R Language provides the built-in functions to make a graphs, statistical analysis and Business logic. Users can utilize the functions to create a chart and sequences. It compiles and runs on a wide variety of UNIXplatforms, Windows and MacOS.

High level applications such as Java can't directly access R objects. JRI (Java R Interface) provides the interface to access R objects from Java.

EspressChart provides the capability of drawing different chart types other than LINE chart with data from R. Many other useful chart attributes can also be added simply by its Chart Designer or by EspressAPI.

We use a standalone EspressAPI java code demonstrate how to use R with EspressChart. The java code draws PIE, COL and BAR charts from R object and resulting chart applet display.

STEP-1: Install R

Let's install R on Windows.

First, download the binaries file.

Then, double click the binaries file to install R on Windows (see Figure 1). After R has been installed, start R to log in to the R Console.

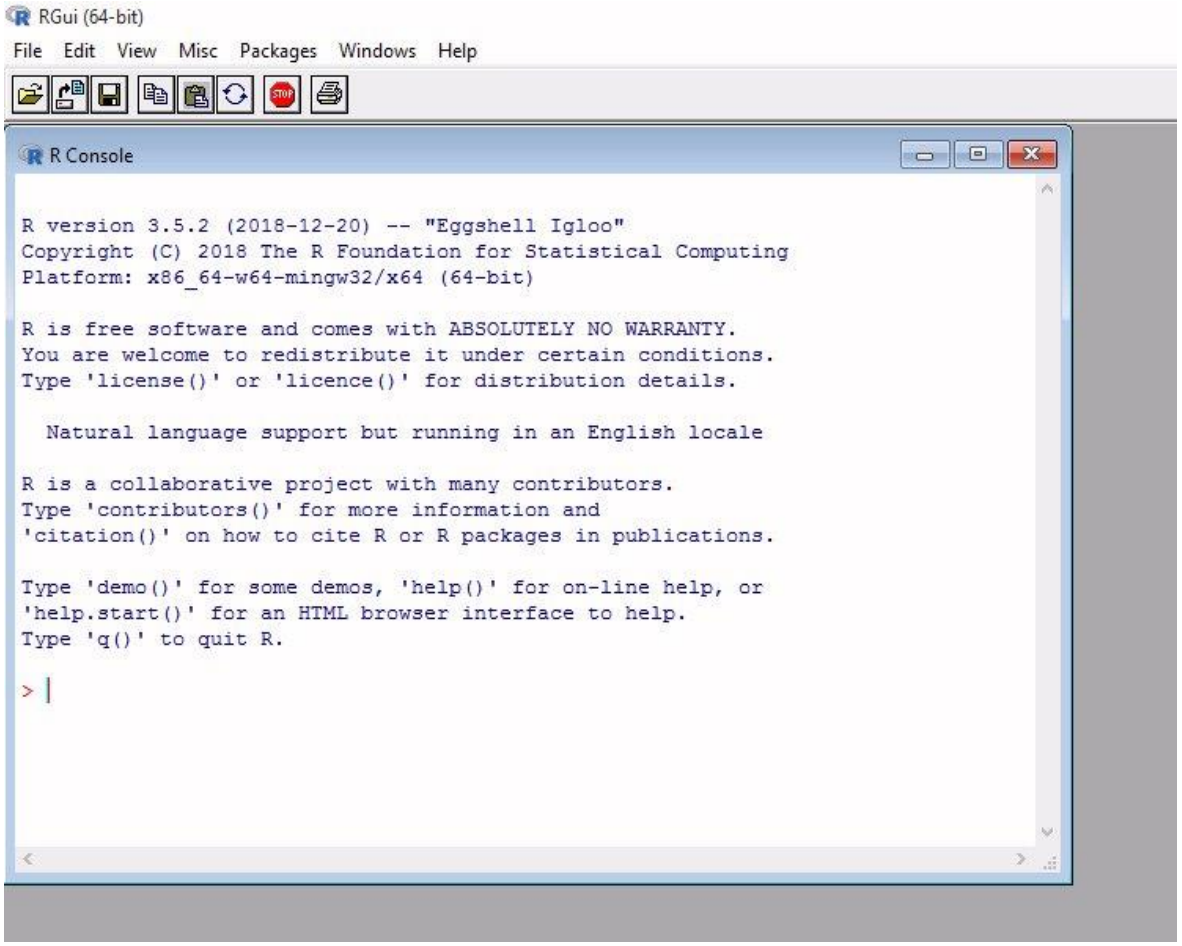

Figure 1: R console

STEP-2: Install rJava Package

JRI comes bundled with `rJava`, so the best way is to simply install `rJava`.

In R Console, run the following command to install `rJava`:

```
> install.packages('rJava')
```

STEP-3: Configure Environment variable

`R_HOME`: the directory where R is installed. For example: `C:\R-3.5.2`

`PATH`: add `%R_HOME%\bin\x64\;%R_HOME%\library\rJava\jri\x64\`

STEP-4: Download java jars

From JRI jars download link, download `JRI.jar`, `JRIEngine.jar` and `REngine.jar`

STEP-5: Compile and Run Java Example

We put Java file and jar files in `C:\Java_Example\`, to compile and run the Java example (`SampleChart.java`), please open a *Command Prompt* and then run the following commands.

```
C:\java_example>set JAVA_HOME=C:\jdk1.8.0_131

C:\java_example>%JAVA_HOME%\bin\javac SampleChart.java

C:\java_example>Set classpath=.;EspressAPI.jar;qblicense.jar;
JRI.jar;REngine.jar;JRIEngine.jar

C:\java_example>%JAVA_HOME%\bin\java SampleChart
```

Java source code (SampleChart.java)

```
//Import all necessary classes
import quadbase.ChartAPI.*;
import quadbase.util.*;
import java.awt.*;
import java.applet.*;

import org.rosuda.JRI.REXP;
import org.rosuda.JRI.Rengine;

public class SampleChart extends Frame {
      public QbChart chart;
      public SampleChart(String[] args) {
            start(args);
      }

public void start(String[] args) {
      QbChart.setChartServerUsed(false);
      setLayout(new BorderLayout());
      Rengine r = new Rengine(args, false, null);
      //Do some calcs and plot the chart but save as a png in the
      working folder
      r.eval("attach(faithful)"); //faithful is the dataset from R
```

```
double[] dvalue=r.eval("summary(eruptions)").asDoubleArray();
String[] dataTypes = {"String", "int"};
String[] colNames = {"Column A", "Column B"};
String[][] records=new String[6][2];

for (int i=0;i<dvalue.length;i++)
{
      for (int j=0;j<=1;j++)
      {
            records[i][0]="Min"+i+"";
            records[i][1]=""+dvalue[i]+"";
      }
}

DbData data = new DbData(dataTypes, colNames, records);

ColInfo colInfo = new ColInfo(-1, 0, -1, 1);

// Changing QbChart.COL in the java code to QbChart.BAR or
QBChart.PIE will produce the BAR and PIE charts, respectively.
QbChart chart = new QbChart((Applet)null, QbChart.VIEW2D,
QbChart.COL, data, false, colInfo, null);

// Set XAxis
IAxis hXAxis = chart.gethXAxis();
hXAxis.setGridVisible(false);

// Set YAxis
IAxis hYAxis = chart.gethYAxis();
hYAxis.setGridVisible(false);

// Set all axises
hXAxis.setArrowhead(false);
hXAxis.setThickness(5);

// Set titles
ITextString hXTitle, hYTitle, hSTitle;
hXTitle = hXAxis.gethTitle();
hXTitle.setValue("Category");
hXTitle.setColor(Color.black);
hYTitle = hYAxis.gethTitle();
hYTitle.setValue("Value Axis");
hYTitle.setColor(Color.black);

// Set labels
hXAxis.gethLabel().setColor(Color.black);
hYAxis.gethLabel().setColor(Color.black);

// Set chart plot
IPlot hChartPlot = chart.gethChartPlot();
hChartPlot.setBorderVisible(true);
hChartPlot.setBorderColor(Color.pink);

// Set legends
```

```
chart.gethLegend().setBackgroundColor(Color.orange);

      add("Center", chart);
      }

      //Start
      public static void main(String[] args) {
            SampleChart t = new SampleChart(args);
            t.resize(500,550);
            t.setVisible(true);
      }

}
```

Images generated from SampleChart.java

The COL chart is generated as shown below. Changing QbChart.COL in the java code to QbChart.BAR or QBChart.PIE will produce the BAR and PIE charts, respectively. Their images are also shown below.

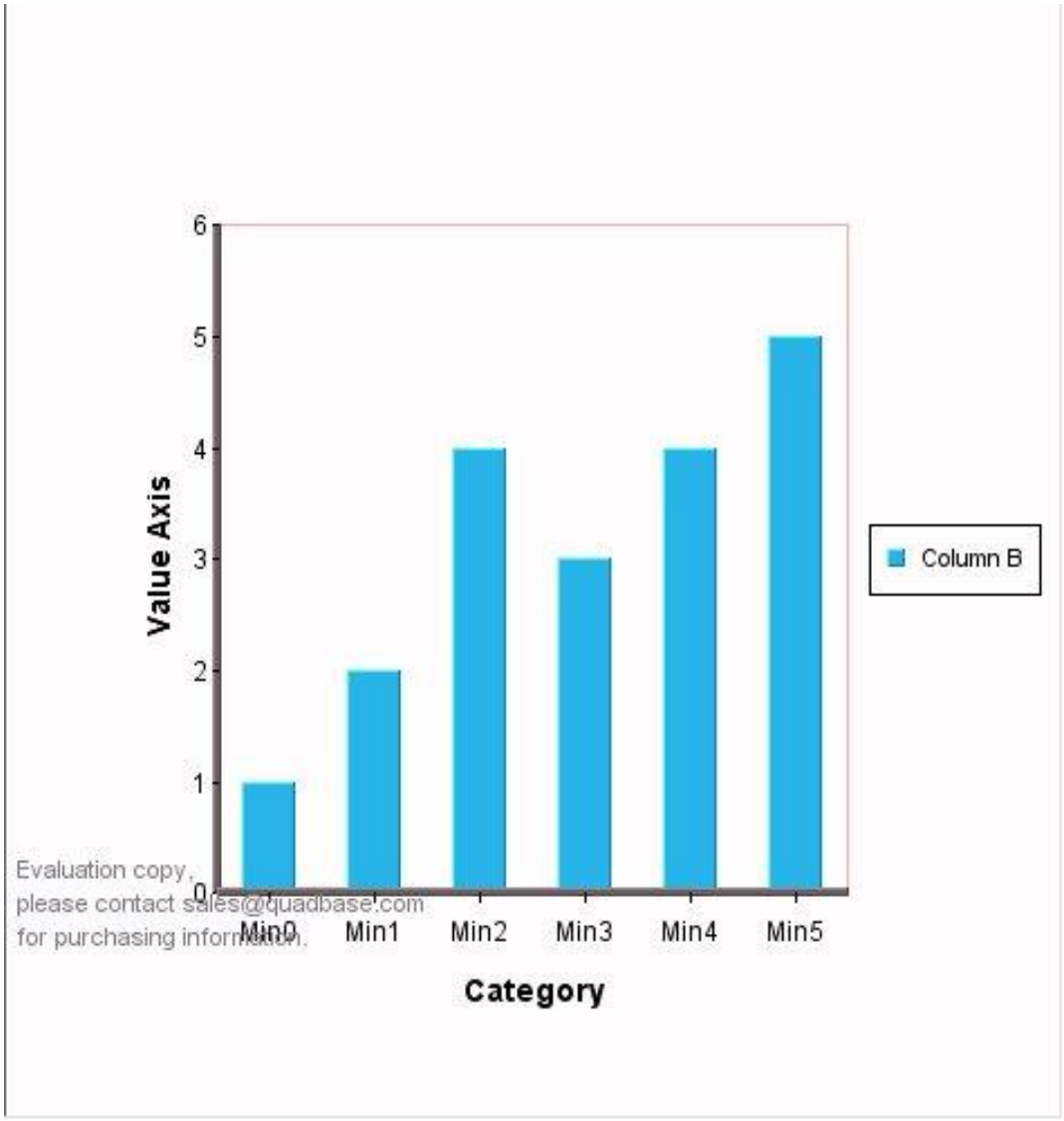

Figure 2: COL Image

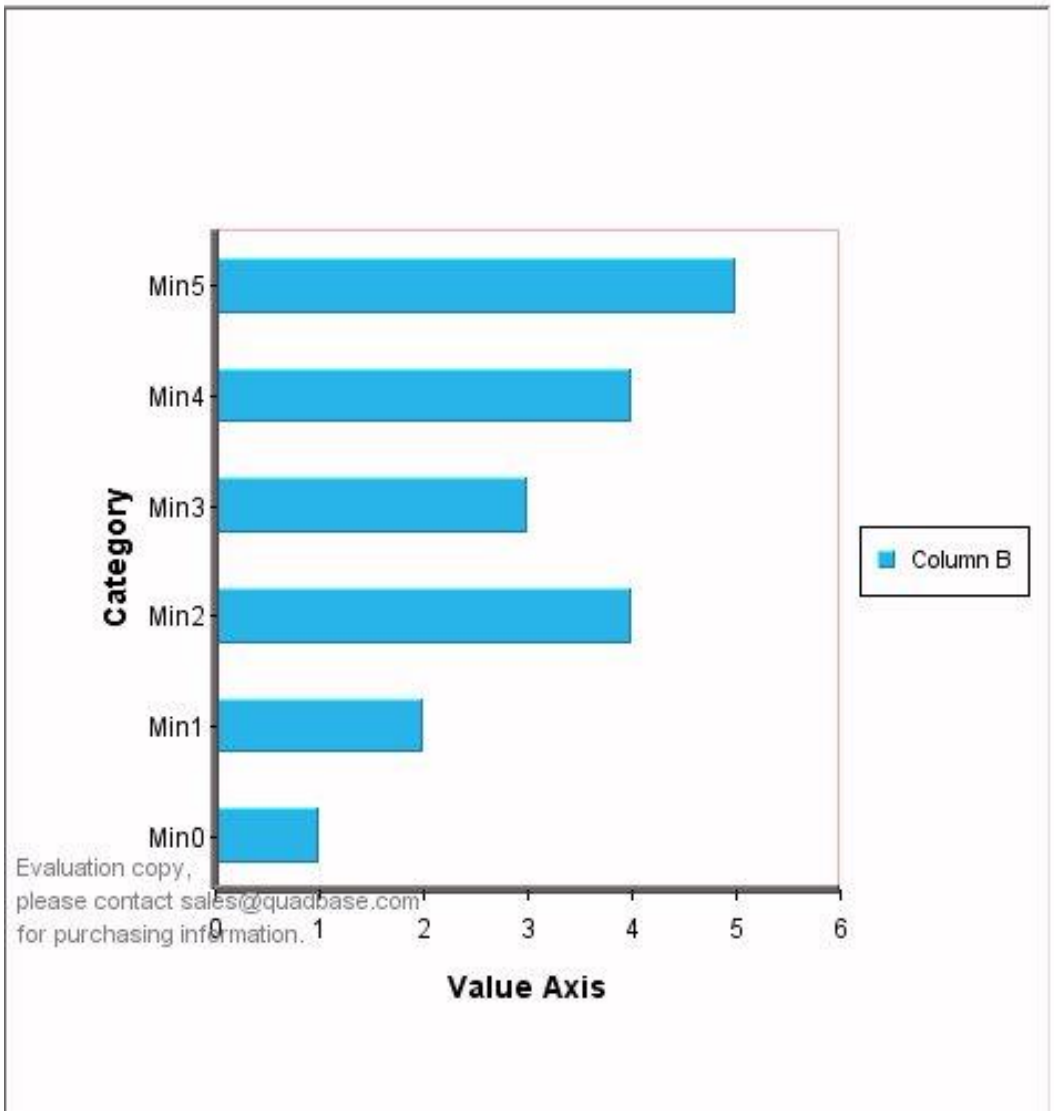

Figure-3: BAR Image

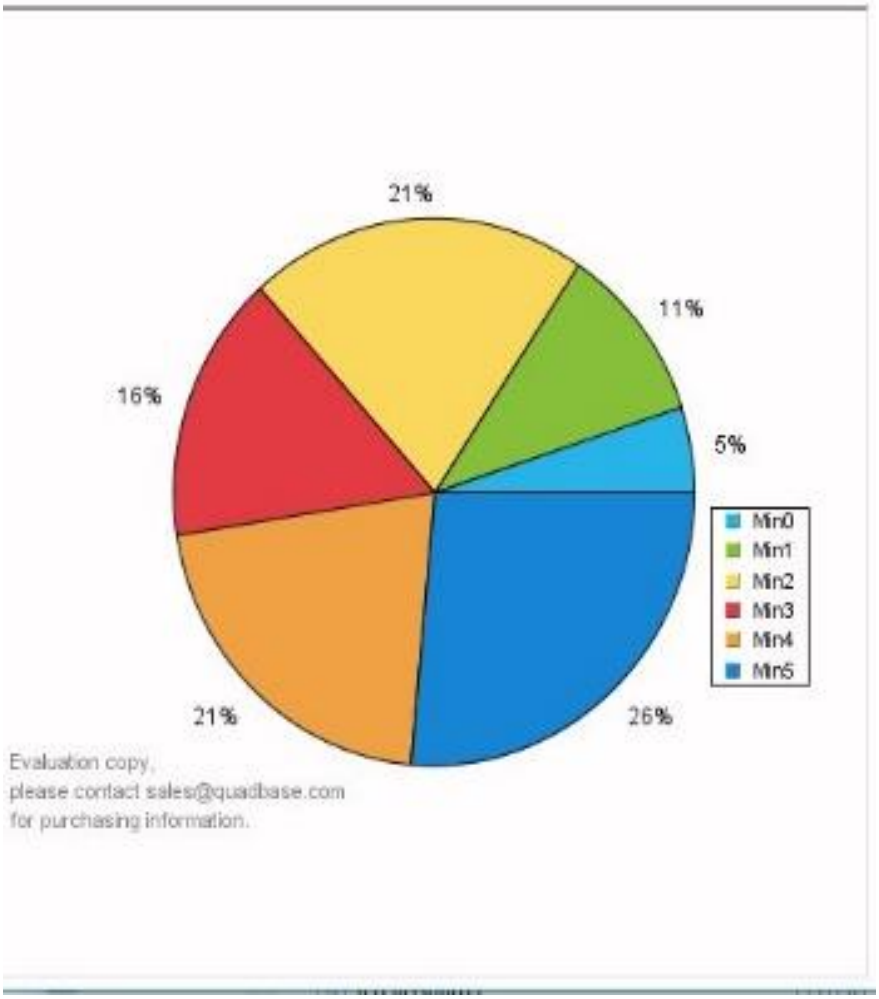

Figure 4: PIE Image

Conclusion

R provides many built -in functions to minimize the user's coding. It surely helps whoever needs a statistical analysis and data mining features.