exposed.

Model configuration allows you to name the HOLD file that will be used to pass the model data from the Report canvas into RStat and define the path where the HOLD file should be placed. You can set your own HOLD path default to be used in all FEXs. The model configuration settings are saved in the FEX when you customize them and each time you execute Run RStat.

Procedure: How to Use a Model Configuration in App Studio

1. Create a report containing the fields that you want to use in your model.
   The fields in the report are brought into RStat through a HOLD file using the steps below.
2. On the Modeling tab, in the Modeling group, click Parameters.
3. Define the Hold File Name. The default name is HOLD. To retain the HOLD file for further access, define a file name that is unique to this data extract.

4. Define the Hold File Path. Set the Hold File Path to be used for this FEX to the app path where you will be performing your modeling work. This will initially be set to the default path for your App Studio configuration.

   Click Default to set the currently defined Hold File Path for this FEX to the default to be used for all FEXs. You can always overwrite the default for each individual FEX.

5. Click OK to finish creating your model configuration.

Procedure: How to Launch RStat in App Studio

To launch RStat, on the Home tab, in the Modeling group, click the Predictive Modeling command.

Using WebFOCUS RStat for Predictive Analytics 35
When RStat opens within App Studio, two windows are opened. RStat opens in front within the minimized R Console window. The R Console window is open for user interaction and custom R scripting. The fields that you define on the Report canvas are imported into RStat in the order specified in your report.

**Starting RStat Without Data Loaded in App Studio**

You can access RStat directly by clicking Predictive Modeling, on the Home tab, in the Modeling group, as shown in the following image.

RStat runs within the R Console. When RStat opens within App Studio, two windows are opened. RStat opens in front within the minimized R Console window. The R Console window is open for user interaction and custom R scripting.

**Loading Data From Within RStat**

If you have entered RStat without data loaded, or if you would like to load new data from within RStat, you can directly access the following data types for modeling:

- FEX - loads the results of a selected FEX (only available in 32-bit RStat)
- CSV
- RData
- Library
- RScript

**Note:**

- A tooltip is available for all buttons listed.
- FEX functionality is disabled in 64-bit versions of RStat. This is due to the conflicts encountered when calling 32-bit DLLs from a 64-bit application.
Procedure: How to Load Data From a FEX

1. On the Data tab, in the Source area, select FEX, and then click the Filename button.
The Select FOCEXC dialog box opens, as shown in the following image.
2. Select the Reporting Server and application path containing your FEX. The available FEXs are displayed in the right pane. An example of the Select FOCEXEC dialog box is shown in the following image.

![Select FOCEXEC dialog box](image)

3. Select the FEX you want to load and click OK.
The file name of the FEX selected appears in the filename control, as shown in the following image.

4. To run the selected FEX and load the data, click *Execute* on the toolbar.
The following image shows the results of running the selected FEX.

The FEX load process generates a hold file from the last TABLE FILE (Report canvas object) within the selected FEX. This hold file is named FEXNAME.CSV. In our example, we selected AB_TRAINDATA.FEX and App Studio generated and loaded AB_TRAINDATA.CSV.
With FEX selected as the Type option, you can see the path of the FEX that generated the CSV. If you select the CSV option, RStat shows that the generated CSV has been loaded, as shown in the following image.

This offers the added benefit of allowing you to access the current data extract in the CSV HOLD file or to refresh the HOLD file from the data source by re-executing the FEX.

**Note:** Each time you execute the FEX extract, the existing HOLD file will be overwritten with the new data extract. If you want to retain versions of the extract, rename it or save the file under a new name. You can use the Export function from RStat to create a copy of the extract under a new name.
RScripts in RStat

An RScript can run plots, charts, summaries, model techniques, or even be used to execute scoring functionality using R. As of RStat 1.3.1, RScripts can run using RStat, and depending on the technique, the RScript, if used to create a scoring routine, can be converted to a C routine like a native RStat scoring technique. For both capabilities, the package referenced by the RScript needs to be loaded to the RStat environment in the R workspace. Without the R package, the script will not run. If the intent of the RScript is to create a model and output a C file for the scoring routine to be used by the WebFOCUS Reporting Server, then the feature requires a few naming conventions and a few other items to be existent.

Running an RScript

To run the RScript, note the following:

- You need the package required by the RScript. This step is not covered in the manual.
- You need the RScript, which is run in the R environment by the WebFOCUS Reporting Server.
- R needs to be on the machine or configured with the WebFOCUS Reporting Server.
- Memory usage depends on the R architecture (32-bit or 64-bit) and the server memory. WebFOCUS will take the request and pass it on if the application is designed to do so. The WebFOCUS Server will take the output and present it in the format desired by the application.

Procedure: How to Run an RScript

1. If starting a new process, in App Studio, on the Home tab, in the Modeling group, click Predictive Modeling to launch RStat. Otherwise, click the New button on the RStat GUI.

2. On the Data tab, in the Source area, select the RScript option.
3. Click the Filename list to browse for an RScript file.

4. If an RScript file with an extension of .R or .r is loaded, click the Execute button. The command in RScript will be loaded in R but no code will be shown in the R GUI.
The following example shows an RScript that is executed to create a plot.
**Note:** Once the RScript button is active (selected), the user can upload an RScript. When a non-script file is selected (for example, a .csv file or a .txt file), a message appears, as shown in the following image. The error message will also be shown in the RStat status bar and log file. The RScript button is looking for a R code file with an extension of .R.

---

**Running an RScript in Order to Create a C File**

To create the Scoring Routine C file from an RScript, note the following:

- You need the package required by the RScript. This step is not covered in the manual.
- You need the RScript, which is run in the R environment by the WebFOCUS Reporting Server.
- R needs to be on the machine or configured with the WebFOCUS Reporting Server.
- The scoring routine technique has to be a native to RStat. For example, Decision Tree, Regression, Survival, Neural Networks, and Clustering.
The following variables must be assigned:

- `crs$dataset`: the dataset used to build the model in the script.
- `crs$rpart`: model name defined in `crs`: glm for regression, rpart for decision trees, and hcluster for cluster.
- `pmml.cmd`: in the format: `pmml.cmd <- "pmml(crs$rpart, dataset=crs$dataset)"`. 

Before clicking `Export`, you must first click the Model tab. Otherwise, it will export a `.csv` file since the RScript button resides on the Data tab. In addition, the file name needs to be assigned by the user.

**Procedure: How to Run an RScript to Create C File Output**

1. If starting a new process, in App Studio, on the `Home` tab, in the `Modeling` group, click `Predictive Modeling` to launch RStat. Otherwise, click the `New` button on the RStat GUI.

2. On the Data tab, in the Source area, select the `RScript` option.
3. Click the Filename list to browse for an RScript file.

4. If an RScript file with an extension of .R or .r is loaded, click the Execute button. The command in RScript will be loaded in R but no code will be shown in the R GUI.

5. If you want to export the model built in the RScript to a C routine, the following variables must be assigned before exporting (take decision tree as example):
   - crs$dataset: the dataset used to build the model in the script.
   - crs$rpart: model name defined in crs: glm for regression, rpart for decision trees, and hcluster for cluster.
   - pmml.cmd: in the format: pmml.cmd <- "pmml(crs$rpart, dataset=crs$dataset)"

6. Before clicking Export, you must first click the Model tab. Otherwise, the execute action will export a .csv file since the RScript button resides on the Data tab.
7. Provide the name for the C file to be created.

Reference: Usage Notes for RScript to Create Scoring Routines

If you choose not to load the data set in RStat but want to export a C routine, you can load an .RData file or use a saved pmml/XML file to export the C routine in the R console with RStat opened. RStat will check crs$dataset before exporting any file. Thus, non-data exporting only can be done in the R console with an RStat session opened.

The .RData file saves the model summary as an R workspace. The example below shows related codes for the decision tree.

```r
decTree <- rpart(PRICE_1991~., dataset)
save(decTree, file = 'wine_training_rpart.rdata')
```

The two code lines above should be included in the user script.

The following code will be the script uploaded by the RScript button in RStat 1.4 and higher, which first loads the .RData file to pass the model summary to crs$rpart, and then further defines the 'pmml.cmd' command.

```r
load('wine_training_rpart.rdata')
crs$rpart <- decTree
pmml.cmd <- "pmml(crs$rpart)"
```

In the R console, executing the following code will export the C routine:
con <- file("dec tree.c", open="w")
cat(pmmltoc(toString(eval(parse(text=pmml.cmd))),
            name="dec tree", includePMML=TRUE, includeMetaData="",
            exportClass=FALSE), file=con)
close(con)

If you generate a pmml/XML file and want to export a C routine, you need to execute the following code lines with RStat open:

cat(pmmltoc(toString("dec tree.xml")
           , "dec tree", TRUE, "", FALSE), file = "dec tree.c")
WebFOCUS RStat is an integrated interface within App Studio that enables you to explore and transform data and build models that can be used to develop and deploy scoring applications.

In this chapter:

- Variable Definitions and Sampling
- Data Exploration
- Hypothesis Testing
- Data Transformation
- Clustering

Variable Definitions and Sampling

When the RStat tool opens for a new model to be built or modified, it appears as shown in the following image. Note that the data from the procedure (FEX) developed in the Report canvas is loaded into RStat and is ready for modeling. You can also open RStat without any data loaded (see Creating a Scoring Application on page 101 for more information).
The first tab of RStat, the Data tab, displays the variables from a CSV file that are loaded into RStat.

The main display features of RStat are described in the following table.

<table>
<thead>
<tr>
<th>Display Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Menu Bar</td>
<td>Displays the RStat menus, such as Project, Tools, Settings, and Help.</td>
</tr>
<tr>
<td>Action Toolbar</td>
<td>Displays the actions available in RStat, such as Execute, New, Open, Save, Export, Stop, and Quit.</td>
</tr>
<tr>
<td>Data Tab</td>
<td>Displays the options to load a data set. For more information, see Data Source Selection on page 54.</td>
</tr>
<tr>
<td>Explore Tab</td>
<td>Displays the options to explore the data to identify how it is distributed. For more information, see Data Exploration on page 57.</td>
</tr>
<tr>
<td>Display Feature</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Test Tab</td>
<td>Displays the options for hypothesis testing, including Distribution Test (Kolmogorov-Smirnov and Wilcoxon Signed Rank Test), Location &amp; Variance Test (Wilcoxon Rank Sum, T test, and F test), and Correlation Test (Chi-SQ Test). For more information, see <em>Hypothesis Testing</em> on page 86.</td>
</tr>
<tr>
<td>Transform Tab</td>
<td>Displays the options to transform the data in various ways. For more information, see <em>Data Transformation</em> on page 91.</td>
</tr>
<tr>
<td>Cluster Tab</td>
<td>Displays the options to build clusters for data sets.</td>
</tr>
<tr>
<td>Associate Tab</td>
<td>Displays the options to create association rules for the data.</td>
</tr>
<tr>
<td>Model Tab</td>
<td>Displays the options to build predictive models.</td>
</tr>
<tr>
<td>Evaluate Tab</td>
<td>Displays the options to evaluate the models.</td>
</tr>
<tr>
<td>Log Tab</td>
<td>The corresponding R Code appears in the Log tab. This enables you to review the R commands that perform the corresponding data mining tasks.</td>
</tr>
<tr>
<td>Status Bar</td>
<td>Indicates when an action is completed.</td>
</tr>
</tbody>
</table>
Data Source Selection

The following options are available on the Data tab in RStat. You may also access the Data tab through the Tools menu.

Source

Different options may be available depending on the data type.

- **FEX.** Loads the data from a FEX.

- **CSV File.** Loads data from a comma-separated value (CSV) file. When CSV File is selected, the following options become available:
  - **Filename:** Indicates the data file that is currently in use.
  - **Separator.** Specifies the separator type. For example, ",", ",", and so on.
  - **Decimal.** Specifies a decimal character. For example, "," or ",". The character that you specify is used in the file for decimal points and supports international currency differences. For example, 30.00 versus 30,00.
  - **Header.** Allows you to indicate whether the first row contains column headings.

- **RData File.** Loads data from an RData file (usually binary). When RData File is selected, the following options become available:
  - **Filename.** Enables you to select a data file.
  - **Data Name.** Allows you to load different data frames of the R data file. Data frames are collections of individual observations (rows of data) across many variables (fields). They are analogous to the SAS or SPSS data sets that organize the data set for statistical analysis in a cases by variables matrix. For example, rows across multiple columns. You can have multiple data frames in one R data file.
  - **Library.** Enables you to select a supplied data set from the R library. When Library is selected, the following option becomes available:
    - **Data Name** Opens a list, which you can use to select a library file.

- **RScript.** Provides the capability of running plots, charts, summaries, and model techniques. It can also be used to execute scoring functionality using R.
  - **Filename.** Enables you to select a data file.
Partition

Partitioning splits the single data set into two data sets, a training data set used for analysis and modeling, and a test data set used to evaluate how well a model performs. It is a common practice to test models on new data, different from the data used to create the model.

You can define the partition size either as a percentage of the total records or as an exact number of records. Changing the percentage will automatically change the count and vice versa.

- **Percentage.** The default is 70%. The sample randomly chooses 70% of the data for a selected data set.

- **Count.** Displays the number of records that will be included in the sample based on the selected percentage. You can manually specify the number of records, which in turn will change the percentage.

- **Seed.** Numerical value used to initialize a random sampling algorithm or to establish a starting point in a table of random numbers. By using the same seed number, you will generate exactly the same sample. You can click Seeds to update the seed with a random number.

- **View.** Opens the Data Viewer window, enabling you to view data. An alternate way to view data is to run the WebFOCUS procedure (FEX) that you use to load the data.

---

- **Input and Ignore Buttons**

---

Using WebFOCUS RStat for Predictive Analytics  55
You can set a group of variables to a single role using the Input and Ignore buttons by:

- Selecting the variables to be set from the Variable Grid. To select multiple variables, hold down the Ctrl key while clicking each variable, or the Shift key to define a range. You can select all variables within the grid by clicking one of the variables and then pressing Ctrl+A.

- Clicking the green Input button to define all selected variables as input, or the red Ignore button to define all selected variables as ignored. This defines a portion to be used as training data and opens with the selected data set loaded. RStat presents nine tabs that reflect the standard modeling workflow. The Data tab shows the variables and the roles each will play in building the model.

- **Target Data Type**

  The data type of the target variable determines the type of modeling available and the specific algorithms that will be used within the modeling process. The data type is defined based on the type of data RStat identifies and the quantity of unique values found in the actual data. In RStat, data types are defined as:

  - **Auto.** This option is selected by default and uses the rules that were most recently defined.

  - **Categoric.** Any character data or any numeric data with 10 or less unique values.

  - **Numeric.** Any numeric data with more than 10 unique values.

  - **Survival.** Allows the user to run a Survival Model (that is, Cox Proportional Hazards or Parametric). A Time variable and a Status variable must be selected from the list of variables.

  The Target option allows you to override these heuristic settings:

  - Auto will use the previously defined rules.
  
  - Categoric will handle character and numeric values as categories.
  
  - Numeric will assign a unique numeric value to each categoric value.

  - **Variable Roles.** Each variable can have only one role.

    - **Input.** This is the exploratory (independent) variable(s) where presence or degree determines the change in the dependent variable.

    - **Target.** This is the dependent variable. In modeling, it is assumed that the dependent variable is influenced by the input variables. The model shows the degree to which the dependent variable is influenced by the input variables.
- **Risk.** Special variable in the data set that measures the amount of risk associated with each record in the data set.

- **Ident.** Identifies the variable as containing the ID for each record in the data set.

- **Ignore.** Ignores the variable for any analysis or modeling.

- **Weight.** Used to identify some observations as more important than others. There is no standard method for calculating this type of weight. This should be NULL or a numeric vector.

- **Comment.** Provides information on the type of data and the values found within the data, including the count of missing values and unique values.

### Data Exploration

The following options are available on the Explore tab in RStat. You may also access the Explore tab through the Tools menu.

**Note:** The options on the Explore tab are different, depending on the Type option you choose.

The screen that follows shows the summary statistics being executed and displayed.
Summary

Summarizes the data set and provides descriptive statistics on each variable in the data set.

- **Summary.** Includes Min, Quartiles, Mean, and Max for numeric, and top level for factors.

- **Describe.** Includes a concise description including missing, unique, sum, mean, and the lowest and highest values, frequencies, and percentages.

- **Basics.** Includes various basic measures of numeric data, including missing, min, max, quartiles, mean, sum, skewness, and Kurtosis.

- **Kurtosis.** Summarizes just the Kurtosis. This is useful for comparing all numeric variables at once. The Kurtosis measures the peakness of the distribution. The higher the peak, the more of the data variation is due to infrequent extreme observations. In the example below, Income has a Kurtosis of 1.9290161, while Age has a Kurtosis of -0.3990564. The difference in the peak is noticeable. The example is taken from the AB_Demo data set, which was used for the credit scoring application.
Note: To generate this chart, click Latticist and then, click Execute. Click Marginals in the Chart interactive panel and then deselect all other variables except for Age and Income. From the Groups/Color drop-down box, select the top row (Empty/Null) to remove any grouping. For more details, see the section in this chapter on Latticist.

- **Skewness.** Measures the asymmetry of the distribution or how a particular distribution deviates from the normal bell-shaped distribution. A negative skew indicates that the left tail is longer, that is, most of the observations are located on that side. A positive skew indicates that the right side is longer. Both Age and Income from the preceding example have a positive skew. The skewness affects the relative position of the mean, median, and mode. If the distribution is normal, for example, bell shaped, those three values are equal.

- **Show Missing.** Displays a summary of the missing values for each variable.

- **Cross Tab.** Summarizes the cross tabulations for each categoric data type, based on the underlying variable, in a contingency table. For example, all components of a specific categoric data type are presented, allowing for a high-level comparison. A total is provided for each categoric data type.
Density Plot. Allows you to view the distribution of one specific variable. Once created, the Density Plot displays the Kurtosis and Skewness, as shown in the following image.

The following options display in the Density Plot GUI:

- **Variable Selections.** Displays the variables that are available for selection and analysis, for example, Age or Income.
Operations. Displays the operations that you can perform, including:

- **Execute.** Generates a Density Plot based on your variable selection.
- **Clear.** Clears the current Density Plot, including the textual interpretation.
- **Save Plot.** Saves the current Density Plot as a .pdf file to a local drive that you specify.
- **Save Text.** Saves the interpretation as a text file to a local drive that you specify.
- **Export Log.** Saves the R commands that are used to create the analysis as a text file to a local drive that you specify.
- **Close.** Closes the Density Plot GUI.

Distributions

Displays various distribution plots for numeric and categoric variables. The chart options for numeric and categoric variables are displayed separately. The chart types for numeric and categoric variables are also different, as shown in the following image.
For numeric variables, the available charts are:

- **Box plot.** Also known as box-and-whisker diagram or plot, displays groups of numerical data through their five-number summaries, the smallest observation, lower quartile (Q1), median (Q2), upper quartile (Q3), and largest observation. If you have selected a target variable, it will be displayed on the x-axis, as shown in the following example, where Credit_Approval is a target variable. The x-axis can take only categoric or count data.
Histogram. This displays the frequencies of each observation. The frequencies are organized in non-overlapping categories (bars). The curve shows the shape of the distribution. The histogram describes the data by displaying five values, the center (location) of the data, the spread (scale) of the data, the skewness of the data, the presence of outliers, and the presence of multiple modes in the data (if any exist).
**Cumulative Histogram.** The cumulative histogram is a variation of the histogram in which the vertical axis gives not just the counts for each group, but rather gives the counts for that group plus all other groups prior to it. So, for income, it indicates what percent of total population is below or above any income level. If a categoric target variable has been selected, separate cumulative curves will be displayed for each category.

**Benford Bars.** According to Benford’s Law, also referred to as the First Digit Law, if you draw a random number from a list of real life numbers, such as income, invoice amounts, and so on, the probability of drawing a number starting with 1 is almost one-third. The larger leading digits occur with lower and lower frequency, to the point where 9 as a first digit occurs less than one time in twenty. It is a pragmatic law.
The practical implication is that people do not know this law and thus cannot manipulate data convincingly. Hence, the law is used in fraud detection. For example, a tax authority uses Benford’s Law to see if cash disbursements on company returns follow its law. If the distribution of disbursements does not follow Benford’s Law, an investigation is triggered. Insurance claims are another typical use case.

Generate a bar plot as shown in the following image. Make sure that no dependent variable is selected on the Data tab. If a dependent variable is selected, its values will be shown on the Benford chart to compare against the other variable. The graph indicates that income follows the Benford distribution. That is, the distribution of incomes starting with 1, 2, 3, and so on, follows the distributions of 1, 2, and 3 in real life.

- **Starting Digit.** The digit for which to plot the distribution. 1st through 9th are allowed.
- **Number of Digits.** The number of digits, which can be 1 or 2, that will display on the Y-axis of the plot.
- **abs.** Plots the distribution of the absolute values.
- +ve. Plots only the distribution of positive values.

- -ve. Plots only the distribution of negative values.

For categoric variables, the following chart types are available:

- **Bar Plot.** Shows the counts of observations within each category. If a target variable is selected, it will plot additional bars for each category in the target. In the following example, for each Occupational Value, there are three bars: the total number of people in this occupation, the number of people with good credit, and the number of people with bad credit.
**Dot Plot.** A dot plot is an alternative to a bar chart. According to some analysts, they are simpler and easier to interpret, as they clearly show the distribution of data. The frequencies are displayed on the x-axis and labels are displayed on the y-axis. If a categoric target is selected, separate dots will be plotted for each category.
Mosaic Chart. The mosaic chart gives a real representation of the distribution of observations within categories. Each category value is displayed as a 100 percent bar. The width of the bar indicates the frequency of observations. The wider the bar, the higher the count in this category. If a categoric target variable is selected, the height of the bars will be split proportionate to the counts of the observations that fall in each category of the target.

You have a few controls that apply to all charts within the Distributions section.

- **Clear.** Clears all check boxes of selected rows in the Numeric or Categoric table.
- **Plots per Page.** The number of plots to draw per page.
- **Annotate.** Includes numeric values within the plots.
In the following image, we have chosen to display four plots per page, and we have selected all four plots for the Income variable. When multiple plots per variable are selected, the Target variable information is ignored and only the total information for the variable is plotted.
Correlation

Correlation indicates the strength and the direction of the relationship between two variables. Correlations should be interpreted carefully, as they depend on the context. In simple terms, a correlation coefficient closer to 0 indicates no relationship, and a coefficient closer to 1 indicates a strong relationship. Positive correlation indicates that as one variable increases, so does the other. Negative correlation indicates that as one variable increases, the other decreases. Correlations are displayed in a table and a graph, storing the pairwise correlations between all numeric variables.

The following image displays the correlations in a table within the RStat output window.
**Chart Displaying the Correlations.** The color and the shape of the circles indicate the strength of the correlation. For example, Income and Age have a correlation near 0, which is represented by a full white circle. Credit_Approval and Income have a correlation of -0.38, which is represented by a pink ellipse. The dispersion of the data becomes more narrow and closer to a straight line, that is, the perfect linear relationship.
Ordered. When selected, displays the variables ordered by the strength of the correlations.

Explore Missing. When selected, displays the correlation between missing values. If the file does not contain missing values, those correlations will not be displayed and a dialog with a warning message will be displayed.

Hierarchical. A correlation between the numeric data is calculated, and then a hierarchical cluster is generated based on the correlations. The hierarchical cluster is then visualized through a dendrogram to give an idea of the groupings of the numeric variables. The length of the lines in the dendrogram provide a visual indication of the degree of correlation. Shorter lines indicate more tightly correlated variables. Once you have identified the groups of variables that are correlated, you may want to reduce the number of variables that you are including in your modeling. For instance, you can compare the Credit_Approval and Income on the dendrogram with the correlations from the prior section, and see that shorter lines correspond to higher correlations.

Method. Select the method for the computation of the correlation coefficient.

- **Pearson.** The Pearson correlation coefficient measures the degree and strength of a linear relationship between two variables with bivariate normal distributions on a scale of -1 to 1. It is the most commonly used method and therefore, is set as the default.

- **Kendall.** The Kendall correlation coefficient is a non-parametric measure of the association of two variables; that is, it does not assume that the data is normally distributed. It is used to measure correlation between rankings and cross tabulations. For example, an analyst may create ranks on income and on job qualifications for a set of individuals in a data set. The correlation test will determine whether people in the higher income ranks are likely to rank higher in qualifications.

- **Spearman.** The Spearman correlation coefficient, like the Kendall correlation coefficient, is also a non-parametric (distribution-free) measure of the association of two variables. It is like the Pearson’s correlation coefficient, but is conducted on the rankings of the original variables. Compared to the Kendall test correlation coefficient, the Spearman correlation coefficient will be less accurate if there are dislocations from the perfect ranking order in the data set.
You can use any of the three methods, Pearson, Kendall, or Spearman, to calculate the correlation coefficients.

![Variable Correlation Clusters ab_credit_training.csv using pearson](image)
Principal Components

Principal components analysis is used for variable reduction, that is, to analyze the numeric variables in the data set and indicate whether a smaller set of new uncorrelated variables can be generated and used for modeling. The potential new variables are called principal components, and usually the first two account for most of the variation in the target variable. Hence, only the components that account for most of the variation can be used for modeling. RStat does not actually generate the new PC variables. Instead, it is used to analyze which of the input variables contribute most to the components. This information helps decide which variables to include in or exclude from the analysis.
The two following images are displayed to exhibit the relationships between the principal components. The following bar chart presents the relative comparison of how much of the variation in the data is accounted for by each of the principal components. The first will account for the most, then the second, and so on.
The plot chart plots the principal component 1 against the principal component 2 (for example, the two principal components that account for most of the variation), also displaying the strength of the component variables.

**Interactive**

The Interactive options include Latticist and GGobi. These are both interactive data visualization packages.
**Latticist**

Latticist is an interactive visualization package. For more information, see the online Help by clicking *Explore* and then clicking *Latticist*. Select the *Latticist* radio button and click *Execute*. This loads all selected variables into a matrix plot, as displayed in the following image. The target variable will be used as a grouping variable. Both numeric and categoric variables can be used as grouping variables.
To remove a grouping variable in the Groups/Color drop-down list, select the top null row, as shown in the following image.
To remove any variables from the plot, select the plot type button, for example, *marginals*, and uncheck the variables to be removed.

NOTE: too many variables will result in a very slow plot, especially so for sploq (scatter plot matrix).
Click OK to display only the selected variables.
To display a scatter matrix plot, select *splom (pairs)*. Conditioning will display a separate scatter matrix for each value in the variable selected in the Condition drop-down list, as shown in the following image.
**GGobi**

Runs the R package GGobi for interactive data visualization. For more information, see the online Help by clicking *Explore* and then clicking *GGobi*.

- With the GGobi radio button selected, click *Execute*. All variables that are selected in the Data tab will be loaded, and a scatter plot will be displayed, as shown in the following image.

- All variables are displayed in the following GGobi panel. Users can interactively change the X and Y variables.
**Displaying a Matrix Scatter Plot.** A matrix scatter plot is a display of all two-by-two scatter plots for all variables in a single panel. It allows you to quickly assess all relationships in a data set.

To generate the matrix scatter plot shown here, select *Display* from the toolbar menu on the GGobi floating panel and select *New Scatterplot Matrix.*

![Matrix Scatter Plot](image)
**GGobi Parallel Coordinates Chart.** A parallel coordinates chart, also known as a profile plot, is a useful way to compare several sets of observations as a combination of different factors. It is useful to detect patterns in the data.

To generate the parallel coordinates chart shown below, select *Display* from the toolbar menu on the GGobi floating panel and select *New Parallel Coordinates Display.*
**GGobi Interactions.** From the toolbar menu, you can select *Interactions* and specify the type, for example, Brush. The brush allows you to select points on the graph and the selection will propagate to all other GGobi graphs. You can see the relationship of the selected items and all other items.

For more information on GGobi, see [http://www.ggobi.org/](http://www.ggobi.org/).
Hypothesis Testing

The Test tab allows you to do hypothesis testing and correlation analysis. RStat supports two types of statistical inferences, estimation and hypothesis testing.

- **Estimation.** Also referred to as predictive modeling, is the process of deriving expected and predicted values from observations. Decision trees, regression, and the other algorithms, which are located on the Model tab, are used to generate estimates. For example, you can estimate whether a prospect is a good target for a particular marketing campaign or you can estimate the expected sales revenues for different stores in order to determine whether store layout and product mix has impact on sales.

- **Hypothesis Testing.** Gives you a way of using samples to test whether or not statistical claims are likely to be true. For example, drug A is more effective than drug B, male customers spend less than female customers, the response rate to offer A is better than that of offer B, machine A produces more defects than machine B, individual expense reports from the southern region are greater than those from the western region, and so on.

Use Case for Hypothesis Testing

Analysts may want to determine if a marketing campaign is successful. They design a test group, which receives an offer, and a control group, which does not. The spending of both groups is tracked in the database. The hypothesis test will determine if the two groups differ significantly in their spending patterns.

Why test? In this example, analysts want to find out if the test group spends more. If the test group spends the same as the control group, they will assume that the campaign is not successful. Rarely are the expenditures of the two groups identical, so the question arises, how different must the expenditures be in order to determine if the campaign has an effect? The test statistics indicate whether the differences are statistically significant.

An image of the Test tab follows. Samples for testing can be selected in one of two ways.

- **Select a single variable from the Sample 1 drop-down box and separate the individual samples by the target variable.** This is the default method. With this method, it is assumed that the variable selected in Sample 1 contains both samples and that the samples can be differentiated by the Target variable. For example, the data set may contain the variable, Spend, and another variable, Test/Control Group. The second variable is used to classify the transactions in the Spend variable based on whether they are made by people who were subjected to a marketing campaign (test group) or not (control group). If the second variable is defined as a Target in the Data tab, you can run the tests to compare the two groups.
Note: This method cannot be used for the Correlation and Wilcoxon Signed-Rank tests. Those options will be grayed out until the user selects the option below.

- The two samples are in the data set as two different variables. In other words, the data can contain the variables Spend Test Group and Spend Control Group. To select the two variables for testing, uncheck the Group by Target check box. This will make the Sample 2 drop-down box active. Select the second variable from the Sample 2 drop-down list. This method applies to all tests in the Test tab.

Note: The drop-down boxes can contain only numeric variables.

The following image shows an example of using a T test to identify two samples, people who have been approved for credit and those who have not, and whether their income differs significantly between the two groups.
The Test tab, which contains the various tests supported by RStat, is divided into three tabs: Distribution Test, Location&Variance Test, and Correlation Test, as shown in the following image.

These tabs organize the different tests into logical categories. This section reviews these tests, and introduces the Chi-SQ GUI.

The types of tests include:

**Distribution Test.** These non-parametric tests make no assumptions that the underlying distribution is normal. They are suitable for many types of data that do not follow the normal distribution, for example, ranked and cross-tabulated data.

- **Kolmogorov-Smirnov.** The test compares the two distributions by being sensitive to both the location (mean, median, mode) and the shape (spread) of the distribution.

- **Wilcoxon Signed Rank Test.** This test differs from other distribution tests because it is used on two related samples, such as matched pairs, before and after tests, and repeated measurements on the same fields.
Location & Variance Test. These tests are used to determine if there is a shift of the distribution. The non-parametric test (Wilcoxon Rank Sum) makes no assumptions that the underlying distribution is normal. They are suitable for many types of data that do not follow the normal distribution, for example, ranked and cross-tabulated data. The parametric tests (T test and F test) make strong assumptions that the underlying distribution is normal, for example, having a bell-shaped curve.

- **Wilcoxon Rank Sum.** This test, also known as the Mann-Whitney-Wilcoxon test, is analogous to the two-sample T test, but is performed on the rankings of the combined data sets instead of on the actual measure. It is based on median and non-normal distributions. If the observation rankings are not different, then the samples are not different. Because it is performed on the rankings, it is more sensitive about the location of the distribution, that is, to the median (not the mean as in the T test).

- **T test.** Is the most commonly used test to determine whether the means of two normally distributed samples are of equal sizes. It assumes that the distribution is normal. The mean is a measure of the location of the distribution. If the two populations are normal (bell-shaped curves) and their means are different, then the two bell-shaped curves will be offset from one another, indicating that the two samples are different. If the means are equal, the bell-shaped curves will overlap.

- **F test.** Is used to determine if the standard deviations of two samples are the same. It assumes that the distribution is normal. If the standard deviations are not the same, the bell-shaped curves will be different for the two samples. If the samples have the same standard deviations, then a T test can be conducted to test if the means are equal. The test is also referred to as a test on the variance of two samples and is used in analysis of variance (ANOVA).

Correlation Test. Determines if there is a linear relationship between two variables. It also measures the strength and direction of the relationship. Correlation analysis does not test whether two samples are different. Only the Pearson correlation test is performed in the Test tab.

- **Chi-Sq Test.** Tests for any observed differences that exist within sets of categorical data. You can run this test from the Correlation Test tab, which is located on the Test tab.

The Pearson Chi-Squared test is a statistical test that is applied to categorical data, which is organized in sets. The test evaluates the likelihood that any observed difference arose between the sets by chance.
Note: To activate the Chi-SQ GUI, access the Correlation Test tab on the Test tab and click Execute. The Textview pane displays the results of your test.

Once you make a selection for Goodness-of-fit Test or Independence Test, the relevant variable drop-down lists are activated.

Test Type. Allows you to select a test type (Goodness-of-fit Test or Independence Test). The Goodness-of-fit Test is selected, by default.

- **Goodness-of-fit Test.** A test that establishes whether an observed frequency distribution, which is a table that displays the frequency of potential outcomes in a sample, differs from a theoretical distribution.

- **Independence Test.** A test that uses two categorical variables from a single population. The Independence test is used to determine whether a significant relationship exists between the two variables.

Select Variables. Depending on the test type with which you are working, you can select one variable (Goodness-of-fit) or two variables (Independence Test). The list of available variables is based on the data set you are using.
Operations. These are the functions that you can perform based on your test type and variable selections.

- **Execute.** Executes the Chi-Square test based on your selections. The results are displayed in the Textview dialog box, and a record of the activity is posted to the log.

- **Save Text.** Enables you to save the resulting text of your Chi-Square test to a text file on your local drive. This is useful for archival or comparative purposes.

- **Save Log.** Enables you to save the log file, which records all of the different R commands executed during the various tests, to a text file on your local drive.

Data Transformation

The following options are available on the Transform tab in RStat. You may also access the Transform tab through the Tools menu.

**Note:** Options may vary depending on the type of data that is selected.

Data transformation allows users to derive new variables from existing ones. The transformation process can change the scale of the variables, the grouping of the values, and the type of the variable. The Transform tab also allows you to impute missing values, for example, replace the missing values with new values. As with transformations, a new variable with the imputed values will be created. Transformations and imputation make the data more useful in the modeling process.

When transformations are performed:

- A new variable name is automatically generated for each transformed variable. The name is derived from the original variable name, and a prefix that indicates the type of transformation is applied to it. For example, if an income variable is transformed using Recenter, the new variable will be called RRC_Income.

- Only one transformed variable per each transformation type is allowed. For example, executing multiple transforms of the same type on the same variable will not result in the creation of multiple transformed variables.

- If you execute the same transformation method on the same variable but with a new value, for example, if you are using imputations and you change the constant to a new method, the transformed variable will be replaced by the new transformed variable.

- Even if sampling is selected, transformations will be applied to the entire data set.
The transformed variable is automatically added to the Data tab. Its role is inherited from the role of the original variable. The original variable role is automatically changed to Ignore. The assumption is that the transformed variable will be used for analysis and modeling.

Types of Transformations Included With RStat

Rescale

There are two types of rescaling transformations.

Normalize. The terms rescaling, normalization, and standardization are frequently used interchangeably. They denote the conversion of one unit of measurement into another by applying a mathematical formula. For example, the conversion from Celsius to Fahrenheit involves a process of multiplying by a constant and adding a constant. There are many reasons to perform normalization of the data. One reason is to make a skewed distribution normal. For example, income is frequently skewed. Using a log transformation will normalize it. Another reason is to make two measures more comparable in magnitude. For example, age and income differ significantly in magnitude, but using scale, they can be rescaled from 0 to 1 and thus used in cluster analysis.

Recenter (RRC). Rescales the values in the data set so that the mean of the transformed variable is 0 and the standard deviation is 1. It is also referred to as standardization, that is, the process of subtracting a measure of location and dividing by a measure of scale. For example, subtracting the mean (location) and dividing by the standard deviation (scale) generates a variable with a mean of 0 and standard deviation of 1.

Scale [0-1] (R01). Rescales the variable so that the new values range between 0 and 1. It is often referred to as Normalization, using the minimum and the range of the variable in order to make all the elements lie between 0 and 1.

-Median/MAD (RMD). Rescales the values so that the median for the new data set is zero and the median absolute deviation is 1. This is a variant of the Recenter methods using the median instead of the mean and the absolute deviation instead of the standard deviation.

Natural Log (RLG). Transforms the original value into the log values. A logarithm is the power (exponent) to which a base number must be raised in order to get the original number. Thus, log10(100)=2.

Matrix (RMA). If one variable is selected, each value will be divided by the sum of the entire data set. If more than one variable is selected, transformed variables for each selected variable will be created. The values of the transformed variables will be formed by dividing each original value by the sum of all selected variables (the matrix).
The following image displays the Rescale transformations.

![Rescale transformations image]

**Impute**

Imputation is used to fill in the missing values in the data. The Zero/Missing imputation is a very simple method. Any missing numeric data is simply assigned 0, and any missing categoric data is put into a new category called Missing. Mean, Median, and Mode replace missing values with the population mean, median, or mode.

- **Zero/Missing (IZR).** The Zero/Missing imputation is a very simple method. Any missing numeric data is simply assigned 0, and any missing categoric data is put into a new category called Missing.
- **Mean (IMN).** The missing values are replaced with the mean value. The mean method cannot be applied to categoric variables.
- **Median (IMD).** The missing values are replaced with the median value. The mean method cannot be applied to categoric variables.
- **Mode (IMO).** The missing values are replaced by the mode. For categoric values, the missing values will be replaced by the most frequently occurring category, excluding Missing.
- **Constant (ICN).** The missing values will be replaced by a user-provided constant. The method can be used on both numeric and categoric variables.

The following is an image of the Impute options in the Transform tab.

---

**Recode**

Recoding is the process of reassigning values to new categories or reassigning a variable to a new type.

**Binning.** Binning is a process of grouping measured data into classes or categories. There are several types of binning transformations.

- **Quantiles (BQx).** Assigns the values to four groups of approximately equal size.

- **KMeans (BKx).** KMeans clustering will be used to assign the observations (rows of data) to clusters (groups). Only numeric variables can be clustered. If any categoric variables are selected, you will receive a message indicating that clustering is available only for numeric data.
- **Equal Width (Bex).** The range between the minimum to maximum values will be split into a user-defined number of groups. The groups will have equal width.

- **Number.** The spinner control allows users to define the number of groups.

- **Indicator Variable (TIN).** Assigns each value of a particular categoric variable to a new numeric variable. The categoric value (for example Male) is set to 1 in the new variable, while all the other categories (Female, unknown) are set to 0. The process is repeated for each value in the original categoric variable. The new indicator variables are also referred to as dummy variables, which can be used independent of one another in a regression analysis.

- **Join Categorics (TNJ).** Concatenates the values in two categoric variables. The transformation can be applied to only two categoric variables.

- **As Categoric (TFC).** Transforms a numeric variable into a categoric value, while turning numbers into strings. This allows you to treat numbers as factors in the modeling process. For example, if you have five groups, labeled 1 to 5, and if they are loaded as numeric variables, the regression model will estimate one coefficient. If you transform them to five factors, the regression will estimate one coefficient for each group.

- **As Numeric (TFN).** Converts a categoric variable into a numeric variable by assigning a number to each category. For example, you may have ordered categories, such as Very Strong, Strong, Somewhat Strong, and so on. It may be useful to convert to numeric to perform various tests on the rankings.
The following is an image of the Recode option on the Transform tab.

**Cleanup**

Cleanup allows users to delete various elements from the loaded data set. This is particularly useful in freeing up memory, especially if the modeler is creating many transformed variables for testing purposes.

- **Delete Ignored.** Deletes any variable whose Role is set to Ignore in the Data tab.
- **Delete Selected.** Deletes the selected variables in the Transform tab.
- **Delete Missing.** Deletes any variable that contains missing values.
- **Delete Obs with Missing.** Deletes any rows in the data that contain missing values in any of the variables.
Clustering

Clustering is a method of organizing objects into distinct groups or clusters based on their similarities. There are different types of clustering methods, including:

- KMeans
- Ewkm
- Hierarchical
- BiCluster

Each option uses different criteria to sort objects into groups for the purpose of analysis. For example, you can determine the degree of association between two objects. This allows you to discover unique structures in data, which would otherwise be unexplained.
The Cluster tab enables you to use various clustering techniques to group data or objects. The results of a KMeans clustering algorithm are shown in the following image.

The different types of clustering algorithms include:

**KMeans**

A partitioning method that is best suited for large amounts of data. It creates a group of mutually exclusive, unique clusters, and then returns an index of those clusters. It then presents the means, or averages, of those clusters. The value of k represents each unique cluster.

**Ewkm**

Used to cluster high-dimensional data, Ewkm outputs the weight of each variable in each cluster.
Hierarchical

A method of cluster analysis that builds a hierarchy of clusters. You can create a dendrogram, show statistics (which displays the data averages), plot the data, or create a discriminant plot. Using an agglomerative, or bottom up, approach, this method is based on matching subsequent clusters to originating clusters, which are based on a single observation.

BiCluster

Also known as co-clustering or two-mode clustering, BiCluster is a technique that is employed during data mining. It allows the rows and columns of a matrix to be clustered simultaneously. The result of the model shows the number of clusters and their respective rows and columns.
Creating a Scoring Application

This chapter describes the process of building a scoring application with RStat. It also includes information on visualizing the Decision Tree Model, including diagramming Decision Tree Model Rules and using FancyPlots.

In this chapter:

- Defining the Model Data
- Building a Model
- Evaluating the Model
- Exporting the Final Model to Build the Scoring Application
- Compiling and Deploying the Scoring Routine
- Displaying Model Information With the RStat Query Command
- Building a Scoring Application Using a Scoring Routine
- Missing Data in Scoring Routines

Defining the Model Data

Load your data, as detailed in *Introducing WebFOCUS RStat* on page 13.

Defining Model Sampling

RStat provides random sampling. You can divide your data set into a training data set and a testing data set. The training data set will be used to build the model. The testing data set, also called the evaluation data set, can be used by the model evaluation techniques to test how well the model predicts.
Define the proportion of data to be included in each data set and the seed to be used to generate the random sample.

Note:

- By default, when new data is loaded, Sampling is turned on.
- You can define the size of the training data set using either the Percentage or Count controls. Notice that they are tied together. As you change the value of the Percentage, the Count will automatically be updated with the actual sample size that will be created.
- To replicate the same sample, use a constant seed value. If you would like to vary the contents of the random sample, modify the seed value.
Defining Variable Roles

For each of the variables within your data set, you can define the role it should play in the model by clicking the appropriate column within the Variable Grid.

RStat automatically assigns roles to variables based on the following variable prefixes.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Identifier</td>
</tr>
<tr>
<td>IGNORE</td>
<td>Ignored</td>
</tr>
<tr>
<td>IMP</td>
<td>Imputed</td>
</tr>
<tr>
<td>RISK</td>
<td>Risk measure</td>
</tr>
</tbody>
</table>

You can have one Target and one Risk variable.
You can override these default settings by clicking the appropriate role for each of your variables.

**Setting Variable Roles for Groups**

You can set a group of variables to a single role using the Input and Ignore buttons by:

- Selecting the variables to be set from the Variable Grid. To select multiple variables, hold down the Ctrl key while clicking each variable, or the Shift key to define a range. You can select all variables within the grid by clicking one of the variables and then pressing Ctrl+A.

- Clicking the green Input button to define all selected variables as input or the red Ignore button to define all selected variables as ignored. This defines a portion to be used as training data and opens with the selected data set loaded. RStat presents nine tabs that reflect the standard modeling workflow. The Data tab shows the variables and the roles each will play in building the model.
Setting the Target Type

The data type of the target variable determines the type of modeling available and the specific algorithms that will be used within the modeling process. The data type is defined based on the type of data RStat identifies and the quantity of unique values found in the actual data. In RStat, data types are defined as:

- **Auto.** This option is selected by default.
- **Categoric.** Any character data or any numeric data with 10 or less unique values.
- **Numeric.** Any numeric data with more than 10 unique values.
- **Survival.** Allows the user to run a Survival Model (that is, Cox Proportional Hazards or Parametric). The Time and Status roles must be assigned to two variables in the data set in order to run the survival techniques.
Note: The target setting does not change the actual data within the data grid. It will change only the way the target data is used when the model is built.

Executing Data Settings

Once you have set or confirmed the Sampling, Data roles, and Target type, click Execute from the RStat toolbar to pass these settings to RStat.

Notice that the status bar will display the:

- Number of rows of data.
- Number of Input variables.
- Target variable and type that will be used for modeling.
Building a Model

In this section, we will review the different model types that are available in RStat and build a Decision Tree model. For more information, see Building the Decision Tree Model on page 140.

Model Tab Options

Using RStat, you can define and execute various models against a selected database.

Note:

- Depending on your data, certain model types may be disabled.
- The Ada Boost option is selected by default.
- You can click Select All to select all relevant models, or select one or multiple models to perform a simultaneous analysis. To view the output of other models that were selected for simultaneous execution, click the model type within the Model Type table.

The supported model types and their exportability are listed in the following table.

<table>
<thead>
<tr>
<th>Model Type</th>
<th>Exportability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ada Boost</td>
<td>C Exportable(.c);PMML Exportable(.xml)</td>
</tr>
<tr>
<td>ARIMA</td>
<td>None</td>
</tr>
<tr>
<td>Bayesian Network</td>
<td>None</td>
</tr>
<tr>
<td>Binomial Regression</td>
<td>C Exportable(.c);PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>C Exportable(.c);PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Gamma Regression</td>
<td>C Exportable(.c);PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Gaussian Regression</td>
<td>C Exportable(.c);PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Inverse Gaussian Regression</td>
<td>C Exportable(.c);PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Linear Regression</td>
<td>C Exportable(.c);PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>C Exportable(.c);PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Multinomial Regression</td>
<td>C Exportable(.c);PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Model Type</td>
<td>Exportability</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------------------------------------------</td>
</tr>
<tr>
<td>Negative Binomial</td>
<td>C Exportable(.c); PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Neural Net</td>
<td>C Exportable(.c); PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Poisson Regression</td>
<td>C Exportable(.c); PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Random Forest</td>
<td>C Exportable(.c); PMML Exportable(.xml)</td>
</tr>
<tr>
<td>Survival</td>
<td>C Exportable(.c); PMML Exportable(.xml)</td>
</tr>
<tr>
<td>SVM</td>
<td>PMML Exportable(.xml)</td>
</tr>
</tbody>
</table>

The following procedure provides basic guidance for executing a model type option.

**Procedure:** How to Execute a Model Type Option

1. Open RStat.
2. Click the folder adjacent to the Filename field and select a data set.
3. Click **Execute** to load the data.
4. Click the **Model** tab.
5. Select a model type.
6. Select or populate fields based on the model type that you select. For more information, refer to each of the model options discussed in this section.
7. Click **Execute** to view the model based on your selections.
For example, the following image illustrates the results of modeling using SVM with the default Kernel value selected (Radial Basis (rbfdot)). The novelty detection: one-svc option was selected.

The following sections present key functionality for each of the different model types.

**Ada Boost Model**

The Ada Boost model uses the ada, which is the underlying algorithm (model builder). Boosting builds multiple, but generally simple, models. The models might be decision trees that have just one split. These are commonly referred to as decision stumps.

**Note:**

- The Ada Boost model returns class and probability.
- Ada Boost is for non-regression binary trees.
The Boost model allows you to specify the number of trees, in addition to other criteria, as shown in the following image.

The following table lists and describes the fields that are used to adjust the Boost model.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Trees</td>
<td>The number of trees to build.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> In order to ensure that every input row is predicted at least a few times, this value should not be set to a number that is too low. The default value is 50.</td>
</tr>
<tr>
<td>Max Depth</td>
<td>Allows you to set the maximum depth of any node of the final tree. The root node is counted as depth 0. The default value is 30.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Values greater than 30 will generate invalid results on 32-bit machines.</td>
</tr>
<tr>
<td>Stumps</td>
<td>If the Stumps check box is selected, you can build stumps using the Boost model. If the Stumps check box is not selected, the results in the default values are deactivated.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Min Split</td>
<td>The minimum number of entities that must exist in a data set at any node for a split of that node to be attempted. The default value is 20.</td>
</tr>
<tr>
<td>Complexity</td>
<td>Also known as the complexity parameter (cp), this value allows you to control the size of the decision tree and select the optimal size tree. If the cost of adding another variable to the decision tree from the current node is above the value of the cp, then tree building does not continue. The default value is 0.0100. <strong>Note:</strong> The main role of this parameter is to save computing time by pruning unnecessary splits.</td>
</tr>
<tr>
<td>X Val</td>
<td>Refers to the number of cross-validation errors allowed. The default value is 10.</td>
</tr>
</tbody>
</table>
Once you have defined your model criteria, you must click the Execute button to review the results, as shown in the following image.

ARIMA Model

The Autoregressive Integrated Moving Average (ARIMA) model is used for short term forecasting. Also known as Box-Jenkins, ARIMA is used with data sets that show a stable, consistent pattern over time. The general idea of the model is to be able to forecast future values based on the patterns of the data points used in the current data set.

ARIMA requires at least 40 historical data points to forecast the future values in a series. The model works best when your data reveals a consistent pattern with few outliers.
Once you select ARIMA as the model, you must click Execute to load the ARIMA Forecasting GUI, as shown in the following image.

The following table lists and describes the fields that are used when working with an ARIMA model.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DATA</strong></td>
<td><strong>Summary and Plot</strong>. Provides a summary of the data that has been loaded. It displays the first few records of the data set, along with a summary of the variable to be forecasted: minimum, first quartile, median, mean, third quartile, and maximum. The plot is a time series.</td>
</tr>
<tr>
<td>Check Data:</td>
<td>Summary and Plot. Provides a summary of the data that has been loaded. It displays the first few records of the data set, along with a summary of the variable to be forecasted: minimum, first quartile, median, mean, third quartile, and maximum. The plot is a time series.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>TRANSFORMATION</strong></td>
<td></td>
</tr>
<tr>
<td>Trans. Method</td>
<td>Select the transformation method used when you click the Transformation button. You can select from the following options:</td>
</tr>
<tr>
<td></td>
<td>□ Identity</td>
</tr>
<tr>
<td></td>
<td>□ Log</td>
</tr>
<tr>
<td></td>
<td>□ SQRT</td>
</tr>
<tr>
<td></td>
<td>□ BoxCox</td>
</tr>
<tr>
<td>Check Trans.</td>
<td>Creates a new vector or data column using the selected transformation on the historical data.</td>
</tr>
<tr>
<td><strong>BUILD ARIMA</strong></td>
<td></td>
</tr>
<tr>
<td>Auto Arima:</td>
<td>Runs the Auto ARIMA function and returns the best ARIMA model according to either AIC, AICc, or BIC.</td>
</tr>
<tr>
<td>Check ACF, PACF</td>
<td>ACF and PACF compute and plot the autocorrelation and partial autocorrelation functions, respectively. You can select from the following options:</td>
</tr>
<tr>
<td></td>
<td>□ ACF (Autocorrelation function). Computes and plots an estimate of the autocorrelation function.</td>
</tr>
<tr>
<td></td>
<td>□ PACF (Partial Autocorrelation function). Computes and plots an estimate of the partial autocorrelation function of a possibly multivariate time series.</td>
</tr>
<tr>
<td></td>
<td>□ ACF and PACF</td>
</tr>
<tr>
<td></td>
<td>□ TS, ACT, and PACF</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> TS is a time series plot of historical information.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Construct Arima:    | Allows you to enter user-defined values for the arguments \( p, d, \) and \( q \). When you execute the model the first time, it is saved in memory as ARIMA_1. All consecutive executions are named ARIMA_\( n \), where \( n \) is the next consecutive number assigned. You can populate values for the following fields:  
  - \( p \). The number of autoregressive terms.  
  - \( d \). The number of non-seasonal differences needed for stationarity.  
  - \( q \). The number of lagged forecast errors in the prediction equation, also known as the number of moving average terms. |

**MODEL DIAGNOSE**

<table>
<thead>
<tr>
<th>Select Model</th>
<th>Lists the models that you have generated while using the Construct Arima section. Click Show Summary to display the ARIMA model summary.</th>
</tr>
</thead>
</table>
| Check Residuals     | Produces a time plot of the residuals, the corresponding ACF, and a histogram. You can select from the following options:  
  - **Residual ACF Plot.** Shows the residuals from the autocorrelation function.  
  - **Residual All.** Plots the Standardized Residuals, ACF of Residuals, and \( p \) values for Ljung-Box statistic. |
### Field Name

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| Box. Test  | Compute the Box-Pierce or Ljung-Box test for examining the null hypothesis of independence in a given time series. Options include:  
  - **lag**. The statistic will be based on lag autocorrelation coefficients.  
  - **type**. Select a test from the drop-down list. Options include: Ljung-Box and Box-Pierce.  
  - **fitdf**. This is the number of degrees of freedom to be subtracted. |

### EXPORT

**What to Export:**

- Provides options for exporting modeling information. Options include:  
  - **Export Summary**. Exports the contents of the Summary window to a user named text file.  
  - **Export Log**. Exports the contents of the Log window to a user named text file.  
  - **Export Plot**. Exports the plot in .jpg, .pdf, .png, or .wmf format.  
    - **Note**: The .svg format is not supported.  
  - **Export C Routine**. Currently, this function is unavailable.

### Bayesian Network Model

Bayesian networks, also known as belief networks (or Bayes nets), are a type of probabilistic graphical model. They can be used to make predictions and automated insights, and perform diagnostics. It represents a set of random variables and their conditional dependencies, thus it allows you to represent and reason about uncertainties.
Bayesian networks are directed acyclic graphs (DAGs), which use random variables to represent observable quantities or hypotheses. Each graph has nodes, which represent these random variables, one node per variable. The edges between the nodes represent conditional dependencies amongst the variables.

**Note:** When working with the Bayesian model and the data is loaded, no variable should be selected as target; all variables should be selected as input.

The Bayesian network model is effective in illustrating probability through the use of graphs and the variables that comprise them. The Bayesian Network options are shown in the following image.

The following table lists and describes the tabs and fields that are used when working with the Bayesian model.
Preparing Data

The following fields display:

- Select Field(s):
- Set Intervals:
- Set Labels:
- Discretize Data

When loading variables with a Categoric data type, the Preparing Data tab will be disabled. When loading variables with a Numeric data type, the Preparing Data tab will be enabled and the Discretize Data button will be available. When you click this button, the selected variables are divided into intervals, converting continuous data into discrete data.

The Preparing Data tab will initially display a summary of the loaded variables. This summary assists in the setting of intervals for discretizing the data.

On this tab:

1. Select the Field from the drop-down list.
2. Set the intervals using the R syntax. For example: c(30,50,70,90).

If Set Labels is left blank, then the actual interval will be displayed. For example, (70,90).

If you want to use labels, use the R syntax. For example, c(1,2,3) or c("A","B","C"). The number of labels must be one less than the number of values used when setting the intervals. For example, in the intervals c(30,50,70,90) are four values. Exactly three labels must be entered.
<table>
<thead>
<tr>
<th>Tab Name</th>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| Structure     | Learning Algorithms                            | In simple cases, an expert can specify a Bayesian network used to perform inference. In more complex cases, the network structure and the parameters of the local distributions must be learned from the data. You can use the following steps to specify a learning algorithm:  
1. From the Learning Algorithm drop-down menu, select the desired algorithm.  
2. On the ribbon, click **Execute**.  
The learning is displayed in the text view and the network is displayed in a separate R graphics window.  
**Note:** A new R graphics window is opened every time you click **Execute**. |
<p>| Learning      | The following field displays:                  |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               | - Learning Algorithms                          |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |   - Options include:                           |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |     - Grow-Shrink (gs)                         |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |     - Incremental Association (iamb)           |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |     - Fast Incremental Association (fast.iamb) |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |     - Interleaved Incremental Association      |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |       (inter.iamb)                             |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |     - Hill-Climbing (hc)                       |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |     - Tabu Search (tabu)                       |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |     - Max-Min Hill-Climbing (mmhc)             |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |     - Restricted Maximization (rmax2)          |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |     - Max-Min Parents and Children (mmpc)      |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |     - Hilton Parents and Children (si.hilton.pc)|                                                                                                                                                                                                                                                                                                                                                                                                   |
|               |     - Chow-Liu (chow.liu)                      |                                                                                                                                                                                                                                                                                                                                                                                                  |
|               |     - ARACNE (aracne)                         |                                                                                                                                                                                                                                                                                                                                                                                                  |</p>
<table>
<thead>
<tr>
<th>Tab Name</th>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter Estimation</td>
<td>The following fields display:</td>
<td>In a distribution model, the set of parameters needs to be estimated. These parameters specify any constants in the model, enabling efficient and accurate use of the data. For the parameter estimation to work correctly, the network must be a directed acyclic graph (DAG), as plotted on the Structure Learning tab. If the graph is not a DAG, then the following error message will display: the graph is only partially directed.</td>
</tr>
<tr>
<td></td>
<td>Method</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Options include:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Maximum Likelihood parameter estimation (mle)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bayesian parameter estimation (bayes; discrete data)</td>
<td></td>
</tr>
<tr>
<td>Inference</td>
<td>The following fields display:</td>
<td>The Inference tab is used to perform conditional probability queries (CPQs). If the query type is Probability, then enter the event of interest as fieldname==&quot;factor level&quot;. Evidence is enter as fieldname==&quot;factor level&quot;. If the query type is Observations, then enter the event of interest as &quot;fieldname&quot;. Evidence is entered fieldname==&quot;factor level&quot;.</td>
</tr>
<tr>
<td></td>
<td>Query Types:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Options include:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Probability</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Observations</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Plot</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Event:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Evidence:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>n:</td>
<td></td>
</tr>
</tbody>
</table>
Plotting Conditional Probability

Before plotting the Conditional Probability, you must take the following steps:

1. Click the Structure Learning tab and select and execute a learning algorithm, for example Hill-Climbing.

2. Click the Parameter Estimation tab and select and execute a Method. For example, Maximum Likelihood parameter estimation (mle).

3. Then, on the Inference tab, click the Observations radio button. For Event, enter any one of the variable names in quotes. For example, "STAT". For Evidence, enter a logical statement, for example, VECT >= 50. To run an unconditional probability distribution, set either event or evidence to TRUE.

4. Click Plot.

5. A histogram displays, showing the frequency distribution.

**Note:** Before the plot is generated, the parameters of the Bayesian network are fitted conditional on its structure and random observations conditional on the evidence using the logic sampling method are generated.
A sample plot is shown in the following image.
Bayes Factor Analysis

After loading the data and selecting a target, you can perform a Bayes factor analysis. From the Model tab, select Linear Regression and click Execute. Click Bayesian Linear Analysis, which displays the Bayes Factor Analysis of Regression user interface and simultaneously computes Bayes factors for groups in regression designs, as shown in the following image.

Note: There are three icons that consistently display when using the Bayes Factor Analysis GUI: the brush, which will clear the output areas Textview and Comparison Plot, the disc icon, which will save the current plot in the following formats: .pdf, .png, .jpg, and .wmf, and the pipe connector icon, which exports the content of the Textview pane.
<table>
<thead>
<tr>
<th>Tab Name</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Linear Regression</td>
<td><strong>Specified Model.</strong> The specified model contains the formula for computing Bayes factors for a linear regression model.</td>
</tr>
<tr>
<td></td>
<td><strong>Show Selected Model.</strong> Enter the formula into the Edit text box and then click Specified Model. This adds the model to the Show Selected Model drop-down list box, making it available for selection. Select a model from the drop-down list and click Show Selected Model, which will display the Bayes factor analysis for the selected model in the Textview pane. Example formulas include: len<del>supp and len</del>dose+supp.</td>
</tr>
<tr>
<td>Regression(Numeric)</td>
<td>The following buttons display in the Bayes Factor Analysis GUI:</td>
</tr>
<tr>
<td></td>
<td><strong>Best Models.</strong> Click this button to show the best six models in the Textview pane. This also shows the Bayes Factor Analysis along with the name of the model.</td>
</tr>
<tr>
<td></td>
<td><strong>Worst Models.</strong> Click this button to show the worst six models in the Textview pane. This also shows the Bayes Factor Analysis along with the name of the model.</td>
</tr>
<tr>
<td></td>
<td><strong>All Models.</strong> Click this button to show all models without the Bayes factor analysis.</td>
</tr>
<tr>
<td></td>
<td><strong>Specified Model.</strong> Retype or copy and paste any one of the model names from the All Models in Textview. Ensure that when you enter this into the text box, the model name is enclosed in quotes. For example, &quot;complaints + privileges&quot;, and then click the Specified Model button to display the Bayes Factor Analysis.</td>
</tr>
<tr>
<td></td>
<td><strong>Comparison Plot.</strong> There are three types of comparison plots: Best Model (all), Most Complex Model (top), and Intercept Only (bottom). Clicking this will update both the Textview pane and the Comparison Plot pane.</td>
</tr>
</tbody>
</table>
Decision Tree Model

The Decision Tree option is used to generate a decision tree, which is the prototypical data mining technique. It is widely used because of its ease of interpretation. The Decision Tree model uses an underlying algorithm (model builder) of rpart, as shown in the following image.

The following table lists and describes the fields that are used to adjust a Decision Tree model.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min Split</td>
<td>The minimum number of entities that must exist in a data set at any node for a split of that node to be attempted. The default value is 20.</td>
</tr>
<tr>
<td>Min Bucket</td>
<td>The minimum number of entities allowed in any leaf node of the decision tree. The default value is one third of the min split.</td>
</tr>
<tr>
<td>Max Depth</td>
<td>Allows you to set the maximum depth of any node of the final tree. The root node is counted as depth 0. The default value is 30.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Values greater than 30 will generate invalid results on 32-bit machines.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Complexity</td>
<td>Also known as the complexity parameter (cp), this value allows you to control the size of the decision tree and select the optimal size tree. If the cost of adding another variable to the decision tree from the current node is above the value of the cp, then tree building does not continue. The default value is 0.0100. <strong>Note:</strong> The main role of this parameter is to save computing time by pruning unnecessary splits.</td>
</tr>
<tr>
<td>Priors</td>
<td>Allows you to set the prior probabilities for each class.</td>
</tr>
<tr>
<td>Loss Matrix</td>
<td>Allows you to weight the outcome classes differently.</td>
</tr>
</tbody>
</table>
Once you have specified your model criteria, you must click the Execute button to view the result. Since Tree was selected, the Summary of the Decision Tree model displays, as shown in the following image.

Note: The default values for all fields were used to create this example.

You can view the summary or access other features of the application, including Rules, Draw, and FancyPlots. For more information, see:

- How to Display the Decision Tree Model Rules on page 142
- How to Diagram the Decision Tree Model Rules on page 144
- Using FancyPlots on page 145
Regression Model

Regression is a traditional approach to modeling. The model builder glm (logit) is used by the Regression model. Logistic regression (using the binomial family) is used to model binary outcomes. Linear regression is used to model a linear numeric outcome. For predicting where the outcome is a count, the Poisson family is used. Generalized Regression is generalization of standard linear regression, allowing for response variables that fall outside of a normal distribution. Multinomial regression generalizes logistic regression, in that it allows more than two discrete outcomes. For more information, see Building a Logistic Model on page 239 and Building a Linear Regression Model on page 225.

RStat supports regression and advanced regression, as shown in the following image.

The following techniques related to regression can be performed:

- **Binomial.** Distribution is binomial distribution, which is a discrete probability distribution of the number of successes in a sequence of a variable number (n) of dependent Yes or No experiments. The LinkFunction could be either logit, probit, or cauchit.
Gamma. The error distribution is Gamma distribution. The LinkFunction could be inverse, identity, or log.

Gaussian. The error distribution is normally distributed. The response method is dependent on the LinkFunction. The LinkFunction can be identity, log, or inverse.

Inverse Gaussian. The error distribution is Gamma distribution. The LinkFunction could be inverse, identity, log, or $1/mu^2$.

Linear. The error distribution is normally distributed. The response variable is linear in relation to the predictive model.

Logistic. Usually referred to as binomial or binary logistic regression, this is used to predict two possible types of outcome (YES or NO).

Multinomial. Referred as multinomial logistic regression, this is used to predict three or more possible types of outcome.

Negative Binomial. The error distribution is negative binomial distribution, which is a discrete distribution of the number of successes in a sequence of Bernoulli trials before a specified number of failures occur.

Poisson. The error distribution or response variable distribution is Poisson distribution. Poisson regression is used to model and predict in cases where count data and contingency tables are used.

Neural Net Model

Neural Network (Neural Net) is an older approach to modeling. The Neural Net model uses a structure that resembles the neural network of a human being. When applied to modeling, the concept is to build a network of neurons that are connected by synapses. Rather than generate electrical signals, however, the network propagates numbers.

When using the Neural Net model, you can include or exclude the interval of networks from your model. The interval of networks is used to define and describe the relationships within your model. Display of these intervals is accomplished using the Skip field. When set to True (default), the interval of networks displays. If the Skip field is set to False, the interval of networks does not display.
The default value of True is shown in the following image.

![Image showing default value of True](image.jpg)

The following table lists the options that are available on the Neural Net model screen.

<table>
<thead>
<tr>
<th><strong>Field Name</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Hidden Layer Nodes</td>
<td>The number of hidden layer nodes to display. The default is 10.</td>
</tr>
<tr>
<td>Skip</td>
<td>Set to a value of TRUE by default, this field switches between the input and output of skip-layer connections, depending on your selection.</td>
</tr>
</tbody>
</table>

In the following diagram, the relationships between the different layers of a neural record are illustrated. The bottom portion represents the input layer, the middle nodes form the hidden layer, and the top components constitute the output layer.
**Note:** The number of nodes varies in different applications. In addition, you must have a data set selected and loaded in order to use this functionality.
The following example shows the results of a Neural Net model with the Skip field set to True. Accordingly, RStat displays the interval of networks.

![Neural Net model results]

**Random Forest Model**

The Random Forest uses an underlying algorithm (randomForest), which builds multiple decision trees from different samples of the data set. While building each tree, random subsets of the available variables are considered for splitting the data at each node of the tree.

**Note:**

- When using the C export with the Random Forest model, class and regression(numeric values) are returned.
- Random Forest trees can be binary and k-ary(k>2).
You can specify a number of trees (the default is 500) and a number of variables (the default is 4), as shown in the following image.

The following table lists and describes the fields that are used to adjust the Random Forest model.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Trees</td>
<td>The number of trees to build.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> In order to ensure that every input row gets predicted at least a few times, this value should not be set to a number that is too low. The default value is 500.</td>
</tr>
<tr>
<td>Number of Variables</td>
<td>This is the number of variables to be considered at any time in deciding how to partition the data set. Each split produces a number of variables which are randomly sampled as candidates.</td>
</tr>
<tr>
<td>Sample Size</td>
<td>Size(s) of a sample to draw.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Impute</td>
<td>This applies to missing variables. If this check box is selected, the variables are transferred by replacing any NA values with one of the following (depending on the current selection):</td>
</tr>
<tr>
<td></td>
<td>□ Missing</td>
</tr>
<tr>
<td></td>
<td>□ 0</td>
</tr>
<tr>
<td></td>
<td>□ Mean</td>
</tr>
<tr>
<td></td>
<td>□ Median</td>
</tr>
<tr>
<td></td>
<td>□ Mode</td>
</tr>
<tr>
<td></td>
<td>□ Constant</td>
</tr>
</tbody>
</table>
Once you have indicated your criteria, you must click the Execute button to see the results, as shown in the following image.
Survival Model

The Survival Model is used to model time-to-event data. When using this option, you can select Cox Proportional Hazards (coxph) or Parametric (survreg) to perform your Survival Analysis, as shown in the following image. For more information on the Survival model, see Building a Survival Model on page 251.

The following table lists and describes the fields that are used to adjust a Survival model.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>The variable that you selected (time) on the Data tab.</td>
</tr>
<tr>
<td>Status</td>
<td>The variable that you selected (status) on the Data tab.</td>
</tr>
<tr>
<td>Model Builder</td>
<td>The name of the model builder (coxph or survreg) related to the selection you made: Cox Proportional Hazards or Parametric.</td>
</tr>
<tr>
<td>Cox Proportional Hazards</td>
<td>A general regression model that predicts individual risk relative to the population.</td>
</tr>
<tr>
<td>Parametric</td>
<td>Also known as the accelerated failure time model, this regression model predicts the expected time to the event of interest.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Survival</td>
<td>This option enables you to view the results of the Cox Proportional Hazards model. For more information, see <em>Building a Survival Model</em> on page 251.</td>
</tr>
<tr>
<td>Residuals</td>
<td>This button enables the testing of the assumption of proportional hazards.</td>
</tr>
</tbody>
</table>

**SVM Model**

The Support Vector Model (SVM) is a modern approach to modeling where the data is mapped to a higher dimensional space. This increases the possibility that vectors separating the classes will be found.

You can select the SVM option to specify a kernel and related options in support of the model.

**Note:**

- The kernel functions are from the SVM defines. Type ksvm can be used for classification, regression, or novelty detection.
- The list of options is the same for all kernels.
The Options drop-down list defaults to C classification: C-svc, as shown in the following image. If you do not make a selection from the Options drop-down list, the default value is used.

The following table lists the options that are available on the SVM model screen.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kernel</td>
<td>The kernel function is used in training and predicting. You can select one of the following kernels:</td>
</tr>
<tr>
<td></td>
<td>- Radial Basis (rbfdot) (Default)</td>
</tr>
<tr>
<td></td>
<td>- Polynomial (polydot)</td>
</tr>
<tr>
<td></td>
<td>- Linear (vanilladot)</td>
</tr>
<tr>
<td></td>
<td>- Hyperbolic Tangent (tanhdot)</td>
</tr>
<tr>
<td></td>
<td>- Laplacian (laplacedot)</td>
</tr>
<tr>
<td></td>
<td>- Bessel (besseldot)</td>
</tr>
<tr>
<td></td>
<td>- ANOVA RBF (anovadot)</td>
</tr>
<tr>
<td></td>
<td>- Spline (splinedot)</td>
</tr>
<tr>
<td>Options</td>
<td>The ksvm model builder can be used for classification, regression, or novelty detection. When using the ksvm model builder, you can select one of the following options from the drop-down list:</td>
</tr>
<tr>
<td></td>
<td>- C classification: C-svc (Default)</td>
</tr>
<tr>
<td></td>
<td>- nu classification: nu-svc</td>
</tr>
<tr>
<td></td>
<td>- bound-constraint svm classification: C-bsvc</td>
</tr>
<tr>
<td></td>
<td>- Crammer, Singer native multi-class: spoc-svc</td>
</tr>
<tr>
<td></td>
<td>- Weston, Watkins native multi-class: kbb-svc</td>
</tr>
<tr>
<td></td>
<td>- novelty detection: one-svc</td>
</tr>
<tr>
<td></td>
<td>- epsilon regression: eps-svr</td>
</tr>
<tr>
<td></td>
<td>- nu regression: nu-svr</td>
</tr>
<tr>
<td></td>
<td>- bound-constraint svm regression: eps-bsvr</td>
</tr>
</tbody>
</table>
Building the Decision Tree Model

This section reviews the basic procedure for building a Decision Tree model.

1. Select the Model tab.

2. From the Model Type drop-down list box, select Decision Tree, as shown in the following image.

3. Click Execute to create the model.
The model metadata or output appears in the model textview area, as shown in the following image.

**Visualizing the Decision Tree Model**

The Decision Tree generates rules that predict the score and divides the sample data into multiple segments (branches). Each branch terminates in a node that associates a subset of the customers with a predicted score. The rules describe the criteria that qualify for each node. The predicted score is a probability value between 0 and 1. Those with a probability of .5 or greater are predicted as a good risk, and those with less than .5 are predicted as a bad risk.

You can display the rules or diagram the nodes.
Procedure: How to Display the Decision Tree Model Rules

1. On the Model tab, click Rules to display the rules associated with the selected database, as shown in the following image.
The Rpart Rules GUI displays, as shown in the following image.

2. Scroll down to review the rules.
3. Optionally, click `Save current output to file` to save the rules information as a .txt or .xml file.

The Export Rpart Rules dialog displays, as shown in the following image. This enables you to specify a file name and a folder location.
**Note:** You can save the file in the current (default) folder or specify a new folder in which to place the file by clicking *Browse for other folders*.

4. Click Save to save the file.

**Procedure:** **How to Diagram the Decision Tree Model Rules**

On the Model tab, click the Draw button, as shown in the following image.
The diagram displays in the RGui.

The colored numbers at the end of each node correspond to the rules, as shown in the following image.

---

**Using FancyPlots**

Accessed from the Model tab in the RStat application, the FancyPlot functionality enables you to plot data from the database with which you are working. You can also split the plot or collapse nodes in the tree. FancyPlots provide unique charting options that can be customized for each data set.

**Procedure:** How to Access FancyPlots

1. Launch RStat.
2. On the Data tab, select a file name to load and then click **Execute**.
3. On the Model tab, from the Model Type list, click **Decision Tree** and then click **Execute**.
4. Click *FancyPlot*, as shown in the following image.
The FancyPlot functionality displays, as shown in the following image.

You can use varying combinations in the primary sections: SplitType and BranchType. Upon selecting different combinations of type and branch, click *Execute* in the toolbar to update the tree image in the drawing area.

The following table defines these options.
<table>
<thead>
<tr>
<th>Group</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SplitType</td>
<td>Default</td>
<td>The default. Draw a split label at each split and a node label at each leaf.</td>
</tr>
<tr>
<td></td>
<td>List All Nodes</td>
<td>Label all nodes, not just the leaves. Similar to text.rpart all=TRUE.</td>
</tr>
<tr>
<td></td>
<td>Labels Under Nodes</td>
<td>Similar to List All Nodes, but draws the split labels below the node labels.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Similar to the plots in the CART book.</td>
</tr>
<tr>
<td></td>
<td>Label Both Directions</td>
<td>Draw separate split labels for the left and right directions.</td>
</tr>
<tr>
<td></td>
<td>List All Nodes and Both Directions</td>
<td>Similar to Label Both Directions, but labels all nodes, not just leaves.</td>
</tr>
<tr>
<td>BranchType</td>
<td>Default</td>
<td>The default. The branch lines are drawn conventionally.</td>
</tr>
<tr>
<td></td>
<td>Deviance</td>
<td>Deviance</td>
</tr>
<tr>
<td></td>
<td>SQRT(Deviance)</td>
<td>Square-root (deviance)</td>
</tr>
<tr>
<td></td>
<td>Deviance/Obs</td>
<td>Deviance / nobs</td>
</tr>
<tr>
<td></td>
<td>SQRT(Deviance/Obs)</td>
<td>The standard deviation when method=anova</td>
</tr>
<tr>
<td></td>
<td>weight</td>
<td>Also known as frame$wt, this is the number of observations at the node,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>unless rpart weight argument was used.</td>
</tr>
<tr>
<td></td>
<td>Complexity</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Abs</td>
<td>This is the predicted value.</td>
</tr>
<tr>
<td></td>
<td>Pred.Val.-min(Pred.Val.)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Constant</td>
<td>For checking visual perception of the relative width of branches.</td>
</tr>
</tbody>
</table>
The GUI also has a toolbar with options that enable some of the basic functionality of charting. From left to right, these options are: Save current output to file, Execute current selections, Clear output area, Collapse nodes and re-plot model, and Export the current tree model.

**Charting With FancyPlots**

You can use any combination of Split Type and Branch Type to build your chart. You can clear the output area before creating a new chart or work with the initial chart that displays by default. You can perform the following tasks when charting with FancyPlots:

- Click **Save current output to file** to save the current image.
- Click **Execute current selections** to update the image with the current setting or show the selected snipped model.
- Click **Collapse nodes and re-plot mode** to enable the Snipping function.
- Click **Clear output area** to clear the drawing area.
- Click **Export the current tree model** to export the selected (current) tree model as a scoring routine.

Some additional concepts for charting with FancyPlots include:

- You can collapse nodes in the tree using the scissors icon, which is available in the toolbar.
- When the plot is active, you will see a Quit option in the upper-left corner of the chart and all nodes will be connected with black lines.
- When you click on a node, the subtree will be colored.
- As you snip one or more charts, the various versions of the chart will display in the Tree Models drop-down list.
- The Rules button displays the RpartRulesGUI.
- At any time, you can save a chart in one of the following formats: .pdf, .png, .jpg, .svg, or .wmf.

**Decision-Tree Snipping**

When working with a decision tree in a FancyPlot, you can edit it using the scissors icon. This enables you to determine which branches of the tree will be included in your decision tree.
When snipping the decision tree, you must snip at the intersection of a branch, as shown in the red circle in the following image.

Once you have made a snip, you can click on the Quit icon to redraw the FancyPlot without the section that you just snipped. The Quit icon displays in the upper-left corner of the screen.
Within the decision tree, the lines of the branches are black until a snip is made. Snipping changes the color to grey, indicating that this portion of the decision tree will be removed based on your snip. This is shown in the following image.

Note: If you snip at the top of the branch (where subordinate or lower branches are included), fewer nodes display, reducing the size of the chart (when the intersection and lower branches are removed).
Once you click Quit, the updated FancyPlot displays (showing the area that was snipped).

Note: In the diagram above, under the Education = Mst section, notice the node with YES = 100% of 8 before snipping and after snipping. The value of the No node is 58% of 184 after snipping, as well as before, however, this is now the end node.
As you are editing the decision tree, different versions of the tree are saved for each change you make, as shown in the following image.

This process creates a history for the chart with which you are working. You can create historical snapshots of your files and routines (PDFs or C Routines), as described in the following sections.

**Saving a Decision Tree as a PDF File**

Each original and snipped tree model can be saved as a unique .pdf.

**Procedure: How to Save a Decision Tree as a PDF File**

1. On the Model tab, select a model type from the Model Type list.
2. Click **Execute** to review the model.
3. Click **Save** to save the current image to disk, as shown in the following image.

   ![PDF Save Dialog](image)

   **Note:** You can specify a different name for the report in the Name field.

**Saving a Decision Tree as a C Routine**

Each original and snipped tree model can be saved as a c routine.
Procedure: How to Export a Decision Tree as a C Routine

1. On the Model tab, select a model type from the Model Type list.
2. Click Execute to review the model.
3. Click Export.

The Export C or PMML dialog box opens.
4. Save the file as a c routine, as shown in the following image.

Note: You can specify a different name for the report in the Name field.

Creating FancyPlots

This section presents a series of screenshots that further illustrate the FancyPlot charting functionality. You can perform the following tasks with the FancyPlot GUI:

- Create a FancyPlot of a Decision Tree
- Snipping and resnipping of FancyPlots
The ability to save the images you create

Functionality in support of exporting to a C routine.

Reference: Create Different Types of Fancy Plots

The first screen shows a basic plotted chart, through which the current tree model can be exported. Using the SplitType and BranchType categories on the left, you can vary the output of your plot.

Note: Some combinations are not allowed. In those cases, an error message will display.
When working with a plotted chart, you can clear the chart from the user interface, leaving a blank canvas with which to work.
Once you have an output, you can save it to a file.

Next, you can then open the plot and use the edit (scissor image) to collapse the nodes in the tree.
**Note:** You will see a Quit icon in the left corner of the diagram. In addition, all nodes are connected by a solid black line.
When you click one of the nodes, the sub-tree will be colored.
Click *Quit* and another tree displays (without the selected sub–trees).
After each snipping procedure, the new tree model name is added to the drop-down list.
Select any portion of the new model (other than the original tree model) and perform a snip.

Select any model in the drop-down list and click *Execute*. The selected model will be drawn in the drawing area.
Note: It is recommended that you try at least three different models before making a determination.

Save the snipped tree and check the saved image.

Evaluating the Model

The information in this section can be used to evaluate the model.

Exporting Rpart Rules

The Rpart Rule Export functionality allows you to create and store unique versions of scoring data for a selected database on a local drive or directory. The basic functionality involves choosing a database, creating a model, and then scoring the data based on the values generated and stored for that modeling scenario.
Note:

- In each step of this process, click the Execute button in order to apply your changes. When you select the Tree model type, you must then execute it so that the model loads with the Rpart model builder. When you score the data, the model will be ready for the selected database and will produce the scoring results based on these prior functions.

- The Tree model type must be selected during modeling. The Tree model type uses rpart as its model builder, as shown in the following image.

From the Model tab, you can model the scenario by clicking Execute on the toolbar after you make different selections. You can then export the rules. For more information, see How to Execute the Rpart Rule Export on page 165.

From the Evaluate tab, if the Score option is selected, RStat appends the rules after the data set within the resulting .csv file. The data specific to this functionality is identified with a column heading of Rpart, allowing you to locate the newly generated Rpart data easily. This is illustrated in the following image.
RStat has a unique method for naming the files associated with the Rpart Rule Export functionality. The file name for the output of the scoring routine is derived from the original database file name. RStat appends _train_score_all to the file name. For example, if the originating database is database.csv, the resulting filename is:

database_train_score_all.csv

Alternatively, you can provide your own file name or append information to the default file name assigned. For example, you may run the same scenario on different dates. You might use a date convention (for example, _mmddyyyy) to archive your files. In general, this file naming convention ensures that the original name of the database is preserved while marking the new output file as one that contains scoring data.

You can store these files locally using the default generated naming conventions (or that which you specify) for future analysis. For example, you can use WebFOCUS to create reports, graphs or other BI-related tasks. For more information, see the Creating Reports With WebFOCUS Language and Creating Reports With Graphical Tools manuals.

**Note:** After scoring and saving the resulting data, RStat displays the path and file name of the file that you saved when the application returns to the Evaluate tab, as shown in the following image.

![Image](image.png)

**Procedure:** How to Execute the Rpart Rule Export

1. Open RStat.
2. Click the folder adjacent to the Filename field and select a database file.
Note: The Rpart Rule Export does not support a sampling or testing data set. In order to execute Rpart rules, you must clear the Partition check box on the Data tab. This converts the data into training data from which rules can be extracted.

3. Click Open to confirm the database selection and return to the RStat interface.

4. Click Execute.

5. On the Model tab, click Decision Tree from the Model Type list.

6. Click Execute.

7. On the Evaluate tab, click Score.

   Note: Clear the check boxes for the Neural Net and SVM models, if necessary.
   
   a. For Data, select Training.
   
   b. For Include, select All.
   
   c. Click Execute to score the Rpart data and save it to a local file, as shown in the following image.
**Note:**
- On the Evaluate tab, the Execute and Export buttons produce the same results.
- You can save the output file to the default directory or optionally, create a new folder for archiving purposes. You can also rename the file as required.

**Evaluating the Decision Tree**

Evaluation techniques in RStat allow you to investigate how well your model will make predictions. The available evaluation techniques are determined by the type of model you have generated.

1. Select the *Evaluate* tab.

![Image of RStat interface with Evaluate tab selected]

Notice that the Model Type is defined as Tree and a variety of evaluation techniques are presented.
2. Select the evaluation data.

You can use the following data sets to evaluate the current model.

- From the current data:
  - **Training.** The initial data used within the model. If you are using sampling, this will be a randomly selected % of your data set based on the definition identified in the Data tab.
  - **Testing.** Only available when sampling has been used to build the model. This will contain the remaining data not used within the training data set. If the sample has been set to 70%, the training data set will contain the remaining 30%.

- From new data sources:
  - CSV
  - R Dataset
Error Matrix

An error matrix shows the relationship between the actual data and the predicted values.

With Error Matrix selected, click **Execute**.

Two error matrices are displayed. The first matrix shows the count of cases and the second shows the percentage of cases.

Looking at the second matrix, you can see that the model predicts the following:

- In 84% of the cases (Cell (0,0)), the actual value of bad credit was matched by the predicted value.
- In 11% of the cases (Cell (1,1)), people with good credit were correctly classified.
- The remaining 4% were misclassified.
- Summing across the correctly classified cases, 84% + 11% = 95% were correctly classified cases.
Scoring New Data

In RStat, you can score new data to see how well your model predicts. The Score data option will create a new CSV file with the scored values.

1. On the Evaluate tab, click Score.

New Score options appear at the bottom of the tab panel.

Report Options

Report options are available only for Binary Trees and Logistic Regressions, where your target is binary (two unique values). For other models, the Report options will be grayed out.
The Report options define the type of score to be returned.

- **Class.** A categorical value that is derived on a zero to 1 scale, where 0 through .5 = 0 and .5 through 1 = 1.

- **Probability.** A numeric value between 0 and 1 representing the likelihood that the result will be a higher value. For character-based targets, the higher value is determined alphabetically. For example, if your target is Gender with Male and Female as the values, the probability will return the likelihood that the outcome will be Male.

**Include Options**

Include options allow you to define which fields should be included in the scored file.

- **Identifiers.** Includes the identifier, the target, and the score value.
- **All.** Includes all variables in the data set plus the score value.

2. Once you have defined the Scoring options, click Execute in the RStat toolbar.

The Score Files dialog box opens.
3. Define the file name and location where the scored data will be saved.

   **Note:** The file name that you define will be the exact name used, so be sure that the file name contains a .csv extension.

In the example below, the Scored option has ALL selected instead of IDENTIFIERS. The output file structure will have all fields (variables) plus the Scored value (Column name=rpart) and the Rules column (column name=RpartRules).

   **Note:** The contents for each data line are the rule details. Check the column name and verify that none are missing rules for any data line, as shown in the following image.

---

**Exporting the Final Model to Build the Scoring Application**

Once you have finalized your model, you can export the model formula as a routine that can be used to score new data outside of RStat.

RStat offer two types of export options:

- **C Routine.** To be deployed within a WebFOCUS scoring application.

- **PMML.** The statistical and data mining XML standard that can be used for in-database scoring.
Exporting Using C or PMML

You can export data using C or PMML. You use C so that the data will work with WebFOCUS and you use PMML in order to see the XML output of the model (or meta) information.

Procedure: How to Export Using C or PMML

You can use this procedure to export using C or PMML.

1. Select the Model tab, as shown in the following image.
2. Click Export from the RStat toolbar. The Export dialog box opens, as shown in the following image.
3. Select C Files or PMML Files from the Type drop-down list to view existing files of the selected type within the current path.

4. Select an Include option. These allow you to selectively include the PMML and Model Metadata in the C routine for further reference. They are embedded as text strings within the routine, which can be extracted using the RStat command in WebFOCUS (see Displaying Model Information With the RStat Query Command on page 187). Depending on the model you have built, these text strings can be very large. In certain environments, you will want to exclude these because they will be too large to successfully compile. Options include:

- PMML. Includes the PMML within the routine.
### Meta Data

Includes key model metadata within the routine. This is the model output from R which appears on the Model Textview on the Model Tab, as highlighted in the following image.

![Model Metadata](image)

**Note:** Clear the check box next to either of these include options to exclude these from the exported routine.

5. Select Teradata UDF to export scoring functions contained within C files to be consumed as a Teradata User Defined Function (UDF). For more information, see [RStat Export for Teradata User Defined Function](#) on page 176.

6. Click Save. The file containing your scoring routine will be generated and placed in the selected location and file name.

### RStat Export for Teradata User Defined Function

As of RStat Version 1.3.1, you can export Scoring functions contained within C files to be consumed as a Teradata User Defined Function (UDF). With the exception of the Teradata Export option, the export process is similar to the regular RStat export.
For Big Data Analytics, RStat routines can be deployed for in-database execution. This means that the actual scoring of large amounts of data is executed in the database engine, alleviating the need for extracting the data prior to scoring. When scoring large amounts of data, running the predictive model as an in-database function may result in significant performance gains.

Procedure: How to Create a C File for a Teradata User Defined Function

After creating the predictive model, click the Export button. Select the Teradata UDF option on the Export dialog box and then Execute. A C file is created in Teradata UDF format, along with the SQL template needed to define the C program to Teradata. All the input fields are automatically defined in the SQL template. However, the actual location of the C file that has been uploaded needs to be modified in the SQL template. Once you have the location and the SQL, the UDF creation process defined by Teradata should be followed.

Once defined as a Teradata UDF, the RStat model can then be defined as an in-database function to WebFOCUS, either at the Metadata or the WebFOCUS procedure level.

The power of an RStat predictive model in-database function means that you can easily incorporate the model into ETL, reporting, Dashboard, or any other native Teradata Client application.

1. In App Studio, from the Home tab, in the Modeling group, click Predictive Modeling.

RStat opens.
2. In the Filename field, on the Data tab, click the folder to browse and select a data file, as shown in the following image.

3. Click Open.

4. Click Execute to load the data.

RStat refreshes to display your data, as shown in the following image.

5. Click the Model tab and select your model.
6. Click *Execute*.
7. On the toolbar, click *Export* to export your data.
8. In the Export C or PMML dialog box, select the check box for Teradata UDF, as shown in the following image.

This creates a C file on your local drive that contains Teradata.

**Note:** You can save the C file to any location using the Browse for other folders options.

Along with the C file, an SQL template is created. The items in double angle brackets (<<,>>) need to be modified in the line below:

```
<< ENTER FULL PATH OF UDF C FILE HERE AND PUT IN SINGLE QUOTE >>
EXAMPLE 'CS!<<model name>>!/home/testdrive/ibi/apps/<<model name>>.c!F! <<model name>>'
```
The following image shows an SQL template that has been modified and pasted into a Basic Teradata Query (BTEQ) tool to create a Teradata UDF.
Using the UDF in an SQL Select, note that the custscore function name matches the C file deployment in Teradata.

The following image shows in-database scored values (Arrows) returned in a report.
RStat models are completely integrated with Information Builders tools. Using predictive models in ETL allows you to segment data based on customer market segmentation.

**Compiling and Deploying the Scoring Routine**

You can deploy an RStat scoring routine in any application directory within your app path on any WebFOCUS server. The process of deployment includes compiling the routine and placing the executable file in a location that can be accessed by a WebFOCUS scoring application.

**Procedure: How to Compile and Deploy the Scoring Routine**

1. In App Studio, on the Home tab, in the Modeling group, click *Model Deployment*, as shown in the following image.

2. Launch the Model Deployment dialog box in one of the following ways:
The Model Deployment dialog box opens, as shown in the following image.

3. Click **Add** to designate the C routine to be used as the deployment source.

   A file selection dialog box opens, automatically pointing to the app path defined as your default path in the model configuration options.
4. Navigate to the app directory containing the routines to be deployed, and select the routines. You can use standard Windows multi-select options to select multiple routines to be deployed in a single step, as shown in the following image.

![Model Deployment](image)

5. Click Open.
The selected files are displayed in the Deployment Source list and the Deployment Destination becomes available, as shown in the following image.
6. Navigate through the servers and application path to the desired deployment destination, as shown in the following image.

Once you have selected an application directory, the Deploy button will become active.

7. Click Deploy.

Once the compile and deploy process is complete, the Deploy Status window opens.
8. Verify that the deployment was successful, as shown in the following image.

9. Click OK on the Deploy Status window.
10. Click Close to exit the Model Deployment.

Displaying Model Information With the RStat Query Command

The RStat command provides access to model information stored in the scoring routines. Each routine contains information on the model created, the model parameters, the model metadata, and the PMML. This is designed to make it easier for the developer of the scoring application and future users of the scoring routine to know what the model does and how to access it successfully.
The RStat command takes two parameters:

- **MODELNAME.** The name of the scoring routine.

- **WHICHINFO.** Which model information to display.

The available values for WHICHINFO include:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFO</td>
<td>Model Name, Model Type, Creation Date, and Time.</td>
</tr>
<tr>
<td>PARM</td>
<td>Model parameters in the order in which they are defined and the required data type. The final parameter is always the target variable.</td>
</tr>
<tr>
<td>META</td>
<td>Metadata and model output.</td>
</tr>
<tr>
<td>PMML</td>
<td>PMML representation of the model.</td>
</tr>
</tbody>
</table>

You can access the RStat command through the Command Console or within a report.

**Accessing Model Information Through the Command Console**

In the Command Console, RStat has been implemented as a query command.

The syntax of the RStat command is:

```
? RSTAT MODELNAME WHICHINFO
```

where:

- **MODELNAME**
  - Is the name of the scoring routine.

- **WHICHINFO**
  - Is the type of information to be displayed.

1. In App Studio, on the Home tab, in the Utilities group, click *Command Console.*
2. Type an RStat command in the Command Console.

3. On the Quick Access bar, click Run.

You will see the information from your model presented in an HTML output window, as shown in the following image.
Building a Scoring Application Using a Scoring Routine

A scoring application enables you to score new data using an RStat scoring routine. Once deployed, scoring routines become user-defined functions (also known as FUSELIB routines). These routines can be used to apply the model formula to new data within any DEFINE or COMPUTE expression. You can define the expression containing the RStat scoring routine using the Function Arguments dialog box or by manually entering the WebFOCUS syntax.

In this example, we have generated and deployed a scoring routine based on a binary Decision Tree Model. Scoring routines can be generated for the following models: Decision Tree (numeric, binary and multinomial), Regression (linear, general linear, poisson, logistic, and multinomial) and Clustering (KMeans, Ewkm, and Hierarchical), NNet, Boost, Random Forest, Survival, Market Basket Analysis, and Association Rules.

RStat brings the power of predictive analytics to the operational enterprise. Any WebFOCUS application can select new data to be scored and then provide ad hoc analytics through active reports, plot the prediction on a map or graph, or support real-time decision-making through KPI dashboards and transactional process flows.

Procedure: How to Score New Data With RStat Scoring Routines in App Studio

1. On the Home tab, in the Content group, click Report, and then navigate through the prompts of the Report Wizard to select the appropriate Master File for the new applicant data set.

   You can use any data source that contains the input variables defined for the model. For the purposes of this example, use the AB_NewCustomers data file.
2. Add the following fields to the report: ID, AGE, EDUCATION, MARITAL, GENDER, OCCUPATION, and INCOME.

3. Right-click inside the Object Inspector, then click New Compute Virtual Field.

   The Computed Field Creator dialog box opens.

   Define the expression as the scoring function with your new data fields as the model input variables and the computed field (SCORE) as the final parameter.

   **Note:** Optionally, you may create a Define field. For the purposes of this example, create a Compute field.

4. Set the field name to SCORE.

5. Set the field format to A2.
6. Create the field expression containing the RStat function.

**Note:** You can define RStat scoring routines within a COMPUTE expression by using the Function Arguments dialog box or by creating a command using WebFOCUS syntax.

- **To build the RStat scoring expression using the Functions Arguments Dialog Box:**
  1. Click the *Functions* button to build the scoring expression.
The Function Arguments dialog box opens and lists all available built-in functions. A list of predefined functions are grouped into categories that include Character, Data Source and Decoding, Date and Time, Format Conversion, Numeric, and System. Each of the available functions is a routine that takes input parameters and returns a value.

2. Click **Retrieve Scoring Routines** to retrieve the RStat scoring routines available within the current application directory.

If any RStat Scoring Routines are found within the current application directory, RStat Scoring Routines is added as a category and the related functions appear in the Select a function list. Once the retrieve is completed successfully, the button text changes to Refresh Scoring Routines. You can use the Refresh button to update the list of available routines if you know that the new routines have been added to the current application directory or the existing ones have changed.
Note: If no routines are found, the button text changes to No Scoring Routines Found, the RStat Scoring Routine Category is not added to the available list, and the button is disabled.

3. Select `ab_creditscore_tree` from the function drop-down list.

The Input Parameters, Types, and Value columns appear for the function.

4. Drag the fields to be used as input parameters from the field list displayed on the right of the Functions Arguments dialog box, or type the field names directly into the value field. For this example, click the field name from the field list and drag the field name to the value field as follows:

- For the AGE input parameter, drag `AGE` into the Value field.
- For the EDUCATION input parameter, drag `EDUCATION` into the Value field.
- For the OCCUPATION input parameter, drag `OCCUPATION` into the Value field.
- For the INCOME input parameter, drag `INCOME` into the Value field.

5. Enter the name of the compute or define (for example, `SCORE`) as the format assigned to the target variable in the `Target CREDIT_APPROVAL (alpha)` input field.

6. Click `OK` to close the Function Arguments dialog box.

The function argument is added to the COMPUTE expression.

Create the RStat function by entering a command using WebFOCUS syntax:

Enter the following command in the expression window of the Computed Field Creator dialog box:
AB_CREDITSCORE_TREE(AGE, EDUCATION, OCCUPATION, INCOME, SCORE)

The COMPUTE expression appears as follows.

7. Click New to create a second Compute field to display YES if the score is 1 or No if the score is 0:

   APPROVED/A3 = IF SCORE EQ '1' THEN 'YES' ELSE 'NO';

The Compute tool generates the following expression:

   COMPUTE SCORE/A2 = ab_creditscore_tree(AB_NEWCUSTOMERS.SEG_01.AGE, AB_NEWCUSTOMERS.SEG_01.EDUCATION, AB_NEWCUSTOMERS.SEG_01.MARITAL, AB_NEWCUSTOMERS.SEG_01.GENDER, AB_NEWCUSTOMERS.SEG_01.OCCUPATION, AB_NEWCUSTOMERS.SEG_01.INCOME, 'A2');
   COMPUTE APPROVED/A3 = IF SCORE EQ '1' THEN 'YES' ELSE 'NO';

8. Click OK to close the Computed Field Creator dialog box.

9. Run the procedure to see the predicted values.

Missing Data in Scoring Routines

Missing data refers to variables that have no data value in the current observation or record. The missing or inapplicable value is indicated by the default character, a dot (.).

Some modeling algorithms cannot generate a score if any of the input parameters are missing and will return the score as missing. Other modeling algorithms can generate a score even if there are missing input parameter values.

Regression and clustering techniques will return a missing value for the score if any of the input parameters are missing.
Decision tree techniques will return a score even if there are missing input parameter values. If there is a missing value, the record is assigned to the majority class of the node in which the missing value occurs.

**Recognizing Missing Input Parameters**

In order for a scoring routine to recognize missing input parameter values and for the algorithm to derive the score appropriately, you must add the SET MISSINGTEST command to the procedure (fex) and the MISSING attribute to the individual calculated field.

1. Set the MISSINGTEST command by either:
   - Adding the following SET command to the procedure (fex):
     ```plaintext
     SET MISSINGTEST = SPECIAL
     ```
   - Or, adding the following ON TABLE SET command to the report request:
     ```plaintext
     ON TABLE SET MISSINGTEST SPECIAL
     ```

2. Add the MISSING attribute to the individual calculated field (MISSING ON). For example:
   ```plaintext
   COMPUTE PREDICTION/D20.8CM MISSING ON =
   W_REG_LINEAR(WRAIN, DEGREES_IN_C, HRAIN,
   TIME_SINCE_VINTAGE, CHATEAU,PREDICTION);
   ```

**Example:** **Calling an RStat Scoring Routine Function**

The following report request includes WebFOCUS syntax that calls the w_reg_linear function, a scoring routine built from a linear regression model. It is designed to account for the possibility that some of the input values may be missing.

**Note:** Scoring routine functions cannot be embedded in other formulas or expressions. The expression on the right side of the command must consist only of the function call.
SET MISSINGTEST=SPECIAL
FILEDEF W_REG_LINEAR DISK
C:\IBI\APPS\rstat\w_reg_linear.CSV
TABLE FILE W_REG_LINEAR
PRINT
  ID
  CHATEAU AS 'Chateau'
  WRAIN/D6.0 AS 'Winter,Rain,(inches)'
  DEGREES_IN_C/D6.0 AS 'AvgTemp,(Celsius)'
  HRAIN/D6.0 AS 'Harvest,Rain,(inches)'
  TIME_SINCE_VINTAGE/I5 AS 'Years,Since,Vintage'
COMPUTE PREDICTION/D20.8CM MISSING ON =
  W_REG_LINEAR(WRAIN, DEGREES_IN_C, HRAIN,
  TIME_SINCE_VINTAGE, CHATEAU,PREDICTION);
HEADING
"Regression Linear"
ON TABLE SET PAGE-NUM OFF
ON TABLE NOTOTAL
ON TABLE PCHOLD AS W_REG_LINEAR.PDF FORMAT PDF
ON TABLE SET STYLE *
  UNITS=IN,
  PAGESIZE='Letter',
  SQUEEZE=ON,
  ORIENTATION=LANDSCAPE,
$ TYPE=REPORT,
  FONT='TREBUCHET MS',
  SIZE=9,
  COLOR=RGB(66 70 73),
.
.
ENDSTYLE
END
The partial output is shown in the image below. By default, the missing value is represented by a dot (.) on the report output. You can change this character designation by using the SET NODATA command. For more information on changing the missing data character or syntax on handling records with missing data in a report request, see the *Handling Records With Missing Field Values* chapter in the *Creating Reports With WebFOCUS Language* manual.

<table>
<thead>
<tr>
<th>ID</th>
<th>Chateau</th>
<th>Winter Rain (inches)</th>
<th>AvgTemp (Celsius)</th>
<th>Harvest Rain (inches)</th>
<th>Years Since Vintage</th>
<th>PREDICTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>Lafite</td>
<td>763.</td>
<td>16.</td>
<td>290.</td>
<td>23</td>
<td>$506.74900572</td>
</tr>
<tr>
<td>2.00</td>
<td>Lafite</td>
<td>.</td>
<td>17.</td>
<td>38.</td>
<td>22</td>
<td>.</td>
</tr>
<tr>
<td>3.00</td>
<td>Lafite</td>
<td>697.</td>
<td>.</td>
<td>52.</td>
<td>21</td>
<td>.</td>
</tr>
<tr>
<td>4.00</td>
<td>Lafite</td>
<td>608.</td>
<td>16.</td>
<td>.</td>
<td>20</td>
<td>.</td>
</tr>
<tr>
<td>5.00</td>
<td>Lafite</td>
<td>402.</td>
<td>17.</td>
<td>96.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>6.00</td>
<td>Lafite</td>
<td>602.</td>
<td>15.</td>
<td>267.</td>
<td>18</td>
<td>.</td>
</tr>
<tr>
<td>7.00</td>
<td>Lafite</td>
<td>819.</td>
<td>17.</td>
<td>86.</td>
<td>17</td>
<td>$967.12191016</td>
</tr>
<tr>
<td>8.00</td>
<td>Lafite</td>
<td>714.</td>
<td>16.</td>
<td>118.</td>
<td>16</td>
<td>$674.33116827</td>
</tr>
<tr>
<td>9.00</td>
<td>Lafite</td>
<td>610.</td>
<td>16.</td>
<td>292.</td>
<td>15</td>
<td>$185.08656569</td>
</tr>
<tr>
<td>10.00</td>
<td>Lafite</td>
<td>575.</td>
<td>17.</td>
<td>244.</td>
<td>14</td>
<td>$346.23273512</td>
</tr>
<tr>
<td>11.00</td>
<td>Latour</td>
<td>763.</td>
<td>16.</td>
<td>290.</td>
<td>23</td>
<td>$625.61510858</td>
</tr>
<tr>
<td>12.00</td>
<td>Latour</td>
<td>830.</td>
<td>17.</td>
<td>38.</td>
<td>22</td>
<td>$1,497.99647011</td>
</tr>
<tr>
<td>13.00</td>
<td>Latour</td>
<td>.</td>
<td>16.</td>
<td>52.</td>
<td>21</td>
<td>.</td>
</tr>
<tr>
<td>14.00</td>
<td>Latour</td>
<td>608.</td>
<td>.</td>
<td>155.</td>
<td>20</td>
<td>.</td>
</tr>
<tr>
<td>15.00</td>
<td>Latour</td>
<td>402.</td>
<td>17.</td>
<td>.</td>
<td>19</td>
<td>.</td>
</tr>
<tr>
<td>16.00</td>
<td>Latour</td>
<td>602.</td>
<td>15.</td>
<td>267.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>17.00</td>
<td>Latour</td>
<td>.</td>
<td>17.</td>
<td>86.</td>
<td>17</td>
<td>$793.19727113</td>
</tr>
<tr>
<td>18.00</td>
<td>Latour</td>
<td>714.</td>
<td>16.</td>
<td>118.</td>
<td>16</td>
<td>$303.95266855</td>
</tr>
</tbody>
</table>
Creating Statistically Valid Data Samples

Advanced sampling techniques enable you to generate statistically representative data extracts for data examination and modeling purposes.

The sampling is executed after aggregation and selection criteria (WHERE/IF) clauses are applied to the data.

In this chapter:

- Sampling Functionality
- Basic Sampling Concepts and Terminology
- Sampling Steps

Sampling Functionality

The sampling functionality is accessible through the Sampling function within the Modeling tab in App Studio. There are three sampling techniques: Random, Numeric Stratified, and Categorical Stratified.

- **Random Sampling** is a basic sampling technique where you select a group of subjects (a sample) from a larger group (a population). Each individual is chosen entirely by chance and each member of the population has an equal chance of being included in the sample.

- **Stratified Sampling** is obtained by taking separate samples from each sub-group of a population (commonly referred to as a stratum).

When you sample a population, stratified sampling allows us to ensure that the proportion of each stratum in the sample is the same as in the population. Numeric stratified sampling is used when the field identifying the strata is a number and can be defined in terms of a range. Categorical stratified sampling is used for all other data.
Basic Sampling Concepts and Terminology

To produce a valid output sample, each sampling definition requires information about the file to be sampled. In each sampling routine you can use the sample calculator to calculate the values for each of the following or enter your own values:

- **Population Size.** The number of records in the overall data set to be sampled. Before performing a sampling routine, the population size must be determined. Within the sampling function, the population size can be retrieved at design time, generated live at run time, or defined as a fixed value. Although the population size is usually the total number of records in the file, there will be times when you want to sample a subset of the file. To ensure a statistically valid sample, the specified population size must be the same as the number of records sampled.

- **Confidence Level.** Is the probability, expressed in percent, that the selected sample will represent the total population. Most guidelines establish a minimum acceptable confidence level of 90%, 95%, or 98%.

- **Margin of Error.** Represents the amount of error, expressed in percent, that you can tolerate. Lower margins of error require larger sample sizes.

- **Response Distribution.** Allows you to correct for skewness in the sample (if the sample deviates from the normal standard deviation). Use the Response Distribution percentage to account for the skewness in population.

- **Seed.** The statistical sampling routines use a seed value for the pseudo random number generator. This generator produces a series of random numbers from the entered seed. These random numbers are then used to determine each record to include in the sample. The seed has no affect on the number of records included in the sample, it only affects which records are selected. A single seed will produce the same set of random numbers, so if you want to replicate a sample, use the same seed, population size, and record order. To generate a unique sample, enter a new seed value each time.

- **Sample Size.** The number of records that you want to store in the output file. The number should be a positive integer, greater than zero and less than the population size.
Sampling Steps

Sampling consists of the following steps, each of which will be described in detail:

1. Selecting the sampling technique to be executed:
   
   - Random
   - Numeric Stratified
   - Categorical Stratified

2. Defining the sampling parameters. For each of the sampling techniques, identify the population size to be sampled and then use either of two available modes to determine the optimal sample size:
   
   - Calculate Sample Size. Define values for the factors in determining the sample size in a sample calculator to automatically determine the recommended sample size.
   - Specify Sample Size. Enter a fixed population or sample size to be used in the extract calculations.

3. Executing the report.

   To confirm the number of records included in the sample extract, you can run the procedure with the Message Viewer on. This provides both the selected output and the messages confirming the sample that was extracted. For information on how to turn the Message Viewer on, see How to Set Message Viewer On on page 222.

Procedure: How to Extract a Sample Using Random Sampling in App Studio

To extract a sample using random sampling functionality:

1. In the Report canvas, add the fields to be included in the sample extract.

2. On the Modeling tab, in the Modeling group, click Sampling.

   The Sampling dialog box opens.
3. Select the Random radio button for the sampling technique, as shown in the following image.
4. Identify the population size by clicking the *Retrieve Population* button to retrieve the current total record count. The current record count will be displayed on the retrieve button, as shown in the image below. The sample size within the calculator will automatically be updated with the recommended sample size for the population size based on the defined margin of error, confidence level, and response distribution.

![Sampling dialog box](image)

5. Using the Population drop-down list, define which population will be used to determine the sample size.

- **Live.** Select *Live* from the Population drop-down list to cause the total record count to be recalculated at run time. Live is the default value.

- **Design-Time Count.** When you retrieve the population size during design time using the Retrieve Population button, the value identified is added to the population drop-down list. You can select this as your population size for the run-time execution. Use this if you are certain the population will not change between executions or if an estimate is sufficient for your circumstances.
User-defined Count. Enter any value for population by typing a numeric value into the Population list box. Note that the Recommended Sample Size automatically updates as you type new values for the population size, as shown in the image below.

6. Define the seed value by using the default seed value or by typing a new value into the Seed input box to be used to generate the random sample.

7. Select how the sample size will be defined by selecting one of the following options:

   **Calculate Sample Size**
   a. Click the Calculate Sample Size radio button to activate the sample calculator.
   b. Use the provided default values or type new values for:
      - Margin of Error (default value is 5).
      - Confidence Level (default value is 95).
      - Response Distribution (default value is 50).
The Recommended Sample Size will be calculated and displayed, as shown in the image below.

Specify Sample Size

Click the Specify Sample Size radio button to specify a user-defined sample size.

- To specify the sample size as a count, type a numeric value into the User Specified Sample Size input box.
The Expected Margin of Error, based on the identified Population size and your specified Sample size, will be displayed, as shown in the image below.

To specify the sample size as a percent, click the **As a percent** check box, and enter the User Specified Sample Size as a number between 1 and 100 to represent the percentage of the overall population to be used as the sample.

The Expected Margin of Error based on the identified Population size and your specified Sample size will be displayed.

8. To execute the extraction with the defined random sampling, close the Sampling dialog box by clicking **OK** and run the report.

**Procedure:** How to Extract a Sample Using Numeric Stratified Sampling in App Studio

To extract a sample using numeric stratified sampling based on user-defined ranges of a numeric field within the data:

1. In the Report canvas, add the fields to be included in the sample extract.
2. On the Modeling tab, in the Modeling group, click **Sampling**.
   
   The Sampling dialog box opens.
3. Select the *Numeric Stratified* radio button for the sampling technique, as shown in the image below.
4. From the list of available numeric fields in the current report, select the field to be used to define the stratum criteria, as shown in the following image.

Each stratum or group is defined by the upper limit for the data value of the current field. Strata are selected in the order they are defined. Therefore, the criteria must be entered in ascending order. In the example below, we will divide our population into four strata based on the values for Profit.

<table>
<thead>
<tr>
<th>Strata</th>
<th>Criteria Value Entered</th>
<th>Who Qualifies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stratum 1</td>
<td>5000</td>
<td>-∞ Profit ≤ 5000</td>
</tr>
<tr>
<td>Stratum 2</td>
<td>10000</td>
<td>5000 &lt; Profit ≤ 10000</td>
</tr>
<tr>
<td>Stratum 3</td>
<td>50000</td>
<td>10000 &lt; Profit ≤ 50000</td>
</tr>
<tr>
<td>Stratum 4</td>
<td>999999</td>
<td>50000 &lt; Profit ≤ 999999</td>
</tr>
</tbody>
</table>

**Note:**

- Numeric strata criteria must be entered in ascending order to return the correct results.
Any data values that fall outside of the defined criteria ranges will not be included in the sample selection.

5. Use the strata toolbar to manage your strata criteria. The strata toolbar is shown in the following image:

- To add a new stratum, either double-click in the next available criteria cell or click the New icon.
- Enter each criteria value. Press Enter to close the current row and add the new criteria row.
- To remove strata criteria, select the row to delete and click the X (delete) icon on the toolbar or close the edit box for the strata criteria with an empty value.
- To rearrange the order of strata criteria, use the move item up and move item down icons to arrange the strata criteria in ascending order.

The example of strata criteria is shown in the following image.
6. Once your strata are defined, click the *Retrieve Population* button to identify the record count and calculate the recommended sample size for each defined stratum, as shown in the following image.

![Sampling Steps](image)

7. For each stratum, you can choose from the following options to define the overall population from which to select your sample.

- **Live.** Select Live from the population drop-down list to cause the record count to be recalculated at run time. Live is the default value.
iqué Design-Time Count. When you retrieve the population size during design time using the Retrieve Population button, the value identified is added to the population drop-down list, as shown in the following image. You can select this as your population size for the run-time execution. Use this if you are certain the population will not change between executions or if an estimate is sufficient for your circumstances.

![Sampling Window](image)

- **User-defined Count.** Enter any value for population by typing a numeric value into the Population list box. Note that the Recommended Sample Size automatically updates as you type new values for the population size.

8. Select how the sample size will be defined by selecting one of the following options:

**Calculate Sample Size**

a. Click the *Calculate Sample Size* radio button to activate the sample calculator. By default, the Calculate Sample Size radio button is selected and the sample calculator is active.

b. Use the provided default values or type new values for:

- **Margin of Error** (default value is 5).
- **Confidence Level** (default value is 95).
Response Distribution (default value is 50). Response Distribution is defined at the stratum level.

**Note:** The value defined in the calculator is used as the default value when new stratum are created. If you want to change the default to be used each time you create a new criteria row, set the default value within the Default Response Distribution field box.

The Recommended Sample Size will be calculated and displayed, as shown in the image below.

![Sampling window](image)

**Specify Sample Size**

Click the Specify Sample Size radio button, as shown in the following image.
By default, each of the stratum sample sizes is set to the value of 1, which is the minimum value allowed.
The sample size can be specified as a count or as a percentage of each stratum population.

- To specify the sample size as a count, type a numeric value into the Sample Size cell in the Strata grid for each stratum in your sample, as shown in the following image.

![Sampling Dialog Box](image)

- To specify the sample size as a percent, click the *As a percent of the population* check box. Within the Stratum grid, type a numeric value between 0 and 100 to represent the percentage of the total records in the current stratum that should be included. This number should represent the proportion of the individual stratum, not the overall population.

**Note:** Click the *Retrieve Population* button to refresh and calculate the Expected Margin of Error based on the identified Population size and your overall strata sample sizes.

9. To execute the extraction with the defined Numeric Stratified Sampling, close the Sampling dialog box by clicking *OK* and run the report.
Procedure: How to Extract a Sample Using Categorical Stratified Sampling in App Studio

To extract a sample using categorical stratified sampling based on alphanumeric categories within the data:

1. In the Report canvas, add the fields to be included in the sample extract.
2. On the Modeling tab, in the Modeling group, click Sampling.
   The Sampling dialog box opens.
3. Select the Categorical Stratified radio button for the sampling technique, as shown in the following image.
   ![Sampling dialog box image]
4. From the list of available fields in the current report, select the field to be used to define the stratum criteria.
Note that all fields (character and numeric) that you selected for your report are available within the Stratum Field drop-down list.

5. Click the **Get Data** button to retrieve the values of the field selected. This retrieves all available values for your stratum field and makes them available for selection as your stratum criteria.

6. Use the strata toolbar to manage your strata criteria. The strata toolbar is shown in the following image:

![Strata Toolbar](image)

- **To add a new stratum**, either double-click in the first criteria cell or click the **New** icon and select a value from the drop-down list.

  - **Enter each criteria value** or select the value from the drop-down list. Press Enter to close the current row and add a new criteria row.

- **To remove strata criteria**, select the row to delete and click the **X (delete)** icon on the toolbar or close the edit box for the strata criteria with an empty value.

- **To rearrange the order of strata criteria**, use the move item up and move item down icons to arrange the strata criteria.
The example of strata criteria is shown in the following image.

Note: Any data values that fall outside of the defined criteria will not be included in the sample selection.
7. Click the *Retrieve Population* button to identify the record count and calculate the Recommended Sample Size for each defined stratum.

![Sampling Steps](image)

8. For each stratum, you can choose from the following options to define the population to use to extract your sample.

- **Live.** Select *Live* from the population drop-down list to recalculate the record count at run time. *Live* is the default value.
**Design-Time Count.** When you retrieve the population size during design time using the Retrieve Population button, the value identified is added to the population drop-down list, as shown in the following image. You can select this as your population size for the run-time execution. Use this if you are certain the population will not change between executions or if an estimate is sufficient for your circumstances.

![Sampling](image)

**User-defined Count.** Enter any value for population by typing a numeric value into the Population list box. Note that the Recommended Sample Size automatically updates as you type new values for the population size.

9. Select how the sample size will be defined by selecting one of the following options:

**Calculate Sample Size**

a. Click the Calculate Sample Size radio button to activate the sample calculator. By default, the Calculate Sample Size radio button is selected and the sample calculator is active.

b. Use the provided default values or type new values for:

   - Margin of Error (default value is 5).
   - Confidence Level (default value is 95).
Response Distribution (default value is 50). Response Distribution is defined at the stratum level.

**Note:** The value defined in the calculator is used as the default value when new stratum are created. If you want to change the default to be used each time you create a new criteria row, set the default value within the Default Response Distribution field box.

The Recommended Sample Size will be calculated and displayed, as shown in the image below.

![Sampling Steps](image)

**Specify Sample Size**

Select the Specify Sample Size radio button, as shown in the following image.
By default, each of the stratum sample sizes is set to the value of 1, which is the minimum value allowed.
The sample size can be specified as a count or as a percentage of the overall population.

- To specify the sample size as a count, type a numeric value into the Sample Size cell in the Strata grid for each stratum in your sample, as shown in the following image.

![Sampling dialog box with sample size specification](image)

- To specify the sample size as a percent, click the As a percent of the population check box. Within the Stratum grid, type a numeric value between 0 and 100 to represent the percentage of the total records in the current stratum that should be included. This number should represent the proportion of the individual stratum, not the overall population.

**Note:** Click the Retrieve Population button to refresh and calculate the Expected Margin of Error based on the identified Population size and your overall strata sample sizes.

10. To execute the extraction with the defined Numeric Stratified Sampling, close the Sampling dialog box by clicking OK and run the report.

**Procedure:** **How to Set Message Viewer On**

To confirm the number of records included in the sampled extract, you can run the procedure with the Message Viewer on. This provides both the selected output and the messages confirming the sample that was extracted.
In App Studio, select Message Viewer ON from the Run menu within the Application menu, as shown in the following image.

![Message Viewer Options](image)

**Procedure:** How to Disable Sampling

By default, sampling is set to disabled. To select a sampling technique, change the state of the Sampling radio button.

**Important:** To remove sampling definitions that you have previously defined, change the Sampling technique back to Disable Sampling. The current sampling definitions are retained in the current procedure (fex), until you save the procedure, but not used during report execution. Once you save a procedure with sampling disabled, any previously defined settings are removed.
Building a Linear Regression Model

WebFOCUS RStat is a statistical modeling workbench embedded in a WebFOCUS desktop product, such as App Studio, that enables data exploration, hypothesis testing, data mining, and model development for scoring applications. RStat enables data miners and Business Intelligence developers to work collaboratively with the same tools to access, manipulate, or transform data, develop predictive models, and create and deploy scoring applications, along with associated reports, to any worker within their organization.

The WebFOCUS RStat tool logically proceeds by progressing through the tabs: first load the data, select variables for exploring and mining, possibly sample the data, explore the data, build your models, and evaluate them.

The Regression, Decision Tree, and Cluster models are the most commonly used models for predictive analytics.

This chapter explains how regression works with one dependent and one independent variable (simple regression), and how regression works with multiple independent variables (multiple regression).

In this chapter:

- Explanation of the Regression Model

Explanation of the Regression Model

What Is Regression Analysis?

Regression analysis is the method of using observations (data records) to quantify the relationship between a target variable (a field in the record set), also referred to as a dependent variable, and a set of independent variables, also referred to as a covariate. For example, regression analysis can be used to determine whether the dollar value of grocery shopping baskets (the target variable) is different for male and female shoppers (gender being the independent variable). The regression equation estimates a coefficient for each gender that corresponds to the difference in value.
The value of quantifying the relationship between a dependent variable and a set of independent variables is that the contribution of each independent variable to the value of the dependent variable becomes known. Once this is known, you need to know only the values of the independent variables in order to be able to make predictions about the value of the dependent variable. For example, when the coefficients for male and female shoppers are known, you can make precise revenue estimates for different distributions of shoppers. Specifically, you can predict the revenues for 80/20 females/males versus 60/40 females/males.

The goal of regression analysis is to generate the line that best fits the observations (the recorded data). The rationale for this is that the observations vary and thus will never fit precisely on a line. However, the best fitted line for the data leaves the least amount of unexplained variation, such as the dispersion of observed points around the line. Stated differently, a relationship that explains 90% of the variation in the observations is better than one that explains 75%. Conversely, a relationship with a better fit has a better predictive power.

**How Does Linear Regression Work?**

To illustrate how linear regression works, examine the relationship between the prices of vintage wines and the number of years since vintage. Each year, many vintage wine buyers gather in France and buy wines that will mature in 10 years. There are many stories and speculations on how the buyers determine the future prices of wine. Is the wine going to be good 10 years from now, and how much would it be worth? Imagine an application that could assist buyers in making those decisions by forecasting the expected future value of the wines. This is exactly what economists have done. They have collected data and created a regression model that estimates this future price. The current explanation of the regression is based on this model.

The provided sample data set contains 60 observations of prices for vintage wines that were sold at a wine auction. There are two columns, one for the price of each wine and another for the number of years since vintage. The data set, referred to as the training data set, is used to estimate the parameters in the equation.

The estimated equation can be used to predict wine prices when the year since vintage is known. In other words, it can be applied to another data set, referred to as the test data set, which contains only vintage years. Prices are derived by applying the equation to the vintage years in the training data set.
The following formula describes the linear relationship between price and years:

\[ y = \beta_0 + \beta_1 x + error \]

- **Dependent Variable Y.** Y represents the price of each of the vintage wines observed in the auction.

- **Independent Variable X.** X is the time since vintage for each of the vintage wines observed in the auction. It is also referred to as a covariate.

- **Parameters.** \( \beta_0 \) and \( \beta_1 \) are parameters that are unknown and will be estimated by the equation. WebFOCUS RStat estimates these parameters when you run the regression model.

- **Intercept.** \( \beta_0 \) is a constant that defines where the linear trend line intercepts the Y-axis.

- **Coefficient.** \( \beta_1 \) is a constant that represents the rate of change in the dependent variable as a function of changes in the independent variable. It is the slope of the linear line, for example, it shows how prices increase with the increase in the number of vintage years, or how wines become more expensive the longer they mature. For other data sets, the trend can be the inverse, that is, the slope can be decreasing.

- **Error.** Represents the unexplained variation in the target variable. It is treated as a random variable that picks up all the variation in Y that is not explained by X.
The following image visually examines the nature of the relationship by plotting $Y$ and $X$ on a scatter plot with the trend line.

- All observations are plotted on the scatter plot.
- The linear trend illustrates the trend in the observed data.
- Variation shows the dispersion of the data points around the trend line.
- The ellipse shows the points that closely fit the line.
- The dashed square shows the observations that do not closely fit the line. These are referred to as outliers. Outliers are typically removed from the modeling data set. They can be filtered out. For example, the preceding scatter plot image above shows all points within the 1100 to 1700 price range as outliers. You can either filter out those outliers, or segment the data into different price ranges and build different regression models for each segment.

The following regression equation shows estimated parameters for the trend line:

$$y = -533 + 55 \times x$$

To predict the price for a 15-year-old vintage wine, substitute $x=15$: 

\[ y = -533 + 55 \times 15 \]
\[ y = -533 + 825 \]
\[ y = 292 \]

**How Does Multiple Linear Regression Work?**

A simple linear equation rarely explains much of the variation in the data and for that reason, can be a poor predictor. In the case of vintage wine, *time since vintage* provides very little explanation for the prices of wines. The regression equation described in the simple linear regression section will poorly predict the future prices of vintage wines. Multiple linear regression enables you to add additional variables to improve the predictive power of the regression equation. On a very intuitive level, the producer of the wine matters. Thus, including the chateau as another independent variable is likely to increase the predictive power of the equation.

The following equation shows a multiple linear regression equation. The equation is conceptually similar to the simple regression equation, except for parameters \( \beta_2 \) through \( \beta_n \), which represent the additional independent variables.

\[ y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \text{error} \]

The independent variables in the vintage wine data set are:

- **T (numeric variable).** Is the average seasonal temperature in Celsius.
- **WR (numeric variable).** Is the winter rain from October to March.
- **HR (numeric variable).** Is the rain during the harvest season from August to September.
- **TSV (numeric variable).** Is the time since vintage, such as the number of years from the year in which the wine was produced until it was sold.
- **Chateau (categorical variable).** Is the name of the chateau in which the wine was produced.

The regression equation estimates a single parameter for the numeric variables and separate parameters for each unique value in the categorical variable. For example, there are six chateaus in the data set, and five coefficients. One chateau is used as a base against which all other chateaus are compared, and thus, no coefficient will be estimated for it. In other words, if prices have to be predicted for the reference chateau, 0 is entered in the equation.

The equation with the estimated parameters is:

\[
\text{Price} = -4744 + 1.03 \times \text{WR} + 295.84 \times \text{T} - 2.09 \times \text{HR} + 8.06 \times \text{TSV} + \text{Chateau}
\]

The coefficients for each chateau are given in the output, as shown in the following image.
### Explanation of the Regression Model

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | -4743.7592 | 1027.7697 | -4.616 | 0.00002762 *** |
| WRAIN | 1.0287 | 0.2513 | 4.093 | 0.000155 *** |
| DEGREES_IN_C | 295.8356 | 58.8231 | 5.029 | 0.00000672 *** |
| HRAIN | -2.0854 | 0.3686 | -5.658 | 0.00000074 *** |
| TIME_SINCE_VINTAGE | 8.0633 | 10.9301 | 0.738 | 0.464133 |
| CHATEAUCos d'Estournel | -280.2000 | 99.3542 | -2.820 | 0.006863 ** |
| CHATEAULafite | 7.8000 | 99.3542 | 0.079 | 0.937738 |
| CHATEAULatour | 177.9000 | 99.3542 | 1.791 | 0.079418 . |
| CHATEAUMontrose | -320.8000 | 99.3542 | -3.229 | 0.002198 ** |
| CHATEAUPichon Lalande | -228.4000 | 99.3542 | -2.299 | 0.025730 * |

**Tip:** The image below shows how the estimated parameter values are used as the coefficient values in the regression equation.

\[
\text{Price} = 1.03 \times \text{WR} + 295.84 \times \text{T} - 2.09 \times \text{HR} + 8.06 \times \text{TSV} + \text{Chateau}
\]

### Note:
Cheval Blanc is the reference chateau with a coefficient of 0. Therefore, it is not listed in the output. However, when scoring data, the predicted prices for the Cheval Blanc will be in the scoring output. For more information about the output, see *Output From Linear Regression* on page 233.

Vintage wine prices can be predicted by substituting the values for the independent variables, as in the simple regression equation used earlier.

A scoring application automatically generates the predicted prices, either for a single record or for a batch of records.
Practical Applications of Regression Analysis

Regression analysis is used for:

- **Predictive Modeling.** Regression is used most frequently for prediction. Credit scoring applications use input variables (data collected on the applicant) to predict the likelihood that the applicant will repay the loan. In budgeting and finance, regression is used to estimate the relationship between profit and cost, which later can be used in applications to generate what if scenarios.

- **Segmentation.** Regression is used to segment or to determine the lifetime value of customers. For example, a retailer may segment category purchases and baskets based on age groups and gender, thus creating a more targeted marketing campaign.

- **Testing.** Regression is used to evaluate the performance of marketing campaigns. It can test whether a campaign has increased the dollars spent in the test period compared to the prior period or whether certain age groups spend more than others, and so on.

**Procedure:** How to Create a Multiple Linear Regression (lm)

**Note:** The following example creates a multiple linear regression, using the wine_train data source, and targets PRICE_1991.

1. Define the Model Data.
   - Load the Wine data into RStat. For more information on loading data into RStat, see *Getting Started With RStat* on page 33.
   - Select Ident for the ID variable.
   - Select Ignore for the Vintage_Year variable.
   - Select Target for the Price_1991 variable.
   - Keep all the other variables as Input.
   - Click Execute to set up the Model Data.
The status bar confirms your data settings, as shown in the following image.

2. Select the Model tab to select the Model Data.

☐ Select Regression as the Type of model.

☐ Select Linear as the Model Builder. This is the default option when Regression is selected.

Note: Other available regression models are Generalized (Gaussian Regression model) and Poisson Regression models. Both models are built using Regression Generalized (glm).

☐ Click Execute to run the model.
The model data output appears in the bottom window of the Model tab, as shown in the following image.

![Image of Model tab with regression model output]

For details about the regression output, see Output From Linear Regression on page 233.

**Reference:** Output From Linear Regression

This section describes the linear regression output. Note that output may vary slightly due to sampling.

- **Summary of the Regression model (built using lm).** This is the title of the summary provided for the model. It also specifies which R function has been used to build the model. The model in this case is built with the `lm` function.

  Summary of the Regression model (built using `lm`):

- **R Function Call.** This section shows the call to R and the data set or subset used in the model. `lm()` indicates that we used the linear regression function in R and `c(3:8)` indicates that columns 3 to 8 from the data set were used in the model.
lm(formula = Price_1991 ~ ., data = crs$dataset[, c(3:8)])

- **Residuals.**
  
<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-404.62</td>
<td>-135.97</td>
<td>-9.47</td>
<td>95.58</td>
<td>4484.79</td>
</tr>
</tbody>
</table>

- **Distribution of the Residuals.** This shows the distribution of the residuals. The residuals are the difference between the prices in the training data set and the predicted prices by this model. A negative residual is an overestimate and a positive residual is an underestimate. Ideally, you should see a symmetrical distribution with a median near zero. In this case, the median of -9.47 is very close to zero.

**Coefficients:**

| Coefficient Name | Estimate | Std. Error | t value | Pr(>|t|) |
|------------------|----------|------------|---------|---------|
| (Intercept)      | -5191.3914 | 1405.3794 | -3.694  | 0.000820 *** |
| WRAIN            | 0.9171    | 0.3502    | 2.619   | 0.013373 * |
| Degrees_in_C     | 329.9655  | 83.0036   | 3.975   | 0.000375 *** |
| HRAIN            | -1.8965   | 0.4753    | -3.990  | 0.000360 *** |
| Time_Since_Vintage | 7.3211   | 13.6187   | 0.538   | 0.594587 |
| ChateauCos d’Estournel | -336.9234 | 130.7021 | -2.578  | 0.014753 * |
| ChateauLafite    | -70.7367  | 128.0801  | -0.552  | 0.584590 |
| ChateauLatour    | -27.4525  | 136.1005  | -0.202  | 0.841422 |
| ChateauMontrose  | -365.6623 | 140.1642  | -2.609  | 0.013698 * |
| ChateauPichon Lalande | -283.3606 | 120.9652 | -2.342  | 0.025538 * |

- **Coefficient Name.** Column 1 displays the names of the coefficients. Notice that for categorical variables, all values except the reference value are listed. For example, five out of six chateaus are listed. See How to Create a Multiple Linear Regression (lm) on page 231.

- **Estimate.** These are the estimated values for the coefficients. Except for the reference chateau, notice the separate coefficient for each unique value of the categorical variable. The displayed coefficients are not standardized, for example, they are measured in their natural units, and thus cannot be compared with one another to determine which one is more influential in the model. Their natural units can be measured on different scales, as are temperature and rain.

- **Standard Error.** These are the standard errors of the coefficients. They can be used to construct the lower and upper bounds for the coefficient. An example is Coefficient ± Standard Error, which provides an indication where the value may fall if another sample data set is used. The standard error is also used to test whether the parameter is significantly different from 0. If a coefficient is significantly different from 0, then it has impact on the dependent variable (see t-value below).
- **t-value.** The t-value is the ratio of the regression coefficient $\beta$ to its standard error ($t = \frac{\text{coefficient}}{\text{standard error}}$). The t statistic tests the hypothesis that a population regression coefficient is 0. If a coefficient is different from zero, then it has a genuine effect on the dependent variable. However, a coefficient may be different from zero, but if the difference is due to random variation, then it has no impact on the dependent variable. In this example, Time_Since_Vintage is different from zero due to random variation and thus has no real impact on the dependent variable. The t-values are used to determine the P values (see below).

- **Pr(>|t|).** The P value indicates whether the independent variable has statistically significant predictive capability. It essentially shows the probability of the coefficient being attributed to random variation. The lower the probability, the more significant the impact of the coefficient. For example, there is less than a 1.3% chance that the WRAIN impact is due to random variation. The P value is automatically calculated by R by comparing the t-value against the Student’s T distribution table. As a rule, a P value of less than 5% indicates significance. In theory, the P value for the constant could be used to determine whether the constant could be removed from the model.

- ***.** The asterisks in the last column indicate the significance ranking of the P values.

- **Signif. codes:**

```
0 '****' 0.001 '***' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The significance codes indicate how certain we can be that the coefficient has an impact on the dependent variable. For example, a significance level of 0.01 indicates that there is less than a 0.1% chance that the coefficient might be equal to 0 and thus be insignificant. Stated differently, we can be 99.9% sure that it is significant. The significance codes (shown by asterisks) are intended for quickly ranking the significance of each variable.

- **Residual standard error:** 237.4 on 37 degrees of freedom.

- **Multiple R-squared:** 0.7958

- **Adjusted R-squared:** 0.7384

- **F-statistic:** 13.86 on 9 and 32 DF, p-value: 0.000000009629.

- **Residual Standard Error.** This is the standard deviation of the error term in the regression equation (see Simple Regression, Error). The sample mean and the standard error can be used to construct the confidence interval for the mean. For example, it is the range of values within which the mean is expected to be if another representative data set is used.
Degrees of Freedom (Df). This column shows the degrees of freedom associated with the sources of variance. The total variance has N - 1 degrees of freedom. A data set contains a number of observations - 60 in the vintage wine data set. The cases are individual pieces of information that can be used either to estimate parameters or variability. Each item (coefficient) being estimated costs one degree of freedom; the rest are used to estimate variability. For the vintage wine data set, you are estimating nine coefficients. Therefore, there are 50 degrees of freedom for the residuals.

For example, 50 = 60 - 1 - 9.

R-squared. R-squared is a measure of the proportion of variability explained by the regression. It is a number between zero and one, and a value close to zero suggests a poor model. In a multiple regression, each additional independent variable may increase the R-squared without improving the actual fit. An adjusted R-squared is calculated that represents the more accurate fit with multiple independent variables. The adjusted R-squared takes into account both the number of observations and the number of independent variables. It is always lower than R-squared.

F-statistics and P-value. The F-value, like the t-value (see t-value above), is calculated to measure the overall quality of the regression.

The F-value is the Mean Square Regression divided by the Mean Square Residual. It is calculated on 9 Df for the coefficients and 50 Df for the residuals.

The P-value associated with this F-value is very small (5.596e-15). The P-value is a measure of how confident you can be that the independent variables reliably predict the dependent variable. P stands for probability and is usually interpreted as the probability that test data does not represent accurately the population from which it is drawn. If the P-value is 0.10, there is a 10% probability that the calculation for the test data is not true for the population. Conversely, you can be 90% certain that the results of the test data are true of the population.

For example, if the P-value were greater than 0.05, the group of independent variables does not show a statistically significant relationship with the dependent variable, or that the group of independent variables does not reliably predict the dependent variable. Note that this is an overall significance test assessing whether the group of independent variables when used together reliably predict the dependent variable, and does not address the ability of any of the particular independent variables to predict the dependent variable. The ability of each individual independent variable to predict the dependent variable is addressed in the coefficients table. (See P-values for the regression coefficients.)
Reference: Analysis of Variance (ANOVA) From Linear Regression

The following image shows the Model tab with the ANOVA table for the regression output. The ANOVA table provides statistics on each variable used in the regression equation.

![Model tab with ANOVA table](image)

- **Analysis of Variance Table.** This is the title of this section.
- **Response: Price_1991.**

<table>
<thead>
<tr>
<th></th>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRAIN</td>
<td>1</td>
<td>2208522</td>
<td>2208522</td>
<td>44.7465</td>
<td>1.838e-08 ***</td>
</tr>
<tr>
<td>Degrees_in_C</td>
<td>1</td>
<td>4462834</td>
<td>4462834</td>
<td>90.4208</td>
<td>8.508e-13 ***</td>
</tr>
<tr>
<td>HRAIN</td>
<td>1</td>
<td>1251728</td>
<td>1251728</td>
<td>27.5417</td>
<td>3.445e-07 ***</td>
</tr>
<tr>
<td>Time_Since_Vintage</td>
<td>1</td>
<td>15331</td>
<td>15331</td>
<td>0.3373</td>
<td>0.564498</td>
</tr>
<tr>
<td>Chateau</td>
<td>5</td>
<td>1516501</td>
<td>303304</td>
<td>6.6697</td>
<td>0.0002934 ***</td>
</tr>
<tr>
<td>Residuals</td>
<td>50</td>
<td>2467813</td>
<td>49356</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Variable Names.** The first column shows the names of the variables used in the equation. It also lists the Error Term (Residuals).
Df. The degrees of freedom taken by each variable. The numeric independent variables take 1 degree of freedom each. The categorical variable has six unique values but since one is set as a reference for which no estimate is generated, the variable Chateau takes only 5 degrees of freedom. 50 degrees of freedom are left for the residuals. For example, 50=60-9-1.

Sum Sq and Mean Sq. The Sum of Squares and Mean Squares are computed so that you can compute the F statistics. (See next section on F-value.)

F-value and P-value. The F-value is computed by dividing the Mean Square by the Mean Square Residual. For example, the F-value for WRAIN is calculated by dividing the Mean Squares for the variable (2,208,522) by the Mean Square for the Residuals (49,356). The compute F-Value is 44.7465. The F-value is used to compute the P-value which indicates whether a variable is significant or not in the over model (for further explanation on the F-value and the P-value, see F-statistics and P-value).

Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

The significance codes indicate how certain we can be that the coefficient has an impact on the dependent variable. For example, a significance level of 0.001, indicates that there is less than a 0.1% chance that the coefficient might be equal to 0 and thus be insignificant. Stated differently, we can be 99% sure that it is significant. The significance codes (shown by asterisks) are intended for quickly ranking the significance of each variable.

Time taken: 0.53 secs.

The time taken to generate the parameter estimates.
Chapter 8

Building a Logistic Model

Logistic regression is a form of regression analysis. A binary logistic regression is used when the target variable is a dichotomy, having only two values, for example, 0 or 1, or Yes or No. The target variable is also referred to as a response variable, since it usually denotes the response to some event.

For example, you may be modeling responses to advertising campaigns or to medical treatments. In these cases, the logistic regression is used to predict which people are more likely to respond to an event, for instance, which people are likely to buy the advertised product. Multinomial logistic regression is used when the target variable has more than two classes.

In this chapter:

- Explanation of Logistic Regression

Explanation of Logistic Regression

What is Logistic Regression?

Logistic regression is used to predict outcomes or responses.

Logistic regression is conceptually similar to linear regression, where linear regression estimates the target variable. Instead of predicting values, as in the linear regression, logistic regression would estimate the odds of a certain event occurring. If predicting admissions to a school, for example, logistic regression estimates the odds of students being accepted in the school. The logistic regression algorithm takes the response (target variable) and transforms it into the odds of the event to occur.
Logistic regression has many analogies to linear regression: logit coefficients correspond to b coefficients, and a pseudo R2 statistic is available to summarize the strength of the relationship, for example, how much of the variation in the data is explained by the independent variables. See the explanation of R2 in Building a Linear Regression Model on page 225. The predictive success of the logistic regression can be assessed by looking at the error table. For details, see How to Evaluate the Logistic Regression on page 246, which shows the correct and incorrect classifications of the target variable. Goodness-of-fit tests, such as the likelihood ratio, indicate the appropriateness of the model and the Z statistics, analogously to the F statistics, which in linear regression indicates the significance of individual independent variables. See F-Statistics in Building a Linear Regression Model on page 225.

How Does Logistic Regression Work?

To see the similarities to linear regression, an example is provided using a data set that predicts admissions into graduate school based on two variables, GRE and GPA. A more complicated example includes categorical variables. The training set contains observations of individuals, and the response variable indicates whether the observed individual was admitted to graduate school. The following is the linear formula. In logistic regression, all coefficients represent odds. The odds ratio allows us to compare the probabilities between groups. For example, the odds of team A winning versus team B is 2:1. In the example below, the odds are calculated for the Binary event - admission versus rejection. The independent variables, GPA and GRE provide an explanation of the odds. Odds can be converted to probabilities as will be explained later in this chapter.

\[ Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \]

Similar to a linear regression equation, substitute the parameters with the estimated coefficients:

\[ Y = -4.949378 + 0.002691 \times GRE + 0.754687 \times GPA \]

For a student with GPA = 4 and GRE = 800, the estimated value of the target variable is 0.22217, which is used to compute the probabilities of the response being 1, as shown in the following formula. In WebFOCUS RStat, the scoring routine calculates the probabilities directly for you.

\[ P = 1 \div (1 + e^{\gamma(x)}) \]

where:

\[ e \] is the base of the natural logarithm, for example, 2.718.
Once applied, it appears that the student with GPA = 4 and GRE = 800 has a 0.56 probability of being admitted to graduate school.

**Practical Applications of Logistic Regression**

Logistic regression is used for:

- **Marketing applications.** Logistic regression is used to optimize direct mail. Response to direct mail is typically low, so marketers want to score prospects and select only those who are most likely to respond to the mailing campaign, thus minimizing the overall cost of the direct mail.

- **Law enforcement applications.** Logistic regression is used to predict the likelihood of particular crimes. Police districts are scored to determine whether aggravated assaults are likely to occur on a particular day.

- **Education.** Logistic regression is used to score students, in order to determine the likelihood of their dropping out of school.

- **Manpower management.** Logistic regression is used to score applications in the government, in the army, and so on, that predict attrition rates.

**Procedure: How to Create a Logistic Regression Model**

This example creates a logistic regression, using the sample college admissions data, and targets admit.

1. Define the Model Data.
   - Load the admissions data into RStat as described in *Getting Started With RStat* on page 33.
   - Select *Ident* for the ID variable.
   - Select *Target* for the admit variable.
   - Keep all the other variables as Input.
   - Click *Execute* to set up the Model Data.
The status bar confirms your data settings, as shown in the following image.

2. Select the Model tab to select the Model Data.
   - Select Regression as the Type of model.
   - Select Logistic as the Model Builder.
     **Note:** Logistic regression is the only available selection, since your target variable is binary.
   - Click Execute to run the Model Data.
The Model Data output appears in the bottom window of the Model tab, as shown in the following image.

For details about the regression output, see *Output From Binomial Logistic Regression* on page 244.
Reference: Output From Binomial Logistic Regression

The following image shows the Model tab with the binomial logistic regression output. For details about creating this output, see How to Create a Logistic Regression Model on page 241.

- **Summary of the Logistic Regression model (built using glm).** This is the title of the output. It also shows you the function used to build the model.

  Summary of the Logistic Regression model (built using glm):

- **R Function Call.** This section shows which model was used in R and which options for the model were specified. The generalized Linear Model and the logit link for the binary target variable were used. It also shows which columns from the data set are used in the model.

  \[
  \text{glm(formula = admit} \sim ., \text{family = binomial(link = "logit"), data = crs\$dataset[crs\$train,}
  \text{c(crs\$input, crs\$target)])}
  \]

- **Deviance Residuals.** This shows the distribution of the residuals for individual cases used in the model. See the explanation on this output in Building a Linear Regression Model on page 225.
Min       1Q   Median       3Q      Max
-1.2936  -0.8931  -0.7381   1.3225   1.8528

- Coefficients. This part of the output shows the coefficients, their standard errors, the z-statistic (sometimes called a Wald z-statistic), and associated P-values. The Z statistics are analogous to the F in the linear regression. For information about P-values, see Output From Linear Regression on page 233.

| Estimate | Std. Error | z value | Pr(>|z|) |
|----------|------------|---------|----------|
| (Intercept) | -3.618671  | 1.298793 | -2.786  0.00533 ** |
| gre       | 0.002584   | 0.001242 | 2.080  0.03750 * |
| topnotch  | 0.491049   | 0.347595 | 1.413  0.15774 |
| gpa       | 0.330209   | 0.391405 | 0.844  0.39886 |
| GenderMale| 0.254581   | 0.264021 | 0.964  0.33492 |

Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

- Evaluation of the Regression. The null deviance and the residual deviance are used to test whether the independent variables provide statistically significant explanation. A chi-square test, using the difference between the two residuals, indicates the overall significance of the model.

Null deviance: 351.65  on 279  degrees of freedom
Residual deviance: 339.63  on 275  degrees of freedom
AIC: 349.63
Number of Fisher Scoring iterations: 4
Log likelihood: -169.813 (5 df)
Null/Residual deviance difference: 12.021 (4 df)
Chi-square p-value: 0.00737176
Pseudo R-Square (optimistic): 0.20130800

- AIC. Akaike’s Information Criterion (AIC) is -2*log-likelihood+2*k where k is the number of estimated parameters. It is useful when comparing models.

- Fisher Scoring Iterations. This is the number of iterations to fit the model. The logistic regression uses an iterative maximum likelihood algorithm to fit the data. The Fisher method is the same as fitting a model by iteratively re-weighting the least squares. It indicates the optimal number of iterations. For example, beyond some number of iterations there are no practical gains. You can think of this as being analogous to the determination of the maximum number of nodes in Decision Trees. For more information, see Building a Decision Tree Model on page 265.
The following output shows the Model tab with the ANOVA table for the binomial logistic regression output. The ANOVA table provides statistics on each variable used in the regression equation. For details about creating this output, see "How to Create a Logistic Regression Model" on page 241.

- Analysis of Deviance Table
- Model: binomial, link: logit
- Response: admit
- Terms added sequentially (first to last)

| Df | Deviance | Resid. Df | Resid. Dev | P(>|Chi|) |
|----|----------|-----------|------------|---------|
| NULL | 279 | 351.65 | | |
| gre | 1 | 7.83 | 278 | 343.81 | 0.01 |
| topnotch | 1 | 2.41 | 277 | 341.40 | 0.12 |
| gpa | 1 | 0.84 | 276 | 340.56 | 0.36 |
| Gender | 1 | 0.93 | 275 | 339.63 | 0.33 |

Time taken: 0.16 secs
Generated by RStat 2009-03-05 12:37:53
======================================================================

The ANOVA table shows how the addition of independent variables reduces the Residual deviance (Resid. Dev). It is used to determine the significance of the independent variables. This is frequently used to determine whether to keep variables in the model.

Time taken: 0.17 secs
Generated by RStat 2008-07-26 18:58:57
======================================================================

Note: This is the same footer as the linear regression model.
b. Percent correctly classified versus incorrectly classified: In this case, 70% (66+4) of the cases are correctly classified. The proportion of correctly classified can be interpreted as a measure of how good the model is.

3. Select **Lift** as the evaluation type and click **Execute**.

Lift charts provide means to assess the performance of a scoring model. The purpose of a targeting model is to identify segments of the population with potentially higher response rates. A model is good if the targeted segment has a higher response rate than the average for the population as a whole. Lift is the ratio of positive responders in a segment to the positive responders in the population as a whole. For example, if a population has an average response rate of 10% and the targeted segment has a response rate of 50%, then the lift is 5 (50% divided by 10%). The lift chart then allows you to quickly determine what proportion of the population you want to target in order to maximize your return.
RStat displays the cumulative Lift curve. It is a downward-sloping curve. The population is sorted in a descending order from high responders to low responders. As you target larger segments of the population, your response rate will become closer to the average response rate of the population, and therefore your cumulative Lift will become closer to 1. At the moment when the cumulative curve converges to 1, there are no further gains from using the model. So imagine now that your lift curve converges to 1, for example, lift becomes 1.01, when the X-axis (rate of positive predictions, also known as hit) is 0.5. That would imply that by using the model you can achieve the same response rate by targeting half the population. If you were sending direct mail, you can realize 50% savings by using the model and sending mails to less people. Identifying all positives in a population (100%), implies targeting the entire population. So, there is a trade-off between how many positives you want to identify and cost. The lift chart allows you to determine the cut off points based on your cost constraints.

The chart in our example indicates that the biggest gains are realized by targeting up to 30% of the population. Beyond this point, the marginal cost of identifying one additional positive response increases. Stated differently, you have to target more and more people for each positive response rate.
4. Select ROC as the evaluation type and click *Execute*.

An ROC Curve (Receiver Operating Characteristic) provides a quick assessment of the quality of a model with respect to the true positive prediction, that is, the targets that have positive response and are classified by the model as positive responders. In every model, there is a trade off. As you target more and more of the population to identify true positives, you are also going to get false positives, that is, respondents who are negative but are classified by the model as positives. In other words, you have to decide how many false positives you are willing to tolerate for each true positive identified. For example, if you are conducting a direct mail and for each positive you have to accept one false positive in the mailing list, your cost will double. Thus, the ROC chart is especially useful to compare models and determine thresholds that yield high proportions of positive hits.

The ROC curve is usually plotted in reference to a baseline model. The base model is represented by the diagonal line starting at (0,0) and ending at (1,1). The area under the ROC Curve is the indicator of the quality of a model. The larger the area under the ROC curve, the higher the likelihood that an actual positive case will be assigned a higher probability of being positive than an actual negative case. For the Base Model, the area under the ROC curve is equal to 0.5. An area of 1 represents a perfect model. A curve with an area of 1 will have Y=1 and X=0, which indicates that all cases are classified correctly. Conversely, if Y=0 and X=1 the model identifies only false positives.

In our example, the ROC curve is close to the baseline indicating that for each positive you are getting approximately one false positive. This can be verified by looking at the error matrix. You can see that the model has identified 4 true positives and 3 false positives.

5. Select *Sensitivity* as the evaluation type and click *Execute*.

This plots sensitivity (the true positive rate) against the specificity (the true negative rate), as shown in the following image.
The Sensitivity versus Specificity chart is an alternative ROC curve, with Sensitivity being the true positive rate (the count of true positives, that is, cases with response = 1, divided by the count of positives) and Specificity being the true negative rate (the count of true negatives, that is, case with response = 0, divided by the count of negatives). In the case of a curve having x=1 and y = 1, the sensitivity correctly identifies both cases with an outcome of 1, and cases with an outcome of 0. It will have an Area of 1. Compared to the ROC curve, sensitivity is a right-sided curve. It enables you to judge whether the prediction is better on the positive or on the negative side of the response variable.
Building a Survival Model

You can use survival analysis to model time-to-event data. You can analyze and estimate how the probability of an event occurring either increases or decreases over time. For example, certain machines parts are more likely to go bad as the time in operation progresses. An engineer may use survival analysis to determine the optimal time to replace certain parts.

In this chapter:

- Explanation of Survival Analysis

Explanation of Survival Analysis

The Cox regression, also referred to as the proportional hazard model, is the most general of the regression models because it is not based on any assumptions concerning the nature or shape of the underlying survival distribution and the corresponding hazard function. The Cox regression predicts individual risk relative to the population.

The parametric survival regression, also referred to as the accelerated failure time model, assumes a particular distribution, such as Weibull, exponential, Gaussian, logistic, lognormal, and log-logistic. In RStat, the default distribution is set to Weibull. The parametric regression predicts the expected time to the event of interest.

What Is Survival Analysis?

A survival model is used to analyze time-to-event historical data and to generate estimates, referred to as survival curves, that show how the probability of the event occurring changes over time. In many life situations, as time progresses, certain events are more likely to occur. The survival models help decision makers to form better estimates than guessing about the expected timing of certain events. The estimates take into account the impact of other variables, referred to as independent, predictor variables or covariates, on the expected timing of the event to occur. A survival analysis can be used to determine not only the probability of failure of manufacturing equipment based on the hours of operations, but also to differentiate between different operating conditions. For example, if the probability changes if the machine is used outdoors versus indoors. For a detailed discussion on independent variables, see Building a Linear Regression Model on page 225.
Originally developed in the biomedical sciences to analyze time to death either of patients or of laboratory animals, survival analysis is now widely used in engineering, economics, finance, healthcare, marketing, and public policy.

**How Does Survival Analysis Work?**

Analogous to a linear regression analysis, a survival analysis typically examines the relationship of the survival variable (the time until the event) and the predictor variables (the covariates). The event of interest is frequently referred to as a hazard. The analysis specifies a linear-like function for the event called the hazard function. Below is the log hazard function, which is analogous to the linear regression function in *Building a Linear Regression Model* on page 225.

\[
\log h_1(t) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_n x_n
\]

where:
- \(t\) represents the time to the event for each observation.
- \(\alpha\) is the baseline hazard, for example, when all the \(x\) values equal zero.
- \(\beta\) are the estimated coefficients.
- \(x\) are the covariates.

Under this model, each covariate either increases or decreases the expected hazard, analogous to the predictor of linear regression. For more information on linear regression, see *Building a Linear Regression Model* on page 225.

There are several well known variations of the survival function, such as the multiplicative model or the proportional hazards model used in the Cox regression. While the math may differ, the conceptual understanding remains the same. Since the technical details of those functions are well documented in the statistical literature, we will omit them from the present discussion.

If survival analysis is so much like regression, why use a different modeling technique? Why not use linear regression?

Survival analysis methods are explicitly designed to deal with data about terminal events where some of the observations can experience the event and others may not. Such observations are called *censored observations*. For example, the target variable represents the time to a terminal event, and the duration of the study is limited in time. Therefore, some observations will not experience the event. Some equipment will fail during the time in which we monitor performance, but some will not.
There are two types of censoring, right and left censoring. Right censoring is most common and it occurs when the study expires or an individual or an item is removed from the study before the event occurs. For example, some individuals may still be alive at the end of a clinical trial, or may drop out of the study for various reasons other than death prior to the termination of the study. An observation is left-censored if its initial time at risk is unknown. This will occur if we do not know when a participant experienced for the first time the condition of interest. For example, when an individual contracted a disease.

The diagram below shows the distinct types of censoring.

Censoring complicates the estimation of the survival function. It requires different techniques than linear regression. Thus, in addition to the target variable, survival analysis requires a status variable that indicates for each observation whether the event has occurred or not and the censoring.

Practical Applications of Survival Analysis
Survival analysis is used for:

- **Engineering.** It is used to perform reliability or failure analysis on parts and components.
- **Warranty Applications.** It is used to estimate optimal time for part replacement versus part service.
- **Healthcare.** It is used to estimate the risk of disease progression in patients, the impact of medications and treatment algorithms, and the timing of medications and procedures.
- **Government and Social Services.** It is used in child welfare to match children with foster parents and to optimize the length of stay of children in the program, to estimate participation time in various social programs, and to estimate the time it takes for various policies to take effect.
- **Law Enforcement.** It is used to estimate the likelihood of recidivism.
- **Marketing Operations.** It is used to assess the length of participation in loyalty programs.

**Procedure:** How to Create a Survival Analysis Model Using Cox Regression

In this example, we are going to create a model for matching children with foster parents. We will use historical data that contains observations on the length of stay of children with foster parents. The predictor variables for the length of stay include child characteristics, such as age and gender, and foster parent characteristics, such as age and occupation.

1. **Define the Model Data.**
   - Load the child welfare training data set into RStat. For more information on loading data into RStat, see *Getting Started With RStat* on page 33.
   - Turn off sampling by clearing the Partition check box.
   - On the Target Data Type, select the **Survival** radio button.
     - Notice that **Target** is replaced with **Time**.
     - Notice that **Risk** is replaced by **Status**.
   - Select **TIME_IN_PROG** as the Time variable.
     - The time variable has to be a continuous numeric variable representing time.
   - Select **EXIT_PROG** as the status variable:
     - The status variable has to be a continuous variable representing the event of interest, for example the code number of the event.
The status variable must clearly define a terminal event, for example, a machine break, death, or exiting a social program. In this particular example, it indicates that a child has exited the program.

The status variable must have at least one value, for example, the terminal event.

For interval censored data, which is not used in this example, the status indicator must be:

- 1 = event at time
- 2 = left-censored
- 3 = interval-censored

Your screen should look as below.

2. Select the Model tab to select the Model Data.
   Survival is the only available method and Cox Proportional Hazards is set as the default choice.

3. Click Execute to run the model.
The model output appears in the bottom window of the Model tab, as shown in the following image.

**Reference:** Output From a Cox Proportional Hazard Regression

This section describes the Cox Proportional Hazard regression output.

**Summary of the Cox regression model.** This is the title of the summary provided for the model.

**Summary of the Survival model (built using coxph)**

**R Function Call.**

```r
coxph(formula = Surv(TIME_IN_PROG, EXIT_PROG) ~ RACE + CHILD_GENDER + OPN_AGE_YR + PARENT_AGE + EDUCATION + MARITAL + PARENT_GENDER + OCCUPATIONGROUP + ECONOMIC_STATUS + EMPLOYED_FOR_YRS + CURRENT_EMPLOYMENT_TERM, data = crs$dataset[, ])
```
where:

```r
coxph
```

Is the library in R that is used to construct the hazard function.

The formula lists the survival target and status variable (Surv(time, status)) and the covariates (age, education, marital, and so on). The data section shows which columns from the data sets are used in the model. In this case, we use all the columns from the data set.

```
n=615
```

Indicates the number of observations used for training the model.

Coefficients are shown in the following image.

![Coefficients for survival model](image)

This part of the output shows the coefficients, their standard errors, the z-statistic, and associated Pr(z)-values.
**Note:** To view additional data regarding the coefficients, use the down arrow to scroll through the output.

The closer the P values are to zero, the more significant the coefficient is. The z statistics records the ratio of each regression coefficient to its standard error. The Z statistics are analogous to the F in the linear regression, see *Output From Linear Regression* on page 233.

The second column displays the exponentiated coefficient. The exponent of the estimated coefficient is the odds ratio (hazard ratio) which can be used to estimate directly the percent of risk that each coefficient contributes to the overall hazard. For example, holding all other covariates constant, a child gender=female reduces the hazard of a child leaving the program by a factor of $e^{(\text{CHILD\_GENDERFemale})} = 0.943859$, or it will reduce the risk by $1 - 0.943859 = 0.056$, or by 5.6%.

The significance codes indicate the level of significance of each coefficient. (See *Output From Linear Regression* on page 233 for more information on significance codes.) For example, OCCUPATIONGROUPSales is significant as indicated by the ".". Its significance can also be inferred from the small Pr(z) value of 0.0492.

The following image shows the continued output for the coefficients table. It shows the exponentiated coefficient and the 95% confidence interval (upper and lower bound) for the exponentiated coefficient.

The exp(--coef) is the inverted hazard ratio. The exp(coef) shows the hazard ratio between female versus male. The inverted hazard ratio shows the male versus female odds ratio.
**Note:** The data for exp(coef) and exp(-coef) are in different sections. To view each, use the down arrow to scroll down through the output.

The image below shows the model diagnostics statistics.
**Rsquare.** Is a measure of the proportion of variability explained by the regression. It is a number between zero and one, and a value close to zero suggests a poor linear regression model. (See *Output From Linear Regression* on page 233 for more information.) Although the generalized R-squared is commonly recommended for the Cox model, it must be noted that it is highly sensitive to the proportion of censored values. The expected value of R-squared decreases substantially as a function of the percent censored observations. Average Rsquare values can decrease by 20% or more with heavy censoring.

**Likelihood ratio tests, the Wald test, and the Score tests.** All three tests are global tests evaluating the hypothesis that the coefficient estimates are different from zero, or in other words that they contribute to the prediction of the hazard. The closer the test result is to zero, the more likely it is that the coefficients are different from zero and thus do contribute to the prediction. The degrees of freedom (df) are equal to the number of parameters in the model minus the number of parameters of a reduced model, for example a model without the coefficients that are equal to zero. Usually the three tests give very similar results. Sometimes the Wald statistics may be different from the Likelihood ratio test for various reasons, and there are many reasons given in the statistical literature why the likelihood ratio test should be used instead of the Wald when the two tests disagree. Those theoretical discussions fall outside of the scope of the present manual. The score test is more useful when the survival distributions of two samples are compared. For example, in clinical trials the score test is used to establish the efficacy of a new treatment compared to a control treatment when the measurement is the time-to-event (such as the time from initial treatment to a heart attack).
The following image shows the tests for proportional hazard assumptions. The proportional hazard regression assumes that for any two groups or strata the survival curves are proportionate over time. The condition implies that the covariates multiply the hazard. The output is a matrix with one row for each variable and a last row for the global test. The columns of the matrix contain the correlation coefficient between transformed survival time and the scaled Schoenfeld residuals (see How to Generate Survival Analysis Plots on page 262), a chi-square, and the two-sided p-value. For the global test there is no appropriate correlation, so an NA is entered into the matrix as a placeholder. The closer the p-value is to zero, the stronger the evidence of the existence of a non-proportional hazard, a hazard that varies over time and thus impacts the effectiveness of the prediction. The presence of a non-proportional hazard requires some corrective actions, many of which are documented in the statistical literature but are outside of the scope of this manual.
**Procedure:**  How to Generate Survival Analysis Plots

To generate a plot, click the *Plot* button.

Two charts are generated, a Survival chart and a Scaled Schoenfeld Residuals chart. The Plot option is available only for a Cox regression.

**Survival Chart**

The Survival chart plots the probability of survival (Y) against Time (X). The two dashed line represent the 95% confidence interval.

![Survival Chart](image)

**Scaled Schoenfeld Residuals Chart**
Schoenfeld residuals are used to test the assumption of proportional hazards. Schoenfeld residuals "can essentially be thought of as the observed minus the expected values of the covariates at each failure time" (Steffensmeier & Jones, 2004: 121). There is a Schoenfeld residual for each subject for each covariate. The image below shows the SSR for the RACECAUCASIAN coefficient. The plot of Schoenfeld residuals against time for any covariate should not show a pattern of changing residuals for that covariate. If there is a pattern, that covariate is time-dependent. As a rule of thumb, a non-zero slope is an indication of a violation of the proportional hazard assumption. The dotted lines outline the 95% confidence interval.
Building a Decision Tree Model

You can use a decision tree for either classification or prediction. A decision tree is easier to understand by a decision maker than, for example, a neural network. The decision tree rules are presented as simple if/then statements.

Building a decision tree involves steps, such as splitting, pruning, and tree selection.

In this chapter:
- Explanation of the Decision Tree Model

Explanation of the Decision Tree Model

What Is a Decision Tree?

A decision tree is a machine learning algorithm that partitions the data into subsets. The partitioning process starts with a binary split and continues until no further splits can be made. Various branches of variable length are formed.

The goal of a decision tree is to encapsulate the training data in the smallest possible tree. The rationale for minimizing the tree size is the logical rule that the simplest possible explanation for a set of phenomena is preferred over other explanations. Also, small trees produce decisions faster than large trees, and they are much easier to look at and understand. There are various methods and techniques to control the depth, or prune, of the tree.

How Do Decision Trees Work?

There are several steps involved in the building of a decision tree.

Splitting. The process of partitioning the data set into subsets. Splits are formed on a particular variable and in a particular location. For each split, two determinations are made: the predictor variable used for the split, called the splitting variable, and the set of values for the predictor variable (which are split between the left child node and the right child node), called the split point. The split is based on a particular criterion, for example, Gini (for classification) or sums of squares (for regression) from the entire data set. The leaf node, also called a terminal node, contains a small subset of the observations. Splitting continues until a leaf node is constructed.
Pruning. The shortening of branches of the tree. Pruning is the process of reducing the size of the tree by turning some branch nodes into leaf nodes, and removing the leaf nodes under the original branch. Pruning is useful because classification trees may fit the training data well, but may do a poor job of classifying new values. Lower branches may be strongly affected by outliers. Pruning enables you to find the next largest tree and minimize the problem. A simpler tree often avoids over-fitting.

Tree Selection. The process of finding the smallest tree that fits the data. Usually this is the tree that yields the lowest cross-validated error. See the explanation for cross-validation error in Output From Decision Trees on page 271.

Decision trees:

- Are popular among non-statisticians as they produce a model that is very easy to interpret. Each leaf node is presented as an if/then rule. Cases that satisfy the if/then statement are placed in the node.

- Are non-parametric and therefore do not require normality assumptions of the data. Parametric models specify the form of the relationship between predictors and a response. An example is a linear relationship for regression. In many cases, however, the nature of the relationship is unknown. This is a case in which non-parametric models are useful.

- Can handle data of different types, including continuous, categorical, ordinal, and binary. Transformations of the data are not required.

- Can be useful for detecting important variables, interactions, and identifying outliers.

- Handle missing data by identifying surrogate splits in the modeling process. Surrogate splits are splits highly associated with the primary split. In other models, records with missing values are omitted by default.
Practical Applications of Decision Tree Analysis

Decision trees can be used either for classification, for example, to determine the category for an observation, or for prediction, for example, to estimate the numeric value. Using a decision tree for classification is an alternative methodology to logistic regression. Using a decision tree for prediction is an alternative method to linear regression. See those methods for additional industry examples.

- **Identify Target Groups.** If you are looking for the best potential customers for a product, you can identify the terminal nodes in the tree that have the highest percentage of sales, and focus your sales effort on individuals described by those nodes.

- **Predict Outcomes.** You may want to predict particular outcomes, for example, the occurrence of crime, fraud, clinical outcome, and so on. In all of these cases the outcome has a binary value, that is, it has either occurred or not. The decision tree will identify the rules that determine whether the event or outcome will occur or not.

- **Data Exploration and Pattern Detection.** Similar to graphs, decision trees for exploratory analysis of large data sets can help you detect and visualize relationships and patterns in much larger sets of variables. For example, if you are analyzing insurance claims, you may find that theft claims are more likely on foreclosed homes in higher income zip codes. The tree diagrams will clearly show you how claims are segmented across different variables. The larger the number of variables, the more valuable is the exploration using decision trees.

**Procedure: How to Create a Decision Tree Model**

The following example uses the credit scoring data set that was explained and used for the scoring application example in *Creating a Scoring Application* on page 101.

To execute the tree model:

1. Load the credit scoring data set into RStat.
   
   For more information on loading data into RStat, see *Getting Started With RStat* on page 33.

2. Use the default sample percentage of 70%.

3. Select the data roles as shown in the following image:
   
   a. *ID* as Ident.
   
   b. *CREDIT_APPROVAL* as Target.
c. All other variables as Input.

Note: This list has been truncated for display purposes.

4. Click Execute.

5. On the Model tab, select Decision Tree for the Type.

   Note: Do not change any of the default parameters. For more information on the default values, see User-Defined Parameters on page 270.

6. Click Execute.
The Summary of the Tree model for Classification appears, as shown in the following image.
Reference: User-Defined Parameters

User-defined parameters include:

- **Priors.** Sets the prior probabilities for each class. All probabilities must add up to 1.
  
  The default priors are proportional to the data counts. The input box is empty by default.

- **Min Split.** The minimum number of values in a node that must exist before a split is attempted. In other words, if the node has two members and the minimum split is set to 5, the node will become terminal, that is, no split will be attempted.
  
  The default value is 20.

  **Consideration:** As a rule, many programs and data miners will not attempt, or advise you, to split a node with less than 10 cases in it.

- **Max Depth.** Controls the maximum depth of the tree that will be created. It can also be described as the length of the longest path from the tree root to a leaf. The root node is considered to have a depth of 0. The Max Depth value cannot exceed 30 on a 32-bit machine.
  
  The default value is 30.

- **Loss Matrix.** Weighs the outcome classes differently.

- **Min Bucket.** The minimum number of entities allowed in any leaf of the tree. The default number is one-third of the value specified for Min Split.

- **Complexity.** Complexity is used to establish a control level that determines whether a split contributes to a better model fit. Any split that increases the model fit by a factor greater than the defined complexity factor is attempted. For instance, with regression splitting, this means that the overall R-square must increase by the defined complexity factor at each step. The main role of this parameter is to save computing time by pruning off splits that are not worthwhile. By specifying the complexity factor, the user informs the program that any split that does not improve the fit will likely be pruned off by cross-validation, and that the program need not pursue it. If you set the complexity factor to 0, the program will create the largest tree.
  
  The default value is 0.01.
Reference: Output From Decision Trees

The model output is described line by line. For illustration purposes, we have pruned the tree by lowering the Max Depth from the default to 3.

This section describes the decision tree output.

- **Summary of the Tree model for Classification (built using rpart)**. This is the title of the output for the decision tree. Rpart is the library in R that is used to construct the decision tree. Classification indicates that the modeling technique was applied to a set with a categorical dependent variable.

  Summary of the Tree model for Classification (built using rpart):

  - **n=1348**. Indicates the number of observations used in the model. The credit scoring data set contains 1926 observations. However, we sampled 70% of them to use as a training set.
Explanation of the Decision Tree Model

n= 1348

- **Print Specifications.** Indicates what information is printed in the output for each node:

  node), split, n, loss, yval, (yprob)
  * denotes terminal node

- **Nodes Information.** This output prints the tree in an extended form, that is, it describes exactly each node in the tree accordingly to the print specifications described in the previous bullet. Notice the indentation. It is used to indicate the tree topology, that is, it indicates the parent and child relationships (also referred to as primary and secondary splits). See the chart on the next page for the graphical representation of the parent-child relationships in the tree.

1) root 1348 204 0 (0.84866469 0.15133531)
   2) INCOME>=33270.53 1074  22 0 (0.97951583 0.02048417)
      4) INCOME>=44861.79 879   0 (1.00000000 0.00000000) *
      5) INCOME< 44861.79 195  22 0 (0.88717949 0.11282051)
         10) EDUCATION=Bachelor,College,HSgrad,Master,Professional 180 7 0 (0.96111111 0.03888889) *
         11) EDUCATION=Doctorate 15   0 1 (0.00000000 1.00000000) *
   3) INCOME< 33270.53 274  92 1 (0.33576642 0.66423358)
   6) AGE>=28.5 185  86 1 (0.46486486 0.53513514)
      12) INCOME>=28790.85 71   29 0 (0.59154930 0.40845070) *
      13) INCOME< 28790.85 114  44 1 (0.38596491 0.61403509) *
   7) AGE< 28.5  89   6 1 (0.06741573 0.93258427) *
**Node Numbering.** Nodes are labeled with unique numbers. Those numbers are generated by the following formula: the child nodes of node X are always numbered 2x (left child) and 2x +1(right child). The root node is 1. The following tree diagram generated by clicking the Draw button shows in color the node numbers for the tree described previously. Only the terminal node numbers are displayed. For example, node 2 and 3 labels are not shown. Node 2 (left child) is derived by multiplying node 1*2, node 3 (right child) by (1*2)+2. Node 4 (left child of node 2 is derived by 2*2. Terminal node 10 is derived from node 5 (right child of node 2) by 5*2. Terminal node 11 is the right child of node 5 derived by (5*2)+1.
**Primary Split.** Income is the predictor variable used for the primary split. The same predictor variable can be used to split many nodes. For example, node 2 is further split using Income. Age is the primary split for node 3.

**Split Point.** Nodes 2 and 3 were formed by splitting node 1 on the predictor variable Income. The split point is 33270.53. If the splitting variable is continuous (numeric), as in this split, the values going into the left and right child nodes will be shown as values less than or greater than some split point (33270.53 in this example). Node 2 consists of all rows with the value of Income greater than 33270.53, whereas node 3 consists of all rows with Income less than 33270.53.

**Number of Node Cases.** The number after the Split Point. For example, for node 2 the first number is 1074, which indicates the total number of rows in the data that belong to this node. For node 4 the number is 879, and for node 7 the number is 89.

**Expected Loss.** This is the total number of rows that will be misclassified if the predicted class for the node is applied to all rows. In the case of node 4, all cases are correctly classified and therefore the number is 0. In the case of node 7, out of the total 89 cases, 6 will be misclassified. This information can also be inferred from the Probability of the Winning Class (see the description that follows), for example, the probability of the winning class, which is indicated by the third number 1 or 0 is 93%. This number is a placeholder for the category in the target variable. Based on this probability, out of 89 cases, 83 will be classified correctly. Therefore, 6 cases will be misclassified.

**Predicted Class for Node.** This is the predicted class for the node. For example, for node 7, this will be 1. In the sample data, 1 indicated good credit risk, and 0 indicated bad credit risk. People were classified into either one of the two categories.

**Probability of Winning Class.** The numbers after the predicted class for the node, for example, for node 7, indicate the probabilities of each class and allow the user to see the probability of the winning class, that is, the factor that determines the final classification. In this particular case, the predicted class for node 7 is 1 and the probability is 0.89. For node 4, the winning class is 0 and the probability is 1.00.

**Terminal Nodes.** An asterisk (*) indicates a terminal node. As the preceding diagram shows, nodes 4, 7, 10, 11, 12, and 13 are terminal. Other input variables that were specified on the Data tab, for example, Gender, were omitted from the model. The algorithm has determined that they did not contribute to the predictive power of the model.

**Variables actually used in tree construction:** Age, Education, Income. Shows the variables that are actually used to construct the tree. If you look at the decision tree image and at the node descriptions, you will notice that splits have occurred on the variables Age, Education, Income.
**Root node error:** $204/1348 = 0.15134$. This is the error rate for a single node tree, that is, if the tree was pruned to node 1. It is useful when comparing different decision tree models.

**Complexity Table:** The complexity table provides information about all of the trees considered for the final model. It lists their complexity parameter, the number of splits, the resubstitution error rate, the cross-validated error rate, and the associated standard error. See the following for an explanation of the items in the complexity table.

<table>
<thead>
<tr>
<th>CP</th>
<th>nsplit</th>
<th>rel error</th>
<th>xerror</th>
<th>xstd</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.441176</td>
<td>0</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>2</td>
<td>0.036765</td>
<td>1</td>
<td>0.55882</td>
<td>0.56373</td>
</tr>
<tr>
<td>3</td>
<td>0.031863</td>
<td>3</td>
<td>0.48529</td>
<td>0.57353</td>
</tr>
<tr>
<td>4</td>
<td>0.010000</td>
<td>5</td>
<td>0.42157</td>
<td>0.52451</td>
</tr>
</tbody>
</table>

**Complexity Parameter.** The complexity parameter was explained in the section *User-Defined Parameters* on page 270.

**Number of Splits.** The number of splits for the tree. As the diagram shows for tree 4, we have 5 splits. You can count the number of splits shown on the diagram on the previous page.

**Resubstitution Error Rate (xstand).** The resubstitution rate is a measure of error. It is the proportion of original observations that were misclassified by various subsets of the original tree. The tree yielding the minimum resubstitution error rate in the present example is tree number 4. The resubstitution rate decreases as you go down the list of trees. The largest tree will always yield the lowest resubstitution error rate. However, choosing the tree with the lowest resubstitution rate is not the optimal choice, as this tree will have a bias. Large trees will put random variation in the predictions as they overfit outliers.

**Cross-Validated Error Rate (xerror).** Instead of selecting a tree based on the resubstitution error rate, X-fold cross-validation is used to obtain a cross-validated error rate, from which the optimal tree is selected. The X-fold cross-validation involves creating X-random subsets of the original data, setting one portion aside as a test set, constructing a tree for the remaining X-1 portions, and evaluating the tree using the test portion. This is repeated for all portions, and an estimate of the error is evaluated. Adding up the error across the X portions represents the cross-validated error rate. The tree yielding the lowest cross-validated error rate (xerror) is selected as the tree that best fits the data. In this case, this is tree number 4, which has 5 splits.

- **Default Cross Validation.** RStat uses by default a 10-fold cross-validation.

- **Standard Error (xstd).** This is the standard deviation of error across the cross-validation sets.
**Procedure: How to Evaluate a Decision Tree Model**

In this procedure, you will produce the error matrix to evaluate how many of the categories are correctly classified.

1. On the Data tab, clear the *Partition* check box.
2. On the Model tab, select the *Decision Tree* model and click *Execute*.
3. On the Evaluate tab, select *Error Matrix* for the Type, load the data set for the Data, then select *Execute*.

There are two matrices produced. The first one gives you the counts of correctly or incorrectly classified records. For example, out of 77 records with good credit, 70 were classified correctly and 7 were misclassified. Out of 501 records with bad credit, 473 were classified correctly and 28 were misclassified. The second table gives you the percent of correctly or incorrectly classified records. If we sum both the correct and incorrect classifications, we get 82+12=94 percent correctly classified cases. However, it is important to see whether the model better classifies the positive or negative cases, for example, whether it predicts more accurately the good credit versus the bad. Those assessments are made by the modeler. See the evaluation techniques and examples in *Building a Logistic Model* on page 239.

**Error matrix for the Tree model on ab_credit_training.csv [test]**

(counts):

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>473 7</td>
</tr>
<tr>
<td>1</td>
<td>28 70</td>
</tr>
</tbody>
</table>

**Error matrix for the Tree model on ab_credit_training.csv [test] (%)**:

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>82 1</td>
</tr>
<tr>
<td>1</td>
<td>5 12</td>
</tr>
</tbody>
</table>

Overall error: [1] 0.06055363

Generated by RStat 2009-03-04 19:55:49

======================================================================
This chapter discusses a typical and widely used example of association rule mining called Market Basket Analysis. You can use Market Basket Analysis to analyze transactions and evaluate what items are frequently purchased together. This kind of analysis can help to determine how to place items in stores and how to create product combinations or promotions.

**In this chapter:**

- Explanation of the Market Basket Model

---

**Explanation of the Market Basket Model**

**What Is Market Basket Analysis?**

Market Basket Analysis is a technique which identifies the strength of association between pairs of products purchased together and identify patterns of co-occurrence. A co-occurrence is when two or more things take place together.

Market Basket Analysis creates If-Then scenario rules, for example, if item A is purchased then item B is likely to be purchased. The rules are probabilistic in nature or, in other words, they are derived from the frequencies of co-occurrence in the observations. Frequency is the proportion of baskets that contain the items of interest. The rules can be used in pricing strategies, product placement, and various types of cross-selling strategies.

**How Market Basket Analysis Works**

In order to make it easier to understand, think of Market Basket Analysis in terms of shopping at a supermarket. Market Basket Analysis takes data at transaction level, which lists all items bought by a customer in a single purchase. The technique determines relationships of what products were purchased with which other product(s). These relationships are then used to build profiles containing If-Then rules of the items purchased.

The rules could be written as:

\[
\text{If } \{A\} \text{ Then } \{B\}
\]

The *If* part of the rule (the \{A\} above) is known as the antecedent and the *THEN* part of the rule is known as the consequent (the \{B\} above). The antecedent is the condition and the consequent is the result. The association rule has three measures that express the degree of confidence in the rule, Support, Confidence, and Lift.
For example, you are in a supermarket to buy milk. Based on the analysis, are you more likely to buy apples or cheese in the same transaction than somebody who did not buy milk?

In the following table (table 1), there are nine baskets containing varying combinations of milk, cheese, apples, and bananas.

<table>
<thead>
<tr>
<th>Basket</th>
<th>Product 1</th>
<th>Product 2</th>
<th>Product 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Milk</td>
<td>Cheese</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Milk</td>
<td>Apples</td>
<td>Cheese</td>
</tr>
<tr>
<td>3</td>
<td>Apples</td>
<td>Banana</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Milk</td>
<td>Cheese</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Apples</td>
<td>Banana</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Milk</td>
<td>Cheese</td>
<td>Banana</td>
</tr>
<tr>
<td>7</td>
<td>Milk</td>
<td>Cheese</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Cheese</td>
<td>Banana</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Cheese</td>
<td>Milk</td>
<td></td>
</tr>
</tbody>
</table>

The next step is to determine the relationships and the rules. For explanation purposes, the following table shows some of the relationships. In total there are 22 rules for the nine baskets. The complete set of rules are shown in the explanation of the RStat output.

<table>
<thead>
<tr>
<th>Basket</th>
<th>How many Baskets Containing The product</th>
<th>Total # Baskets</th>
<th>Support</th>
<th>Confidence</th>
<th>Lift</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Milk</td>
<td>6</td>
<td>9</td>
<td>0.666666667</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Cheese</td>
<td>7</td>
<td>9</td>
<td>0.777777778</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>Milk</td>
<td>6</td>
<td>9</td>
<td>0.666666667</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Cheese</td>
<td>7</td>
<td>9</td>
<td>0.777777778</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>Milk</td>
<td>6</td>
<td>9</td>
<td>0.666666667</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Cheese</td>
<td>7</td>
<td>9</td>
<td>0.777777778</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>(Apples, Milk)</td>
<td>1</td>
<td>9</td>
<td>0.111111111</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>&gt;&gt; Cheese</td>
<td>1</td>
<td>9</td>
<td>0.111111111</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>(Apples, Cheese)</td>
<td>1</td>
<td>9</td>
<td>0.111111111</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>&gt;&gt; Milk</td>
<td>1</td>
<td>9</td>
<td>0.111111111</td>
<td>1</td>
</tr>
</tbody>
</table>

The first measure called the support is the number of transactions that include items in the \{A\} and \{B\} parts of the rule as a percentage of the total number of transactions. It is a measure of how frequently the collection of items occur together as a percentage of all transactions.
The support formula written out would look something like:

\[
\text{Support} = \frac{(A + B)}{\text{Total}}
\]

Support for Basket 1 = \(\frac{(\text{Milk} + \text{Cheese})}{\text{Total}} = \frac{6}{9} = .6666667\)

Interpreted as: Fraction of transactions that contain both A and B.

The second measure called the confidence of the rule is the ratio of the number of transactions that include all items in \(\{B\}\) as well as the number of transactions that include all items in \(\{A\}\) to the number of transactions that include all items in \(\{A\}\).

The confidence formula written out would like something like:

\[
\text{Confidence} = \frac{(A + B)}{A}
\]

Confidence for Basket 1 = \(\frac{(\text{Milk} + \text{Cheese})}{\text{Milk}} = \frac{6}{6} = 1.000\)

Interpreted as: How often items in B appear in transactions that contain A only.

The third measure called the lift or lift ratio is the ratio of confidence to expected confidence. Expected confidence is the confidence divided by the frequency of B. The Lift tells us how much better a rule is at predicting the result than just assuming the result in the first place. Greater lift values indicate stronger associations.

The lift formula written out would look something like:

\[
\text{Lift} = \left( \frac{(A + B)}{A} \right) \left( \frac{B}{\text{Total}} \right)
\]

Lift for Basket 1 = \(\left( \frac{(\text{Milk} + \text{Cheese})}{\text{Milk}} \right) \left( \frac{\frac{6}{9}}{\frac{7}{9}} \right) = \left( \frac{1}{.7777778} \right) = 1.2857\)

Interpreted as: How much our confidence has increased that B will be purchased given that A was purchased.
Practical Applications of Market Basket Analysis

When one hears Market Basket Analysis, one thinks of shopping carts and supermarket shoppers. It is important to realize that there are many other areas in which Market Basket Analysis can be applied. An example of Market Basket Analysis for a majority of Internet users is a list of potentially interesting products for Amazon. Amazon informs the customer that people who bought the item being purchased by them, also reviewed or bought another list of items. A list of applications of Market Basket Analysis in various industries is listed below:

- **Retail.** In Retail, Market Basket Analysis can help determine what items are purchased together, purchased sequentially, and purchased by season. This can assist retailers to determine product placement and promotion optimization (for instance, combining product incentives). Does it make sense to sell soda and chips or soda and crackers?

- **Telecommunications.** In Telecommunications, where high churn rates continue to be a growing concern, Market Basket Analysis can be used to determine what services are being utilized and what packages customers are purchasing. They can use that knowledge to direct marketing efforts at customers who are more likely to follow the same path.

  For instance, Telecommunications these days is also offering TV and Internet. Creating bundles for purchases can be determined from an analysis of what customers purchase, thereby giving the company an idea of how to price the bundles. This analysis might also lead to determining the capacity requirements.

- **Banks.** In Financial (banking for instance), Market Basket Analysis can be used to analyze credit card purchases of customers to build profiles for fraud detection purposes and cross-selling opportunities.

- **Insurance.** In Insurance, Market Basket Analysis can be used to build profiles to detect medical insurance claim fraud. By building profiles of claims, you are able to then use the profiles to determine if more than 1 claim belongs to a particular claimee within a specified period of time.

- **Medical.** In Healthcare or Medical, Market Basket Analysis can be used for comorbid conditions and symptom analysis, with which a profile of illness can be better identified. It can also be used to reveal biologically relevant associations between different genes or between environmental effects and gene expression.

**Data Requirement**

1. **Baskets**

   - This column identifies the individual baskets.

   - Values can be categoric or numeric to identify the baskets.
2. Products

- This column has all the items that are included in each basket.
- Values of items can be categoric or numeric.

For example, from the table 1 below:

<table>
<thead>
<tr>
<th>Basket</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Milk</td>
</tr>
<tr>
<td>1</td>
<td>Cheese</td>
</tr>
<tr>
<td>2</td>
<td>Milk</td>
</tr>
<tr>
<td>2</td>
<td>Apples</td>
</tr>
<tr>
<td>2</td>
<td>Cheese</td>
</tr>
</tbody>
</table>

**Procedure: How to Create an Association Model Using Market Basket Analysis**

In this example, we are going to create a model for Market Basket Analysis of purchases at a grocery store. We will use the Basket data set that contains observations on the purchases of particular items, such as milk, cheese, and apples.

1. Define the Model Data.

- Load the Baskets data set into RStat. For more information on loading data into RStat, see *Getting Started With RStat* on page 33.
- Turn off sampling by unchecking the Partition check box.
- For the Target Data Type, leave the Auto radio button selected.
- Select *BASKET* as the Ident variable, which defines the basket.
- Select *PRODUCT* as the Target variable, which defines the products in the basket.
- Click Execute to run the Model Data.
The Status bar confirms your data settings, as shown in the following image.

2. Select the Associate tab, as shown in the following image.

- Select the Baskets check box.
Leave the default values for Support and Confidence. Changing Support and Confidence control values will increase or reduce the number of rules that get created.

**Note:** Support is a numeric value for the minimal support of an item set (the default value is 0.1). Confidence is a numeric value for the minimal confidence of the rules or association hyperedges (the default value is 0.1).

Click *Execute* to run the Model Data.
The model output appears. You may need to scroll to see the complete output, depending on the size of your window.
Reference: Output From the Market Basket Analysis

- **Summary of the Apriori Association Rules.** This is the title of the output. Apriori is the best known algorithm to mine association rules. Apriori iteratively discovers pairs with the largest frequencies and then with decreasing frequencies.

- **Number of Rules: 80.** The number indicates how many rules are generated from the data with the parameters selected.

- **Summary of the Measures of Interestingness.** This is a summary of the descriptive statistics of the distribution values for Support, Confidence, and Lift.

- **Summary of the execution of the apriori commands.**
  
  This is a summary of the settings that come with the apriori algorithm. Except for Support and Confidence, which you can change in the GUI, the remaining settings are set to default values.

- The mining parameters (parameter) change the characteristics of the mined item sets or rules (for example, the minimum support).

  parameter specification:
  
  confidence minival smax arem aval originalSupport support minlen maxlen target ext
  .01 0.1 1 none FALSE TRUE 0.1 1 5
  rules FALSE

- Control parameters (control) influence the performance of the algorithm (for example, enable or disable initial sorting of the items with respect to their frequency).

  algorithmic control:
  
  filter tree heap memopt load sort verbose
  0.1 TRUE TRUE FALSE TRUE 2 TRUE

Note: For more information on the apriori algorithm parameters, see the R documentation for the arules package at: [http://cran.r-project.org/web/packages/arules/arules.pdf](http://cran.r-project.org/web/packages/arules/arules.pdf).
**Procedure:** How to Generate Rules

To generate rules, click the *Show Rules* button.

The output will be printed below the original information presented.

- The LHS is the Antecedent ({A} from the example above).
- The RHS is the Consequent ({B} from the example above).

**Note:** Based on the data, the rules are created. For rule 1:

- Support says that 67% of customers purchased milk and cheese.
- Confidence is that 100% of the customers that bought milk also bought cheese.
- Lift represents the 28% increase in expectation that someone will buy cheese, when we know that they bought milk. This is the conditional probability.
**Procedure: How to Generate a Frequency Plot**

To generate a frequency plot, click the *Freq Plot* button. The output appears in a new window. This frequency plot shows the percent of times each unique item occurs in all baskets.

The bar chart below shows the frequency of the individual items in the analysis.

![Frequency Plot Example](image)

**Using a Market Basket Analysis Routine for Scoring**

In the example that is used in this chapter, the data set contains products that a customer in a grocery store might purchase (for example, milk, cheese, bananas, and apples). To run the Market Basket Analysis, the data set only needs to contain the basket and the product information. Once the Market Basket technique is run in RStat, a scoring routine can be exported, which would apply the output (rules with regard to the products and the confidence number) to the new data sets. This section provides procedures for the post-Market Basket Analysis execution process.
Procedure: How to Execute the Market Basket Analysis

To execute the Market Basket Analysis:

1. Load the Baskets data set into RStat. For more information on loading data into RStat, see Getting Started With RStat on page 33.
2. Disable sampling by clearing the Partition check box.
3. For the Target Data Type, leave the Auto radio button selected.
4. Select NUMBER as the Ident variable.
5. Select X_IF_PRODUCT as the Target variable.
6. Ignore all of the remaining variables.
7. Click Execute.
8. Click the Associate tab.
9. Select the Baskets check box.
10. Click Execute.

Procedure: How to Export the Market Basket Analysis Function

To export the Market Basket Analysis:

1. Click Export.
The Export C or PMML dialog box opens, as shown in the following image.

There are two export types that can be selected:

- **Item.** Exports rules that will determine the products in the new data set.
- **Confidence.** Exports the confidence number for the products that are selected in the result.

2. Select *Item* or *Confidence* as the export type, depending on your requirements.
3. Create a scoring file to run the Market Basket Analysis rules against.
This will show the format and file structure for the scoring data set. The Max Inputs for the scoring file should be the total items in the training data set minus 1.

a. Create a test file, as shown in the following image.

b. Using the Upload Data option in App Studio, create a Master File, as shown in the following image.

Note: The Upload Data option allows you to upload new data to an application, creating a new, unique Master File. Define or create a data source, right-click a folder from the relevant application folder, and then click Upload Data. The Reporting Server Console opens and you then select a file to upload. The Business View and Prepared result display. On the ribbon, click Load and Next, set your load options, and then click Proceed to Load to create the Master File in your repository. For more information on uploading data, see the Business Intelligence Portal manual.
c. Deploy the C files.
d. Test the C routines.

The following image shows the output that is generated for the sample WebFOCUS report in a web browser.

The above report output lists the items and confidence value for each item to be selected. Values in the ITEM_2 and ITEM_3 columns are inputs. Values in the item and confidence columns are the results of the Market Basket Analysis routine. In other words, item is the product recommendation that the customer is most likely to buy after buying item 2 and item 3 together according to the associated rules generated by the historical data.

In the first case, item 2 is empty, so the suggested item is Milk for people who only purchase Cheese. In the second case, Beer is not within the historical data for generating the rules, so No match found is returned. This means that there is no product recommendation for people who purchase Beer. The result of the third case indicates that people who purchase Milk and Apples will also purchase Cheese. This is followed by a confidence value that shows the possibility of buying Cheese after purchasing Milk and Apples together.
Explanation of the Market Basket Model
Appendix A

Deploying RStat Scoring Routines on WebFOCUS Reporting Servers Prior to Release 7.6.8

RStat scoring routines have been developed to run on WebFOCUS Version 7 Release 6.8 and higher Reporting Servers. Special features including enhanced handling of missing values, support for long procedure names, and support for enhanced application development have been added to the server to enhance development and deployment.

Note: This appendix describes how to deploy RStat scoring routines prior to Version 7.6.8.

In this appendix:

- Deploying RStat Scoring Routines Prior to Version 7.6.8

Deploying RStat Scoring Routines Prior to Version 7.6.8

Developer Studio Version 7 Release 6.8 is the first release with RStat support and the first release of RStat which generated usable 'C' code for deploying a model in FOCUS. In Version 7 Release 6.9, a better integration of MISSING DATA was implemented. This integration was done on both the server and Developer Studio sides. The Version 7.6.9 server also marked a change in the search strategy for finding FUSELIB functions on the server side, on the APPPATH. As deployed models are simply FOCUS user library (also known as FUSELIB) routines, they can be deployed to any server. Version 7.6.9 marks changes in the server for the handling of MISSING DATA, changes in the methods of compilation of 'C' code on the server regarding the destination paths of the generated DLLs, and the use of APPPATH to find FUSELIB files.

RStat scoring routines can be deployed and used on Reporting Servers prior to Version 7.6.8 with minor modifications and considerations:

- Flagging Scoring Routines

  In order for the scoring routines to operate successfully on Reporting Servers prior to Version 7.6.8, you will need to add the following line of code to the beginning of the scoring routine before compilation.

  ```
  #define PRE768_SERVER
  ```
## Missing Values

Special functionality has been added to the Reporting Server and the scoring routines to allow the routines to receive and return missing values. The routines recognize missing data within the data set, handle the processing of the missing values according to accepted standards for each model, and where appropriate, return a missing value in the score.

Scoring routines deployed on servers prior to Version 7.6.8 do not have access to this new functionality. All processing of missing values must be handled in the data preparation process prior to executing the scoring routine.

To ensure that routines do not return incorrect answers for records with missing values, use the MISSING ON NEEDS ALL command within the scoring COMPUTE or DEFINE. This command causes the COMPUTE or DEFINE to execute the scoring routine only when all input variables contain non-missing values. If any of the input parameters contain missing values, the returned value will be set to missing.

## Routine Name Length

There is no longer a procedure name limit of eight characters for Reporting Servers in Version 7.6.8 and higher. Procedures can have long names limited only by the rules of the specific deployment environment. This supports the development of more meaningful routine names that reflect the models and business scenarios they support.

In deployments prior to Version 7.6.8, procedure names cannot exceed a maximum of 8 characters in length excluding the file extension. For example,

**MODEL123.c**

Is a valid procedure name.

**MODEL123_TREE.C**

Is not a valid procedure name for deployment prior to Version 7.6.8.

**Note:** You cannot rename an existing scoring routine because the name is also embedded within the routine. The name needs to be defined at generation time.

## Remove Long Strings

In some environments, the default compiler functionality does not accept the long text strings used to embed the model PMML or model metadata within the routine. We recommend for deployment prior to Version 7.6.8, you exclude these strings from the routines when they are built.
To exclude the long strings from the routine when creating the scoring routine, turn off the Include PMML and Include Meta data options in the Export dialog box before saving. For more information on exporting the final model to build the scoring application, see Creating a Scoring Application on page 101.

## Compiling and Deploying the Scoring Routines

Scoring routines prior to Version 7.6.8 can be compiled using GENCPGM to create the load module (.dll). For further detail, see the Compiling and Storing a Subroutine topic in the Using Functions manual.

Once the compile is completed successfully, copy the resulting .dll to the location of your dynamic link functions library file as specified by the IBICPG environment variable. If no specific location is defined, the location defaults to:

```
..\ibi\srvXX\wfs\user
```

Once you have deployed the scoring routine using these rules, the scoring routine will be available for use with any DEFINE or COMPUTE commands within your WebFOCUS applications and reports in the same way as standard routines.
This is a glossary of key terms and concepts in this manual as they relate to RStat.

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>advanced regression</td>
<td>A functionality that enables the performance of more advanced regression techniques. Depending on the underlying data set, the types of advanced regression options available may include Normal (Linear), Gaussian, Poisson, Logistic, Multinomial, Binomial, Negative Binomial, Gamma, and Inverse Gaussian.</td>
</tr>
<tr>
<td>algorithm</td>
<td>A procedure for solving a mathematical problem (such as finding the greatest common divisor) in a finite number of steps, that frequently involves repetition of an operation. In RStat, algorithms form the basis of the model builder.</td>
</tr>
<tr>
<td>association rule mining</td>
<td>A popular technique for determining variable commonalities and associations when working with large databases. Association rule mining can also be used as a classification method.</td>
</tr>
<tr>
<td>Auto Target Data Type</td>
<td>The default data type.</td>
</tr>
<tr>
<td>BiCluster</td>
<td>Biclustering, which is also known as co-clustering or two-mode clustering, is a technique that is employed during data mining. It allows the rows and columns of a matrix to be clustered simultaneously.</td>
</tr>
<tr>
<td>binomial distribution</td>
<td>A discrete probability distribution of the number of successes in a sequence of a variable number (n) of dependent Yes or No experiments.</td>
</tr>
<tr>
<td>Boost Model</td>
<td>Uses the Ada (Adaptive) Boost Model, which generates and calls the classifiers a series of rounds to achieve better classification.</td>
</tr>
<tr>
<td>Term</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>C</strong></td>
<td>A general purpose programming language. C is used by WebFOCUS as a means for creating functions to be used in data analysis.</td>
</tr>
<tr>
<td><strong>Categoric Data Type</strong></td>
<td>Data organized into groups of categories. Categoric Data Type could be any character data or any numeric data with 10 or fewer unique values.</td>
</tr>
<tr>
<td><strong>class</strong></td>
<td>A component of multinomial regression.</td>
</tr>
<tr>
<td><strong>clustering</strong></td>
<td>A method of organizing objects into distinct groups or clusters based on their similarities.</td>
</tr>
<tr>
<td><strong>coefficient</strong></td>
<td>A constant that represents the rate of change in the dependent variable as a function of changes in the independent variable. It is the slope of the linear line, for example, it shows how prices go up with the increase in the number of vintage years, or how wines become more expensive the longer they mature. For other data sets, the trend can be the inverse, that is, the slope can be decreasing.</td>
</tr>
<tr>
<td><strong>Comma-Separated Value (CSV)</strong></td>
<td>A common file format that enables plain text storage of data (typically tabular data values separated by commas).</td>
</tr>
<tr>
<td><strong>correlation</strong></td>
<td>A measure of relation between two or more variables.</td>
</tr>
<tr>
<td><strong>correlation analysis</strong></td>
<td>Determines if there is a linear relationship between two variables. It also measures the strength and direction of the relationship. Correlation analysis does not test whether two samples are different.</td>
</tr>
<tr>
<td><strong>correlation coefficient</strong></td>
<td>A measure that determines the degree to which the movements of two variables are associated. A correlation coefficient closer to 0 indicates no relationship, while a correlation coefficient closer to 1 indicates a strong relationship.</td>
</tr>
<tr>
<td><strong>Cox Proportional Hazards</strong></td>
<td>A general regression model that predicts individual risk relative to the population.</td>
</tr>
<tr>
<td><strong>cross tabulation</strong></td>
<td>A method used to summarize data that is grouped into categories. This process creates contingency tables, which illustrate the summarized categorical data.</td>
</tr>
<tr>
<td><strong>data extract</strong></td>
<td>An extraction of data based on a set of parameters.</td>
</tr>
<tr>
<td><strong>data frame</strong></td>
<td>Collections of individual observations (rows of data) across many variables (fields).</td>
</tr>
<tr>
<td><strong>data set</strong></td>
<td>The underlying data used for modeling.</td>
</tr>
<tr>
<td><strong>data type</strong></td>
<td>Determines the type of modeling available and the specific algorithms that will be used within the modeling process. It is defined based on the type of data RStat identifies and the quantity of unique values found in the actual data.</td>
</tr>
<tr>
<td><strong>decision stump</strong></td>
<td>A decision tree with only one split.</td>
</tr>
<tr>
<td><strong>decision tree</strong></td>
<td>A predictive data mining tool that predicts a categorical or continuous response using a tree-like model.</td>
</tr>
<tr>
<td><strong>Decision Tree Model</strong></td>
<td>The prototypical data mining technique and default model in RStat. The Decision Tree Model is widely used because of its ease of interpretation.</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>dendogram</strong></td>
<td>A tree diagram that illustrates the arrangement or categorization of clusters.</td>
</tr>
<tr>
<td><strong>dependent (target) variable</strong></td>
<td>In experimental conditions, the dependent variable is a value whose measure is driven by some independent, manipulated variable. This variable is typically the target variable in regression analysis.</td>
</tr>
<tr>
<td><strong>DLL</strong></td>
<td>Dynamic-Link Library.</td>
</tr>
<tr>
<td><strong>end node</strong></td>
<td>The final or end node on a FancyPlots diagram, from which no new branches derive.</td>
</tr>
<tr>
<td><strong>estimation (predictive modeling)</strong></td>
<td>One of two types of statistical inferences supported by RStat, estimation is the process of deriving expected and predicted values from observations. Decision trees, regression, and other models are used to generate estimates. For example, the user can estimate whether a prospect is a good target for a particular marketing campaign, or the expected sales revenues for different stores in order to determine whether store layout and product mix has an impact on sales.</td>
</tr>
<tr>
<td><strong>Ewkm (Entropy Weighted KMeans)</strong></td>
<td>A clustering algorithm used to cluster high-dimensional data. It outputs the weight of each variable in each cluster.</td>
</tr>
<tr>
<td><strong>FancyPlots</strong></td>
<td>A technique within R to interactively and graphically represent decision trees, whereby users are able to prune the trees and output the results in C. FancyPlots is an advanced plot function for a decision tree.</td>
</tr>
<tr>
<td><strong>FEX</strong></td>
<td>A WebFOCUS executable report procedure (FOCUS executable, also known as FOCEEXEC).</td>
</tr>
<tr>
<td><strong>F-test</strong></td>
<td>Used to determine if the standard deviations of two samples are the same. If the standard deviations are not the same, then the bell-shaped curves will be different for the two samples. If the samples have the same standard deviations, then a T-test can be conducted to test if the means are equal. The test is also referred to as a test on the variance of two samples and is used in analysis of variance (ANOVA).</td>
</tr>
<tr>
<td><strong>Generalized Regression</strong></td>
<td>Regression technique that generalizes standard linear regression, allowing for response variables that fall outside of a normal distribution.</td>
</tr>
<tr>
<td><strong>GLM (Generalized Linear Models)</strong></td>
<td>Specific to regression.</td>
</tr>
<tr>
<td><strong>GUI</strong></td>
<td>Graphical User Interface.</td>
</tr>
<tr>
<td><strong>hidden layer nodes</strong></td>
<td>The number of hidden layers to display in a Neural Network (NNet) model. In RStat, NNet is, by default, a single layer neural net model, with the option to display the hidden layer.</td>
</tr>
</tbody>
</table>
**hierarchical clustering**

A method of cluster analysis that builds a hierarchy of clusters. Using an agglomerative, or bottom up, approach, this method is based on the pairing of subsequent clusters to originating clusters, which are based on a single observation.

**hypothesis testing**

Provides the user with the ability to use samples to test whether or not the null hypotheses are likely to be true.

**KMeans**

Is a partitioning method, used in clustering, that is best suited for large amounts of data. It creates a group of mutually exclusive, unique clusters, and then returns an index of those clusters. It then presents the means, or averages, of those clusters. The value of k represents each unique cluster.

**Kendall Correlation Coefficient**

Non-parametric measure of the association of two variables. It is a method for computing the correlation coefficient, that is, it does not assume that the data is normally distributed. The Kendall Correlation Coefficient is used to measure correlation between rankings and cross-tabulations.

**Kernel Value**

Also known as a Kernel Function, the Kernel Value is a parameter on the Model Tab (specific to SVM) that is used in training and prediction. It can be set to any function of class kernel. The options are Radial Basis (rbfdot), Polynomial (polydot), Linear (vanilladot), Hyperbolic Tangent (tanhdot), Laplacian (laplacedot), Bessel (besseldot), ANOVA RBF (anovadot), and Spline (splinedot).

**Kolmogorov-Smirnov**

A non-parametric test for quantifying the distance of continuous, one-dimensional probability distributions. Kolmogorov-Smirnov can be a one sample test, which compares a sample with a reference probability distribution, or a two sample test, which compares the relationship between two empirical distribution functions.
<table>
<thead>
<tr>
<th>Glossary Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>leaf node</td>
<td>Also known as the terminal node, the leaf node contains a small set of observations. In Decision Tree modeling, branch nodes are typically pruned into leaf nodes.</td>
</tr>
<tr>
<td>linear regression</td>
<td>A regression technique used to model a linear numeric outcome.</td>
</tr>
<tr>
<td>logarithm</td>
<td>The power (exponent) to which a base number must be raised in order to get the original number.</td>
</tr>
<tr>
<td>logistic regression</td>
<td>A form of regression analysis used to model binary outcomes.</td>
</tr>
<tr>
<td>modeling</td>
<td>The ability to model different scenarios and develop models that can be deployed as scoring applications. Modeling is part of the process of developing predictive outcomes.</td>
</tr>
<tr>
<td>multinomial regression</td>
<td>A regression technique that generalizes logistic regression, in that it allows for more than two discrete outcomes.</td>
</tr>
<tr>
<td>negative binomial</td>
<td>A discrete distribution of the number of successes in a sequence of Bernoulli trials before a specified number of failures occur.</td>
</tr>
<tr>
<td>distribution</td>
<td>Neural Network (NNet) model</td>
</tr>
<tr>
<td>model</td>
<td>Uses a structure that resembles the neural network of a human being. When applied to modeling, the concept is to build a network of connections that are connected by nodes. Once in place, the network propagates numbers.</td>
</tr>
<tr>
<td>node</td>
<td>Represents a branch or leaf in a Decision tree.</td>
</tr>
<tr>
<td>Glossary</td>
<td>Definition</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
</tr>
<tr>
<td><strong>non-parametric test</strong></td>
<td>A non-parametric test makes no assumption of the underlying distributions. For example, some of the data does not follow the normal distribution, such as ranked and cross-tabulated data. (Kolmogorov-Smirnov and Wilcoxon Rank-Sum fall into this category.)</td>
</tr>
<tr>
<td><strong>Numeric Data Type</strong></td>
<td>Any numeric data with more than 10 unique values.</td>
</tr>
<tr>
<td><strong>outliers</strong></td>
<td>Observations in a regression model that do not closely fit the line in any statistical plot. Outliers deviate from the majority of the data.</td>
</tr>
<tr>
<td><strong>paired difference tests</strong></td>
<td>Location test used to assess whether population means differ between two sets of measurements. (Specific types of T-tests and the Wilcoxon Signed Rank fall into this category.)</td>
</tr>
<tr>
<td><strong>parametric tests</strong></td>
<td>Hypothesis tests that make strong assumptions that the underlying data belongs to a certain type of distribution, which is defined by several parameters. (T-tests and F-tests fall into this category.)</td>
</tr>
<tr>
<td><strong>Pearson Correlation Coefficient</strong></td>
<td>The most commonly used method for the computation of the correlation coefficient. Measures the degree and strength of a linear relationship between two variables with bivariate normal distributions on a scale of -1 to 1.</td>
</tr>
<tr>
<td><strong>plot</strong></td>
<td>A graphical representation of data, such as a scatter plot.</td>
</tr>
<tr>
<td><strong>Poisson</strong></td>
<td>Regression technique used to model and predict in cases where count data and contingency tables are used.</td>
</tr>
<tr>
<td><strong>Predictive Model Markup Language (PMML)</strong></td>
<td>An XML-based markup language developed by the Data Mining Group to provide a way for applications to define models related to predictive analytics and data mining, and to share those models between the PMML-compliant applications.</td>
</tr>
<tr>
<td><strong>pruning</strong></td>
<td>The process of reducing the size of the decision tree by some branch nodes into leaf nodes, and removing the leaf nodes under the original branch.</td>
</tr>
<tr>
<td><strong>quartiles</strong></td>
<td>A variable value that divides the distribution of that variable into four groups with equal frequency.</td>
</tr>
<tr>
<td><strong>R</strong></td>
<td>A powerful scripting environment designed for technical users. It is known as the most powerful and flexible statistical programming language available.</td>
</tr>
<tr>
<td><strong>Random Forest</strong></td>
<td>Builds a series of un-pruned decision tree models for a data set. While building each tree, random subsets of the available variables are considered for splitting the data at each node of the tree.</td>
</tr>
<tr>
<td><strong>regression</strong></td>
<td>A traditional approach to modeling used to estimate relationships between variables. The common types of regression are linear, logistic, generalized regression, Poisson, and multinomial regression. Advanced Regression types include Gaussian, Binomial, Negative Binomial, Gamma, and Inverse Gaussian.</td>
</tr>
<tr>
<td><strong>root node</strong></td>
<td>The starting point of the Decision Tree.</td>
</tr>
<tr>
<td><strong>Rpart</strong></td>
<td>A base algorithm, or model builder, in RStat.</td>
</tr>
<tr>
<td><strong>Rpart Rule Export Functionality</strong></td>
<td>Enables the exporting of Rpart rules.</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td><strong>RScript</strong></td>
<td>Script that can run plots, charts, summaries, model techniques, or even be used to execute scoring functionality using R.</td>
</tr>
<tr>
<td><strong>R-squared</strong></td>
<td>Also known as the coefficient of determination, R-squared is a measure of how well data fits a particular statistical model. Many statistical modeling techniques employ R-squared, which is a statistic that has a numerical value ranging from 0 to 1.</td>
</tr>
<tr>
<td><strong>sampling</strong></td>
<td>A common practice to test models on new data. Splits the single data set into two data sets: a training data set used for analysis and a test data set used to evaluate how well a model performs.</td>
</tr>
<tr>
<td><strong>scoring application</strong></td>
<td>Deploys analytic models for repeated use on new data sets by non-technical users to support decision-making. In simple terms, the scoring application labels a prospect as either good or bad.</td>
</tr>
<tr>
<td><strong>scoring routine</strong></td>
<td>Stores model information, including when the model was created, the parameters of the model, the model meta data, and the PMML.</td>
</tr>
<tr>
<td><strong>seed</strong></td>
<td>A numerical value used to initialize a random sampling algorithm or to establish a starting point in a table of random numbers.</td>
</tr>
<tr>
<td><strong>snip</strong></td>
<td>The process of trimming nodes in a tree.</td>
</tr>
<tr>
<td><strong>Spearman's Rank Correlation Coefficient</strong></td>
<td>A non-parametric indicator of the dependencies between two variables which are used to compute the correlation coefficient. That is, it does not assume that the data is normally distributed.</td>
</tr>
</tbody>
</table>
split

The process of splitting a node in a tree into multiple branches or leaves.

standard deviation

Functionality that illustrates the variance from the average (mean) or expected value. Depending on the underlying data set, the types of advanced regression options available may include Normal (Linear), Gaussian, Poisson, Logistic, Multinomial, Binomial, Negative Binomial, Gamma, and Inverse Gaussian.

Support Vector Model (SVM)

A modern approach to modeling where the data is mapped to a higher dimensional space, increasing the possibility that vectors separating the classes will be found.

Survival Data Type

Allows the user to run a Survival Model, that is, Cox Proportional Hazards or Parametric.

Survival Model

Allows the user to run Time-to-Event analysis. When using this model in RStat, the user can select Cox Proportional Hazards or Parametric tests to perform their analysis.

T-test

Used to determine if two sets of data are significantly different from each other, and commonly, it assumes the test statistic follows a normal distribution. There are four common usages:

1. A one-sample location test, which determines if the mean of the normally distributed populations has the specified value in the null hypothesis.

2. A two-sample location test, which determines, under the null hypothesis, if the means of two normally distributed populations are equal.

3. A paired difference test of the null hypothesis, which determines if the difference of two responses measured on the same statistical unit has mean zero.
4. A test of regression slope, which determines if the regression slope is 0.

**variable grid**

A grid that displays the available variables in the current data set. The variable grid appears once the data set has been loaded.

**variance**

The variance measures the dispersion between numbers in a set of numbers.

**Wilcoxon Rank-Sum**

Also known as the Mann-Whitney-Wilcoxon test, Wilcoxon Rank-Sum is analogous to the two-sample T-test, but is performed on the rankings of the combined data sets instead of on the actual measure. If the observation rankings are not different, then the samples are not different. Because it is performed on the rankings, it is more sensitive about the location of the distribution, that is, to the median (not the mean, as in the T-test).

**Wilcoxon Signed Rank**

A non-parametric hypothesis test used to identify the difference in the means amongst sample populations, which can be two related samples, matched samples, or repeated measurements. Analogous to the T-test, Wilcoxon Signed Rank is also a paired difference test.
# Index

64-bit, FEX functionality in 36

## A

Ada Boost 112
analysis of variance from binomial logistic regression 246
analysis of variance from linear regression models 237
ANOVA tables 237, 246
architecture 16
ARIMA 107, 112

## B

Basic Sampling Concepts 200
   confidence level 200
   Margin of Error 200
   population size 200
   Response Distribution 200
   Sample Size 200
   Sampling Steps 201
   Seed 200
Bayesian Network 117
Baysien Network 107
BiCluster 99
Binary Logistic Regression 239
Binomial Logistic regression 244
Boost 107
building decision tree models 107
building linear regression models 225
building logistic models 239
building scoring applications 172, 190

## C

C Routine 172
Chi-SQ Test 89
Coefficients 245
Complexity table 275
correlation analysis 86
Correlation Test 89
creating logistic regression models 241
creating scoring applications 101
Cross Tab 59

## D

data exploration 51, 57
data extracts 199
Data Miners 13
data mining 15
data transformations 91
Decision Tree Analysis
   Data Exploration 267
   identify target groups 267
   Pattern Detection 267
   practical applications 267
   predict outcomes 267
Decision Tree Model 127
  building 140
  creating 267
  defined 265
  diagramming 144
  evaluating 276
  goal of 265
  output 271
  pruning 266
  Rules 142
  splitting 265
  tree selection 266
  Visualizing 141
decision tree models 107
Decision Tree
  Save as a C Routine 153
  Save as unique PDF 153
decision trees
  facts 266
defining variable roles 103
Density Plot 60
Design-Time Count 210
Deviance Residuals 244
disabling sampling 223
displaying model information 187
Distribution Test 88

E

Error Matrix 169

Error rate
  cross-validated 275
  cross-validated, default cross validation 275
  cross-validated, standard error 275
  Resubstitution 275
  estimation 86
  evaluating decision tree models 276
  Ewkm 98
  executing data settings 106
  Expected Loss 274
  exploring data 57
  exporting the final model 172
  extracting samples using categorical stratified sampling 215
  extracting samples using numeric stratified sampling 206
  extracting samples using random sampling 201

F

F test 89
FancyPlots 145
  basic plotted chart 155
  Branch Type 149
  charting 149
  creating 154
  Decision-Tree Snipping 149
  SplitType 149
features 13
FEX 39, 40, 42
  load process 41
Fisher Scoring Iterations 245
Forest 108
Frequency Plot generating 287
fully integrated environment 15

G
GGobi 17
GNU Open Source Gtk+ Runtime Environment 17

H
Hierarchical 99
hypothesis testing 86

I
installing RStat 17
introduction 13

K
kernel 137
KMeans 98

L
Linear Regression 226
output 233
Live 210
loading data from a FEX 37
loading data from within RStat 36
Location&Variance Test 89
logistic models 239
Logistic Regression 239
Education 241
Evaluate 246
Law Enforcement applications 241
Manpower management 241
Marketing applications 241
model 241

M
Market Basket Analysis 277
creating an association model 281
executing 288
exporting function 288
generate rules 286
output 285
Practical Application of 280
Message Viewer 222
missing values 293
Model Configuration
Hold File Name 35
Hold File Path 35
model information 187
Model Sampling 101
Model Types 107
Multiple Linear Regression 229
Multiple Linear Regression models 225, 229
Multiple Linear Regression, creating 231
Index

N
Neural Net 108
Neural Net Model 129
Node cases, number of 274
Node Numbering 273
Number of Splits 275
numeric variable 229

O
output from linear regression models 233

P
PMML 172
Predicted Class for Node 274
predictive modeling 86
Primary Split 274
pruning 266

R
R 16
R Console 36
R Function Call 233, 244
R-squared 236
Random Forest Model 132
random sampling 199
Regression 107
   Cox 251
   Parametric Survival 251
Report canvas 34
Root node error 275
Rpart Rules 143
Rpart Rules Export 163
RScripts 43
   button 46
   C File 46
   command in 48
   creating scoring routines 49
   memory usage 43
   running 43
   to create a C File 46
Rsquare 260
RStat 13, 225
   64-bit 14
   App Studio and 16
   data mining 15
   explore data visually 13
   Getting Started 33
   integration with R 16
   Lift Curve 248
   load CSV 36
   Load Data from FEX 37
   load FEX 36
   load Library 36
   load RData 36
   load RScript 36
   Loading Data From Within 36
   open source R 16
   predictive values 13
RStat 13, 225
  scores 13
  Scoring New Data 170
  Starting with Data Loaded 33

S
  sampling 51, 199
    Random Sampling 199
    Stratified Sampling 199
  Schoenfeld residuals 263
  scoring applications 15, 101, 190
  scoring new data 170
  Scoring Routines 190
    Compiling 182
    deploying 293
    Deploying 182
    Flagging 293
    missing data in 195
    Missing Values 294
    renaming 294
  setting Message Viewer on 222
  setting target types 105
  setting variable roles for groups 104
  simple linear regression models 225, 226
  skewness 59
  Split Point 274
  splitting 265
  stratified sampling 199
  Survival Analysis 251, 254
    Cox Regression 254

Survival Analysis 251, 254
  Engineering 254
  Government and Social Services 254
  Healthcare 254
  Law Enforcement 254
  Marketing Operations 254
  Warranty Applications 254
  Survival Model 136
  SVM 137
    Kernel 137
  system requirements 17

T
  T test 89
  Target Setting 106
  Target Type 105
    Auto 105
    Categoric 105
    Numeric 105
    Survival 105
  Teradata 176
  Terminal Nodes 274
  transformations 51, 91
  Tree 107
  tree construction, variables in 274

U
  User Defined Function 176
  User-defined Count 211
Index

User-Defined Parameters 270

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complexity</td>
<td>270</td>
</tr>
<tr>
<td>Consideration</td>
<td>270</td>
</tr>
<tr>
<td>Default Value</td>
<td>270</td>
</tr>
<tr>
<td>Loss Matrix</td>
<td>270</td>
</tr>
<tr>
<td>Max Depth</td>
<td>270</td>
</tr>
<tr>
<td>Min Bucket</td>
<td>270</td>
</tr>
<tr>
<td>Min Split</td>
<td>270</td>
</tr>
</tbody>
</table>

User-Defined Parameters 270

priors 270

V

variable definitions 51

W

Winning class, probability of 274
Feedback

Customer success is our top priority. Connect with us today!

Information Builders Technical Content Management team is comprised of many talented individuals who work together to design and deliver quality technical documentation products. Your feedback supports our ongoing efforts!

You can also preview new innovations to get an early look at new content products and services. Your participation helps us create great experiences for every customer.

To send us feedback or make a connection, contact Sarah Buccellato, Technical Editor, Technical Content Management at Sarah_Buccellato@ibi.com.

To request permission to repurpose copyrighted material, please contact Frances Gambino, Vice President, Technical Content Management at Frances_Gambino@ibi.com.