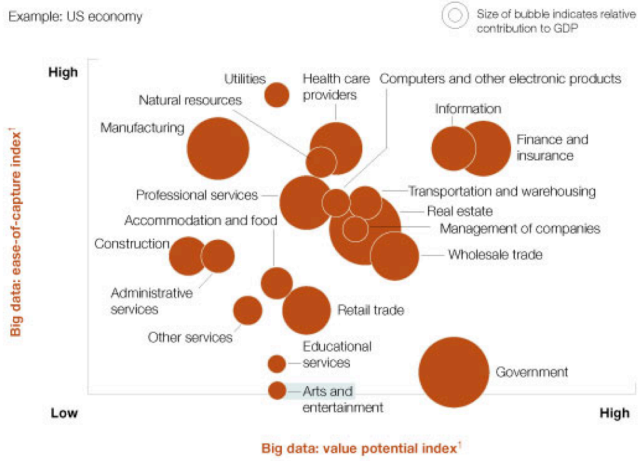
REVOLUTION ANALYTICS

Big data analytics with RRE: Introduction

Example: US economy



[1] For detailed explication of metrics, see appendix in McKinsey Global Institute full report *Big data: The next frontier for innovation, competition, and productivity*, available free of charge online at mckinsey.com/mgi.

Source: US Bureau of Labor Statistics; McKinsey Global Institute analysis

The challenges of Big Data

- Move
- Merge
- Manage
- Munge

The challenges of big data for R

- All data must fit in memory
- Produces multiple copies of read-only data
- Slows down with data size

Airline Delay

Will you still get there on time?

FAA Dataset

2007, USA, 7.5M obs, 29 variables

Import the data

```
dataDir <- "../data"
AirlineCsv <- file.path(dataDir, "2007.csv")
AirlineXdf <- file.path(dataDir, "2007.xdf")
system.time(rxImport(inData = AirlineCsv, outFile = AirlineXdf, overwrite = TRUE))

## Rows Read: 500000, Total Rows Processed: 500000, Total Chunk Time: 11.707 seconds
## Rows Read: 500000, Total Rows Processed: 1000000, Total Chunk Time: 11.950 seconds
## Rows Read: 500000, Total Rows Processed: 1500000, Total Chunk Time: 11.784 seconds
## Rows Read: 500000, Total Rows Processed: 2000000, Total Chunk Time: 11.879 seconds
## Rows Read: 500000, Total Rows Processed: 2500000, Total Chunk Time: 11.463 seconds
## Rows Read: 500000, Total Rows Processed: 3000000, Total Chunk Time: 12.257 seconds
## Rows Read: 500000, Total Rows Processed: 3500000, Total Chunk Time: 12.304 seconds
## Rows Read: 500000, Total Rows Processed: 4000000, Total Chunk Time: 12.289 seconds
...

##    user  system elapsed
##   96.23   83.79  183.08
```

Inspect the data

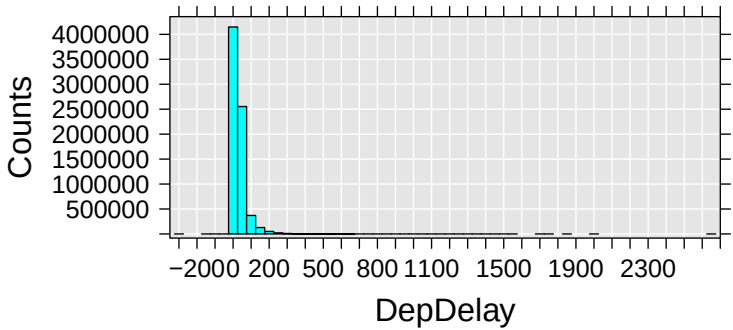
```
rxOptions(reportProgress = 0)
rxGetInfo(data = AirlineXdf, getVarInfo = TRUE, varsToKeep = c("ArrDelay",
  "DepDelay", "Distance"))

## File name: E:\data\2007.xdf
## Number of observations: 7453215
## Number of variables: 29
## Number of blocks: 15
## Compression type: zlib
## Variable information:
## Var 1: ArrDelay, Type: integer, Low/High: (-312, 2598)
## Var 2: DepDelay, Type: integer, Low/High: (-305, 2601)
## Var 3: Distance, Type: integer, Low/High: (11, 4962)
```

Summarize Some Variables

```
rxSummary(~DepDelay + Distance, data = AirlineXdf)

## Call:
## rxSummary(formula = ~DepDelay + Distance, data = AirlineXdf)
##
## Summary Statistics Results for: ~DepDelay + Distance
## Data: AirlineXdf (RxXdfData Data Source)
## File name: E:\data\2007.xdf
## Number of valid observations: 7453215
##
##  Name     Mean      StdDev    Min  Max  ValidObs MissingObs
##  DepDelay  11.39914  36.14189 -305 2601 7292467  160748
##  Distance 719.80579 562.30512   11 4962 7453215       0
```

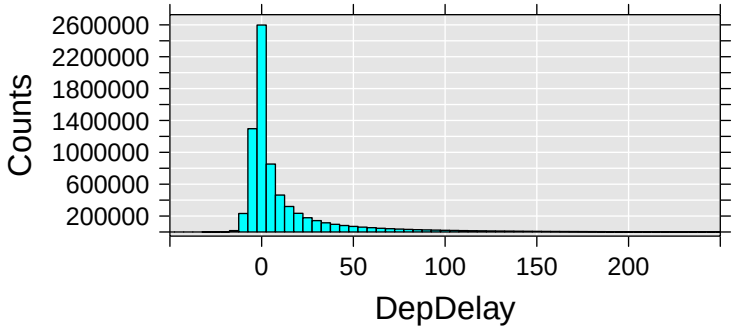
Visually Inspect

```
rxHistogram(~DepDelay, data = AirlineXdf)
```



Visual Inspect a Subset

```
rxHistogram(~DepDelay, data = AirlineXdf, xAxisMinMax = c(-50, 250),
  numBreaks = 500, xNumTicks = 10)
```



Transforms: Create a Variable

```
newAirlineXdf = file.path(dataDir,"2007_airSpeed.xdf")
system.time(rxDataStep(inData=AirlineXdf, outFile=newAirlineXdf,
            varsToKeep=c("AirTime", "Distance","DepDelay","ArrDelay"),
            transforms = list(airSpeed = Distance / AirTime),
              overwrite=TRUE))

##    user  system elapsed
##    5.33    0.32    6.14
```

Summarize the XDF

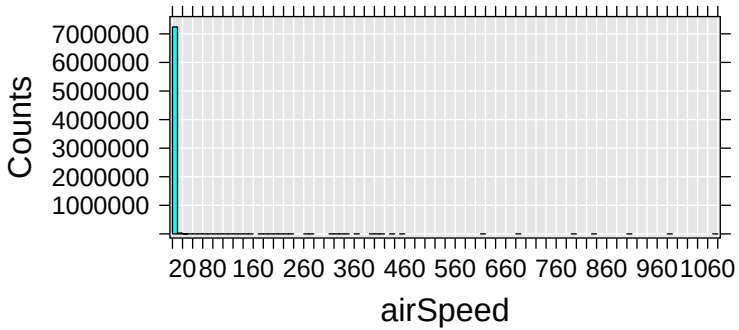
```
rxGetInfo(data = newAirlineXdf, getVarInfo = TRUE)

## File name: E:\data\2007_airSpeed.xdf
## Number of observations: 7453215
## Number of variables: 5
## Number of blocks: 15
## Compression type: zlib
## Variable information:
## Var 1: AirTime, Type: integer, Low/High: (0, 1257)
## Var 2: Distance, Type: integer, Low/High: (11, 4962)
## Var 3: DepDelay, Type: integer, Low/High: (-305, 2601)
## Var 4: ArrDelay, Type: integer, Low/High: (-312, 2598)
## Var 5: airSpeed, Type: numeric, Low/High: (0.2053, 1074.0000)
```

Summarize the New Variable

```
rxSummary(~airSpeed, data = newAirlineXdf)

## Call:
## rxSummary(formula = ~airSpeed, data = newAirlineXdf)
##
## Summary Statistics Results for: ~airSpeed
## Data: newAirlineXdf (RxXdfData Data Source)
## File name: E:\data\2007_airSpeed.xdf
## Number of valid observations: 7453215
##
##  Name     Mean     StdDev    Min       Max  ValidObs MissingObs
##  airSpeed 6.559545 1.986091 0.2052506 1074 7275191  178024
```

Visualize the New Variable

```
rxHistogram(~airSpeed, data = newAirlineXdf)
```

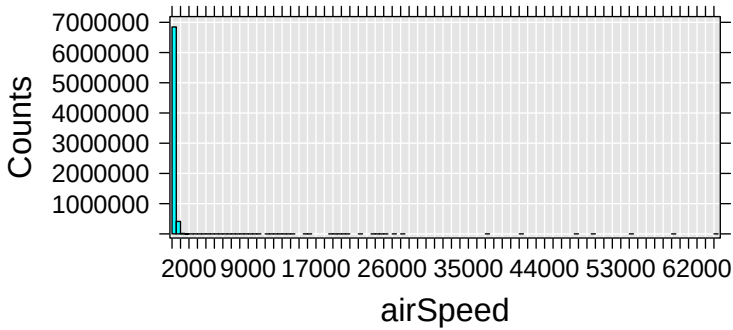


Scale Airspeed to mph

```
rxDataStep(inData=newAirlineXdf, outFile=newAirlineXdf,
           varsToKeep=c("airSpeed"),
               transformVars=c("airSpeed"),
             transforms=list(airSpeed=airSpeed * 60),
               overwrite=TRUE)
```

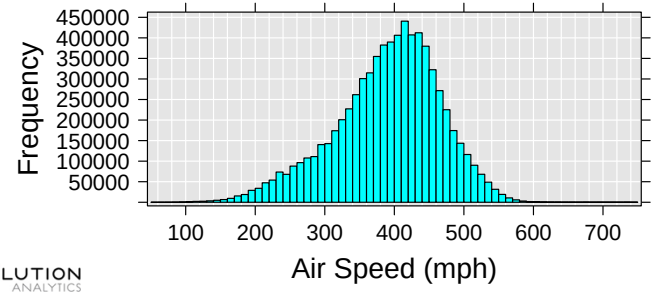
Visualize airSpeed

```
rxHistogram(~airSpeed, data = newAirlineXdf)
```



Visualize airSpeed in Detail

```
rxHistogram(~airSpeed, data=newAirlineXdf,
            rowSelection=(airSpeed>50) & (airSpeed<750),
            scales=list(x=list(at=seq(100,700,by=100))),
            xlab=list(label="Air Speed (mph)"),
            ylab=list(label="Frequency"),
            numBreaks=5000,xNumTicks=40)
```



Correlations

```
rxCor(formula = ~DepDelay + ArrDelay + airSpeed, data = newAirlineXdf)

##             DepDelay    ArrDelay    airSpeed
## DepDelay 1.00000000  0.93150405  0.01403577
## ArrDelay 0.93150405  1.00000000 -0.04502653
## airSpeed 0.01403577 -0.04502653  1.00000000
```

Correlations (subset)

```
rxCor(formula = ~DepDelay + ArrDelay + airSpeed, data = newAirlineXdf,
  rowSelection = (airSpeed > 50) & (airSpeed < 800))

##             DepDelay    ArrDelay    airSpeed
## DepDelay 1.00000000  0.93192674  0.02083635
## ArrDelay 0.93192674  1.00000000 -0.06705657
## airSpeed 0.02083635 -0.06705657  1.00000000
```

Linear Model

```
system.time(reg1 <- rxLinMod(formula = airSpeed ~ DepDelay, data = newAirlineXdf,
  rowSelection = (airSpeed > 50) & (airSpeed < 800)))

##    user  system elapsed
##    1.13    0.16    1.42
```

rxLinMod Object

```
class(reg1)
```

```
## [1] "rxLinMod"
```

```
names(reg1)
```

```
##  [1] "coefficients"      "residual.squares"  "condition.number"
##  [4] "rank"              "aliased"           "coef.std.error"
##  [7] "coef.t.value"      "coef.p.value"      "total.squares"
## [10] "y.var"             "sigma"             "residual.variance"
## [13] "r.squared"         "f.pvalue"          "df"
## [16] "y.names"           "deviance"          "aic"
## [19] "params"            "formula"           "call"
## [22] "fstatistics"       "adj.r.squared"     "nValidObs"
## [25] "nMissingObs"       "coefLabelStyle"
```

Linear Model Summary

```
print(summary(reg1), header = FALSE)
```

```
## Coefficients:
##                Estimate  Std. Error  t value Pr(>|t|)
## (Intercept) 390.4019846   0.0295551 13209.28 2.22e-16 ***
## DepDelay      0.0438940   0.0007822    56.12 2.22e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 75.9 on 7250010 degrees of freedom
## Multiple R-squared: 0.0004342
## Adjusted R-squared: 0.000434
## F-statistic:  3149 on 1 and 7250010 DF,  p-value: < 2.2e-16
## Condition number: 1
```

Nonlinear relationship?

One approach: Discretize departure delay.

```
rxDataStep(inData = newAirlineXdf, outFile = newAirlineXdf,
           varsToKeep="DepDelay", transformVars="DepDelay",
           transforms = list(F_DepDelay=cut(DepDelay, breaks=seq(from=-10, to=100, by=10))),
           append="cols", overwrite=TRUE)
```

Categorical Variables

```
print(rxSummary(~F_DepDelay, data = newAirlineXdf), header = FALSE)
```

```
##
## Category Counts for F_DepDelay
## Number of categories: 11
## Number of valid observations: 6819077
## Number of missing observations: 634138
##
##  F_DepDelay Counts
##  (-10,0]    3894806
##  (0,10]     1316316
##  (10,20]     554103
##  (20,30]     321440
##  (30,40]     211002
##  (40,50]     150920
##  (50,60]     114696
##  (60,70]      87504
```

linMod and Categorical variables

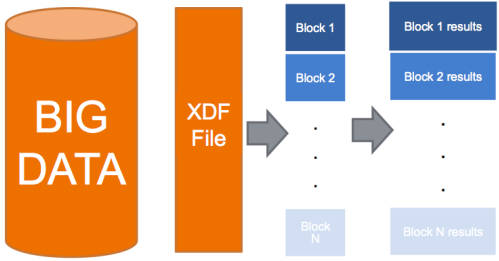
```
reg2 <- rxLinMod(formula = airSpeed ~ F_DepDelay, data = newAirlineXdf,
  rowSelection = (airSpeed > 50) & (airSpeed < 800), dropFirst = TRUE)
```

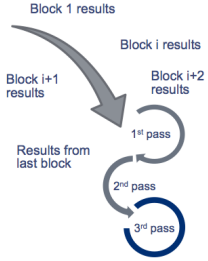
rxlinMod Summary

```
print(summary(reg2), header = FALSE)

## Coefficients: (1 not defined because of singularities)
##                       Estimate Std. Error   t value Pr(>|t|)
## (Intercept)          387.89666    0.03821 10151.854 2.22e-16 ***
## F_DepDelay=(-10,0]     Dropped    Dropped   Dropped  Dropped
## F_DepDelay=(0,10]     10.97080    0.07603   144.291 2.22e-16 ***
## F_DepDelay=(10,20]     9.72038    0.10829    89.758 2.22e-16 ***
## F_DepDelay=(20,30]     8.74882    0.13843    63.199 2.22e-16 ***
## F_DepDelay=(30,40]     7.56250    0.16862    44.850 2.22e-16 ***
## F_DepDelay=(40,50]     6.25256    0.19792    31.591 2.22e-16 ***
## F_DepDelay=(50,60]     5.52251    0.22610    24.425 2.22e-16 ***
## F_DepDelay=(60,70]     4.47101    0.25797    17.332 2.22e-16 ***
## F_DepDelay=(70,80]     3.76816    0.29028    12.981 2.22e-16 ***
## F_DepDelay=(80,90]     3.32997    0.32385    10.283 2.22e-16 ***
## F_DepDelay=(90,100]    2.50516    0.36051     6.949 3.68e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

RevoScaleR Intro: PEMA

Parallel External Memory Algorithms





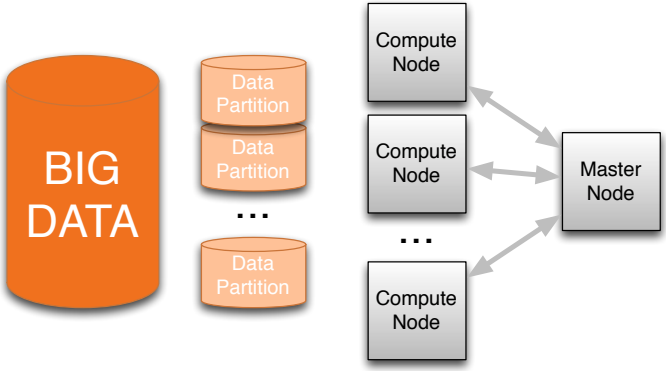
Read Blocks, Compute Intermediate Results in Parallel, Iterating as Necessary

Distributed Environments

- Local
- Cluster
  - Microsoft HPC
  - Microsoft Azure Burst
  - LSF

REVOLUTION ANALYTICS

Distributed Environments

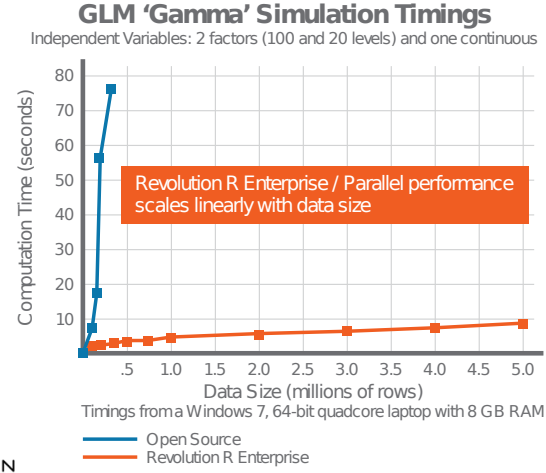


RevoScaleR ComputeContext

One Line of Code for all supported architectures

- Defines Hardware
- Handles Distribution, Monitoring, and Failover via native job scheduler

Performance



Summary

RevoScaleR provides Fast and efficient ways to process Big Data:

- Import
- Explore
- Manipulate
- Visualize
- Analyze

REVOLUTION ANALYTICS

Thank you

Revolution Analytics is the leading commercial provider of software and support for the popular open source R statistics language.

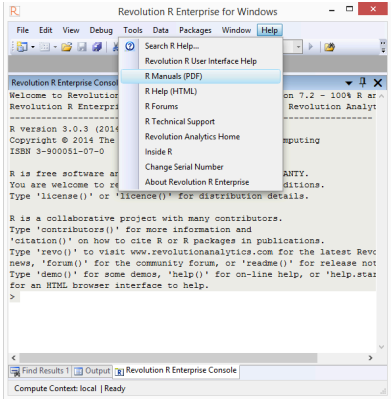
www.revolutionanalytics.com, 1.855.GET.REVO,
Twitter: @RevolutionR

REVOLUTION ANALYTICS

Big data analytics with RRE: Data Exploration

September 11, 2014

Documentation



http://packages.revolutionanalytics.com/doc/7.2.0/

rxOptions()

```
## help(rxOptions)
names(rxOptions())

##  [1] "cintSysDir"             "includeDir"
##  [3] "libDir"                 "linkDllName"
##  [5] "unitTestDir"            "unitTestDataDir"
##  [7] "sampleDataDir"          "demoScriptsDir"
##  [9] "fileSystem"             "hdfsPort"
## [11] "hdfsHost"               "xdfCompressionLevel"
## [13] "computeContext"         "blocksPerRead"
## [15] "reportProgress"         "rowDisplayMax"
## [17] "memStatsReset"          "memStatsDiff"
## [19] "numCoresToUse"          "numDigits"
## [21] "showTransformFn"        "defaultDecimalColType"
## [23] "defaultMissingColType" "dataPath"
## [25] "outDataPath"            "transformPackages"
## [27] "useSparseCube"          "dropMain"
...
```

Sample Data

```
sampleDataDir <- rxOptions()[["sampleDataDir"]]
dir(sampleDataDir)
```

```
##  [1] "AirlineDemo1kNoMissing.csv" "AirlineDemoSmall.csv"
##  [3] "AirlineDemoSmall.xdf"       "AirlineDemoSmallUC.xdf"
##  [5] "CensusWorkers.xdf"          "claims.dat"
##  [7] "claims.sas7bdat"            "claims.sav"
##  [9] "claims.sd7"                 "claims.sqlite"
## [11] "claims.sts"                 "claims.txt"
## [13] "claims.xdf"                 "claims_.txt"
## [15] "claims4blocks.xdf"          "claimsExtra.txt"
## [17] "claimsQuote.txt"            "claimsTab.txt"
## [19] "claimsTxt"                  "claimsXdf"
## [21] "CustomerSurvey.xdf"         "DJIAdaily.xdf"
## [23] "fourthgraders.xdf"          "Kyphosis.xdf"
## [25] "mortDefaultSmall.xdf"       "mortDefaultSmall2000.csv"
## [27] "mortDefaultSmall2001.csv"   "mortDefaultSmall2002.csv"
...
```

Summarize the XDF

```
DJIAdata <- file.path(sampleDataDir, "DJIAdaily.xdf")
rxGetInfo(DJIAdata)

## File name: C:\Revolution\R-Enterprise-7.2\R-3.0.3\library\RevoScaleR\SampleData\DJIAdaily.xdf
## Number of observations: 20636
## Number of variables: 13
## Number of blocks: 4
## Compression type: zlib
```

Summarize the variables

```
## rxGetInfo(DJIAdata, getVarInfo = TRUE)
rxGetVarInfo(DJIAdata)

## Var 1: Date, Type: character
## Var 2: Open, Type: numeric, Storage: float32, Low/High: (41.6300, 14165.0195)
## Var 3: High, Type: numeric, Storage: float32, Low/High: (42.6100, 14279.9600)
## Var 4: Low, Type: numeric, Storage: float32, Low/High: (40.5600, 13980.9004)
## Var 5: Close, Type: numeric, Storage: float32, Low/High: (41.2200, 14164.5303)
## Var 6: Volume, Type: numeric, Storage: float32, Low/High: (130000.0000, 11456230400.0000)
## Var 7: Adj.Close, Type: numeric, Storage: float32, Low/High: (41.2200, 14164.5303)
## Var 8: Year, Type: integer, Low/High: (1928, 2010)
## Var 9: Month, Type: integer, Low/High: (1, 12)
...
```

GetVarInfo - Details

```
GetVarInfoObject <- rxGetVarInfo(DJIAdata)
class(GetVarInfoObject)

## [1] "rxGetVarInfo"


is.list(GetVarInfoObject)

## [1] TRUE


str(GetVarInfoObject)

## List of 13
##  $ Date         :List of 4
##   ..$ varType: chr "character"
##   ..$ storage: chr "string"
##   ..$ low    : logi NA
##   ..$ high   : logi NA
##   ..- attr(*, "class")= chr "rxVarInfo"
##  $ Open         :List of 4
##   ..$ varType: chr "numeric"
...
```

VarInfo - Details

```
str(GetVarInfoObject[1])
```

```
## List of 1
##  $ Date:List of 4
##   ..$ varType: chr "character"
##   ..$ storage: chr "string"
##   ..$ low    : logi NA
##   ..$ high   : logi NA
##   ..- attr(*, "class")= chr "rxVarInfo"
```

```
class(GetVarInfoObject[[1]])
```

```
## [1] "rxVarInfo"
```

VarInfo for Open

```
GetVarInfoObject[["Open"]]
```

```
## Type: numeric, Storage: float32, Low/High: (41.6300, 14165.0195)
```

```
class(GetVarInfoObject[["Open"]])
```

```
## [1] "rxVarInfo"
```

```
names(GetVarInfoObject[["Open"]])
```

```
## [1] "varType" "storage" "low"     "high"
```

```
GetVarInfoObject[["Open"]][["storage"]]
```

```
## [1] "float32"
```

VarInfo for DayOfWeek

```
names(GetVarInfoObject[["DayOfWeek"]])
```

```
## [1] "varType" "storage" "low"     "high"    "levels"
```

```
GetVarInfoObject[["DayOfWeek"]][["levels"]]
```

```
## [1] "Monday"    "Tuesday"   "Wednesday" "Thursday"  "Friday"
```

rxSummary

```
rxSummary(~Open, data = DJIAdata)
```

```
## Call:
## rxSummary(formula = ~Open, data = DJIAdata)
##
## Summary Statistics Results for: ~Open
## Data: DJIAdata (RxXdfData Data Source)
## File name:
##     C:/Revolution/R-Enterprise-7.2/R-3.0.3/library/RevoScaleR/SampleData/DJIAdaily.xdf
## Number of valid observations: 20636
##
##  Name Mean StdDev Min   Max   ValidObs MissingObs
##  Open 2516 3650   41.63 14165 20636    0
```

rxQuantile

```
rxQuantile(varName = "Open", data = DJIAdata)
```

```
##     0%   25%   50%   75%  100%
##     41   241   837  2694 14165
```

```
## help(rxQuantile)
```

rxSummary: More Variables

```
dji.rxs <- rxSummary(~Open + High + Low + Close, data = DJIAdata)

##  Name  Mean StdDev Min   Max   ValidObs MissingObs
##  Open  2516 3650   41.63 14165 20636    0
##  High  2544 3692   42.61 14280 20636    0
##  Low   2489 3607   40.56 13981 20636    0
##  Close 2517 3650   41.22 14165 20636    0

## rxSummary(~., DJIAdata)
```

rxSummary help

```
## help(rxSummary)
args(rxSummary)

## function (formula, data, byGroupOutFile = NULL, summaryStats = c("Mean",
##     "StdDev", "Min", "Max", "ValidObs", "MissingObs"), byTerm = TRUE,
##     pweights = NULL, fweights = NULL, rowSelection = NULL, transforms = NULL,
##     transformObjects = NULL, transformFunc = NULL, transformVars = NULL,
##     transformPackages = NULL, transformEnvir = NULL, overwrite = FALSE,
##     useSparseCube = rxGetOption("useSparseCube"), removeZeroCounts = useSparseCube,
##     blocksPerRead = rxGetOption("blocksPerRead"), rowsPerBlock = 100000,
##     reportProgress = rxGetOption("reportProgress"), verbose = 0,
##     computeContext = rxGetOption("computeContext"), ...)
## NULL
```

rxSummary: Weighted Averages

```
print(rxSummary(~Open + High + Low + Close, data = DJIAdata, fweights = "Volume"),
  header = FALSE)
```

```
##  Name  Mean StdDev Min   Max   SumOfWeights  MissingWeights
##  Open  9682 2714   41.63 14165 8612070155344 0
##  High  9800 2734   42.61 14280 8612070155344 0
##  Low   9559 2691   40.56 13981 8612070155344 0
##  Close 9682 2712   41.22 14165 8612070155344 0
```

rxCrossTabs

```
print(rxCrossTabs(~DayOfWeek, data = DJIAdata), header = FALSE)
```

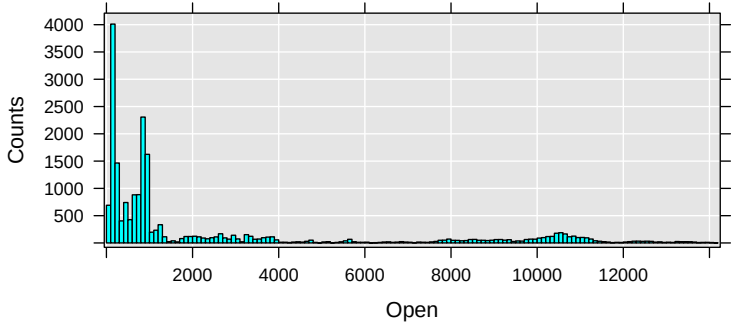
```
## DayOfWeek (counts):
##
## Monday    3989
## Tuesday   4182
## Wednesday 4205
## Thursday  4139
## Friday    4121
```

rxCrossTabs: Weighted

```
print(rxCrossTabs(~DayOfWeek, data = DJIAdata, fweights = "Volume"),
  header = FALSE)

## DayOfWeek,fweight=Volume (counts):
##
## Monday    1512617311744
## Tuesday   1774277389504
## Wednesday 1821943518576
## Thursday  1802207933264
## Friday    1701024002256
```

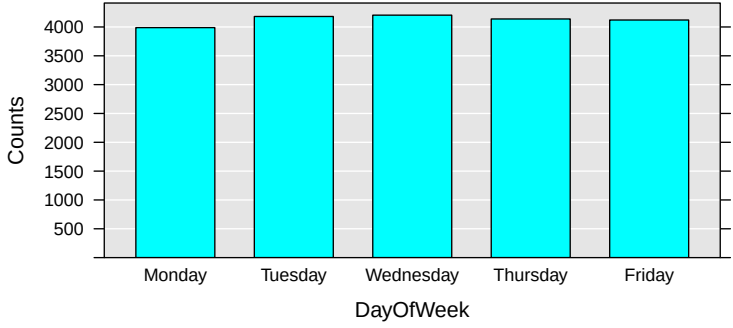
Visualization

```
rxHistogram(~Open, data = DJIAdata, xNumTicks = 10, )
```



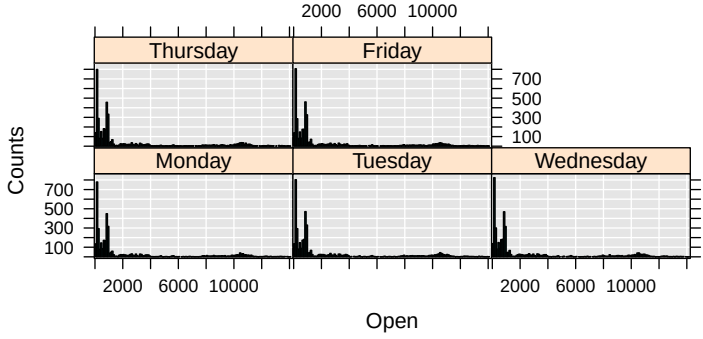
Visualization: Factors

```
rxHistogram(~DayOfWeek, data = DJIAdata)
```



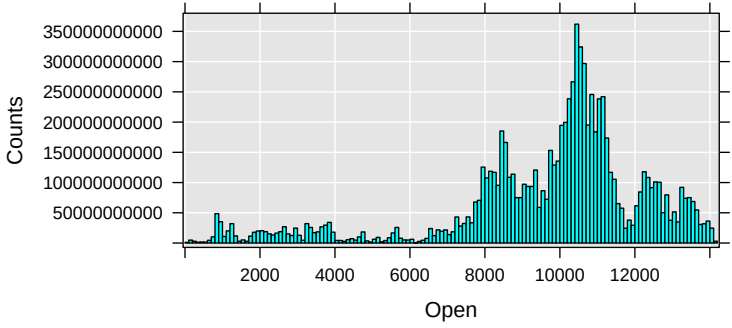
Visualization: Facets

```
rxHistogram(~Open | DayOfWeek, data = DJIAdata, xNumTicks = 10)
```



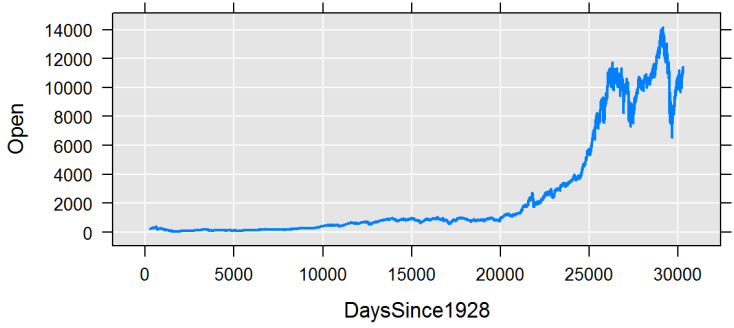
Visualization: weights

```
rxHistogram(~Open, data = DJIAdata, fweights = "Volume", xNumTicks = 10)
```



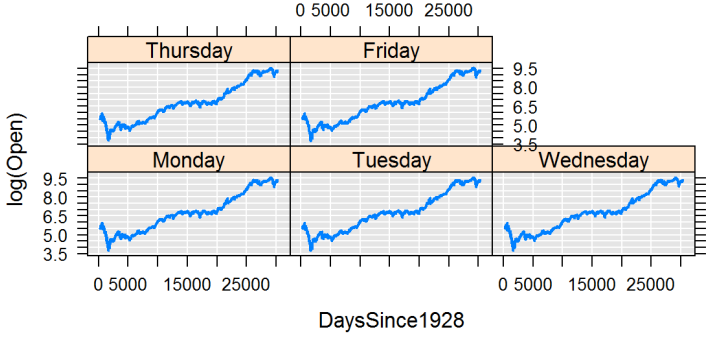
Scatter plot

```
rxLinePlot(Open ~ DaysSince1928, data = DJIAdata)
```



Scatter plot: log(y)

```
rxLinePlot(log(Open) ~ DaysSince1928 | DayOfWeek, data = DJIAdata)
```



Census data

```
censusWorkers <- file.path(sampleDataDir, "CensusWorkers.xdf")
rxGetInfo(censusWorkers, getVarInfo = TRUE)

## File name: C:\Revolution\R-Enterprise-7.2\R-3.0.3\library\RevoScaleR\SampleData\CensusWorkers.xdf
## Number of observations: 351121
## Number of variables: 6
## Number of blocks: 6
## Compression type: zlib
## Variable information:
## Var 1: age, Age
##        Type: integer, Low/High: (20, 65)
## Var 2: incwage, Wage and salary income
##        Type: integer, Low/High: (0, 354000)
## Var 3: perwt, Type: integer, Low/High: (2, 168)
## Var 4: sex, Sex
##        2 factor levels: Male Female
## Var 5: wkswork1, Weeks worked last year
##        Type: integer, Low/High: (21, 52)
## Var 6: state
##        3 factor levels: Connecticut Indiana Washington
```

Census data: perwt

```
rxGetInfo(censusWorkers, numRows = 3)
```

```
## File name: C:\Revolution\R-Enterprise-7.2\R-3.0.3\library\RevoScaleR\SampleData\CensusWorkers.xdf
## Number of observations: 351121
## Number of variables: 6
## Number of blocks: 6
## Compression type: zlib
## Data (3 rows starting with row 1):
##   age incwage perwt    sex wkswork1   state
## 1  23   22000    39 Female       45 Indiana
## 2  25   18000    27 Female       52 Indiana
## 3  43   12000    24 Female       52 Indiana
```

CrossTabs: sum with weights

```
censusCTabs <- rxCrossTabs(wkswork1 ~ F(age):sex, data = censusWorkers,
  fweights = "perwt")

## wkswork1 (sums):
##       sex
## F_age    Male  Female
##    20 3149071 2902334
##    21 3221508 2989308
##    22 3500479 3014440
##    23 3724265 3221561
##    24 3868281 3315162
##    25 3874528 3328932
##    26 3913464 3141570
##    27 4214677 3433142
##    28 4630205 3715891
##    29 4991119 4069394
##    30 5102476 4033802
...
```

CrossTabs: mean with weights

```
censusCTabs <- rxCrossTabs(wkswork1 ~ F(age):sex, data = censusWorkers,
  fweights = "perwt", means = TRUE)

## wkswork1 (means):
##       sex
## F_age  Male Female
##    20 44.30  43.89
##    21 45.28  44.45
##    22 46.07  45.10
##    23 46.75  45.69
##    24 47.51  46.74
##    25 48.26  47.15
##    26 48.18  47.50
##    27 48.60  47.57
##    28 48.90  47.99
##    29 49.25  48.09
##    30 49.42  47.90
...
```

rxCube

```
cube <- rxCube(wkswork1 ~ F(age):sex, data = censusWorkers, fweights = "perwt",
  means = TRUE)
```

```
##    F_age sex    wkswork1 Counts
## 1  20    Male   44.30     71089
## 2  21    Male   45.28     71150
## 3  22    Male   46.07     75979
## 4  23    Male   46.75     79663
## 5  24    Male   47.51     81412
## 6  25    Male   48.26     80288
## 7  26    Male   48.18     81227
## 8  27    Male   48.60     86728
## 9  28    Male   48.90     94681
## 10 29    Male   49.25    101341
## 11 30    Male   49.42    103242
## 12 31    Male   49.54    100234
## 13 32    Male   49.41     96325
...
```

rxCube to data.frame

```
class(cube)
```

```
## [1] "rxCube" "list"
```

```
cubeDF <- as.data.frame(cube)
head(cubeDF)
```

```
##   F_age  sex wkswork1 Counts
## 1    20 Male    44.30  71089
## 2    21 Male    45.28  71150
## 3    22 Male    46.07  75979
## 4    23 Male    46.75  79663
## 5    24 Male    47.51  81412
## 6    25 Male    48.26  80288
```

REVOLUTION ANALYTICS

Thank you

Revolution Analytics is the leading commercial provider of software and support for the popular open source R statistics language.

www.revolutionanalytics.com, 1.855.GET.REVO, Twitter: @RevolutionR

REVOLUTION ANALYTICS

Big data analytics with RRE: Data Manipulation

Sample Directory

```
sampleDataDir <- rxOptions()[["sampleDataDir"]]
list.files(path = sampleDataDir, pattern = "mortDefaultSmall2.*.csv")
```

```
##  [1] "mortDefaultSmall2000.csv"
##  [2] "mortDefaultSmall2001.csv"
##  [3] "mortDefaultSmall2002.csv"
##  [4] "mortDefaultSmall2003.csv"
##  [5] "mortDefaultSmall2004.csv"
##  [6] "mortDefaultSmall2005.csv"
##  [7] "mortDefaultSmall2006.csv"
##  [8] "mortDefaultSmall2007.csv"
##  [9] "mortDefaultSmall2008.csv"
## [10] "mortDefaultSmall2009.csv"
```

Import Setup

```
mort2001 <- file.path(sampleDataDir, "mortDefaultSmall2001.csv")
workDir <- "../data"
mort2001Xdf <- file.path(workDir, "mortDefaultSmall2001.xdf")
```

REVOLUTION ANALYTICS

Import 1 Year of Mortgage Data

```
system.time(rxImport(inData = mort2001, outFile = mort2001Xdf,
  overwrite = TRUE))

##    user  system elapsed
##    0.45    0.06    0.55
```

REVOLUTION ANALYTICS

Mortgage variables

```
rxGetInfo(mort2001Xdf, getVarInfo = TRUE)

## File name: E:\BigData\DataCamp_DataManipulation\data\mortDefaultSmall2001.xdf
## Number of observations: 10000
## Number of variables: 6
## Number of blocks: 1
## Compression type: zlib
## Variable information:
## Var 1: creditScore, Type: integer, Low/High: (513, 890)
## Var 2: houseAge, Type: integer, Low/High: (0, 40)
## Var 3: yearsEmploy, Type: integer, Low/High: (0, 13)
## Var 4: ccDebt, Type: integer, Low/High: (0, 13384)
## Var 5: year, Type: integer, Low/High: (2001, 2001)
## Var 6: default, Type: integer, Low/High: (0, 1)
```

REVOLUTION ANALYTICS

Import all data

```
mortData <- file.path(workDir, "mortgages.xdf")
firstYear <- TRUE
AppendType = "none"
for (iYear in 2000:2009) {
  csvBaseName <- paste("mortDefaultSmall", iYear, ".csv",
    sep = "")
  csvFileName <- file.path(sampleDataDir, csvBaseName)
  rxImport(inData = csvFileName, outFile = mortData, append = AppendType,
    overwrite = firstYear)
  firstYear <- FALSE
  AppendType = "rows"
}
```

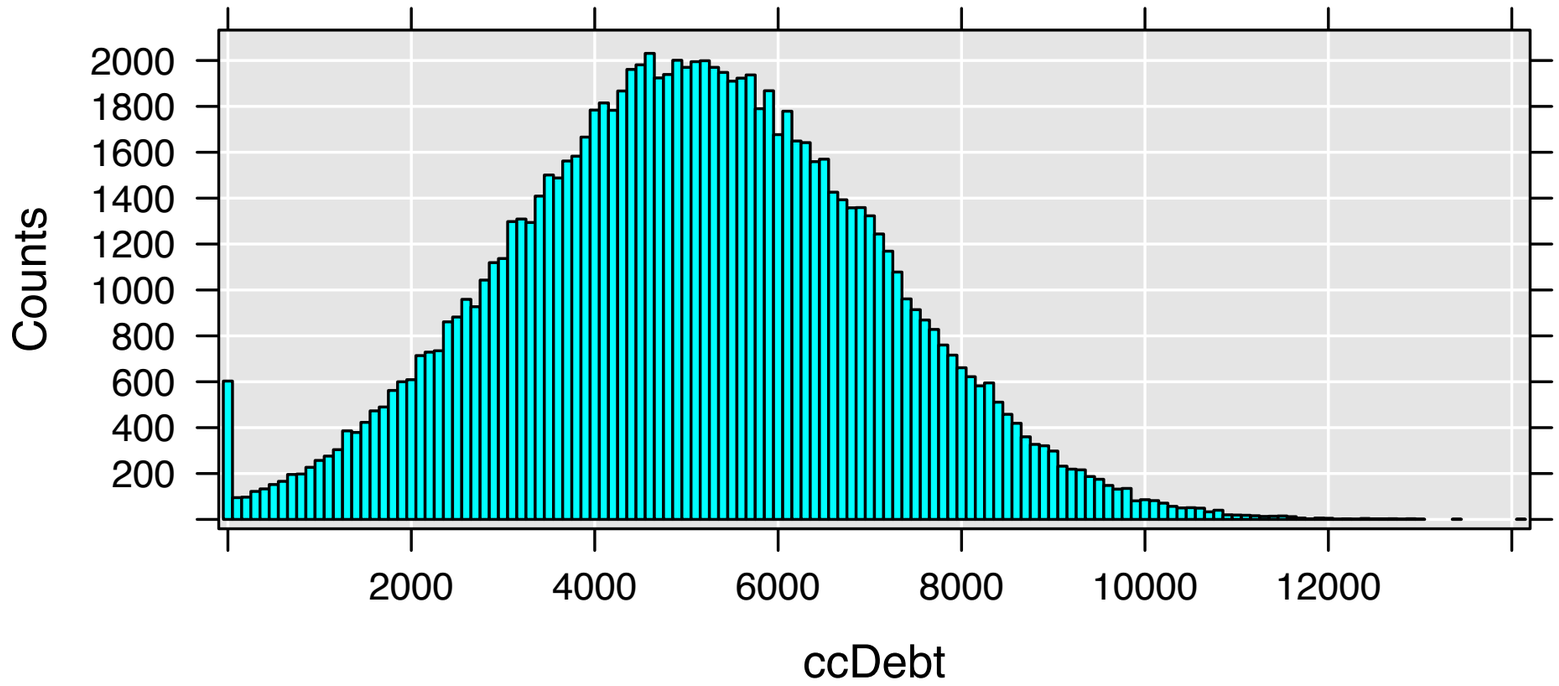
REVOLUTION ANALYTICS

Inspect XDF

```
rxGetInfo(mortData, getVarInfo = TRUE)

## File name: E:\BigData\DataCamp_DataManipulation\data\mortgages.xdf
## Number of observations: 100000
## Number of variables: 6
## Number of blocks: 10
## Compression type: zlib
## Variable information:
## Var 1: creditScore, Type: integer, Low/High: (470, 925)
## Var 2: houseAge, Type: integer, Low/High: (0, 40)
## Var 3: yearsEmploy, Type: integer, Low/High: (0, 14)
## Var 4: ccDebt, Type: integer, Low/High: (0, 14094)
## Var 5: year, Type: integer, Low/High: (2000, 2009)
## Var 6: default, Type: integer, Low/High: (0, 1)
```

Inspect ccDebt

```
rxHistogram(~ccDebt, data = mortData, xNumTicks = 10)
```



Simple Example

```
myData <- data.frame(x = rnorm(100), y = rnorm(100) + 4,
  z = rep(c("sdff", "hgdd"), 50))
summary(myData)
```

```
##        x                   y           z
##  Min.   :-2.99309   Min.   :1.975   hgdd:50
##  1st Qu.:-0.61669   1st Qu.:3.409   sdff:50
##  Median : 0.08980   Median :3.931
##  Mean   : 0.03251   Mean   :3.913
##  3rd Qu.: 0.66156   3rd Qu.:4.462
##  Max.   : 2.28665   Max.   :6.702
```

data.frame to XDF

```
rxDataFrameToXdf(data = myData, outFile = "hello.xdf", overwrite = TRUE)
rxGetInfo(data = "hello.xdf", getVarInfo = TRUE)

## File name: E:\BigData\DataCamp_DataManipulation\doc\hello.xdf
## Number of observations: 100
## Number of variables: 3
## Number of blocks: 1
## Compression type: zlib
## Variable information:
## Var 1: x, Type: numeric, Low/High: (-2.9931, 2.2866)
## Var 2: y, Type: numeric, Low/High: (1.9753, 6.7019)
## Var 3: z
##        2 factor levels: hgdd sdff
```

REVOLUTION ANALYTICS

Summary

```
print(rxSummary(~., data = "hello.xdf"), header = FALSE)
```

```
##  Name Mean       StdDev    Min       Max      ValidObs
##  x    0.03251482 1.0413570 -2.993090 2.286645 100
##  y    3.91251629 0.9041735  1.975322 6.701891 100
##  MissingObs
##  0
##  0
##
## Category Counts for z
## Number of categories: 2
## Number of valid observations: 100
## Number of missing observations: 0
##
##  z    Counts
##  hgdd 50
##  sdff 50
```

rxDataStep

```
args(rxDataStep)

## function (inData = NULL, outFile = NULL, varsToKeep = NULL, varsToDrop = NULL,
##     rowSelection = NULL, transforms = NULL, transformObjects = NULL,
##     transformFunc = NULL, transformVars = NULL, transformPackages = NULL,
##     transformEnvir = NULL, append = "none", overwrite = FALSE,
##     removeMissings = FALSE, computeLowHigh = TRUE, maxRowsByCols = 3000000,
##     rowsPerRead = -1, startRow = 1, numRows = -1, returnTransformObjects = FALSE,
##     blocksPerRead = rxGetOption("blocksPerRead"),
##     reportProgress = rxGetOption("reportProgress"),
##     xdfCompressionLevel = rxGetOption("xdfCompressionLevel"),
##     ...)
## NULL
```

Two Options for Transformations:

- `Transforms` is an ‘inline’ list
- `TransformFunc` allows you to write a whole function

Transforms Example

```
rxDataStep(inData=mortData, outFile=mortData,
           transforms=list(experiencedWorker = yearsEmploy>=5),
           append="cols", overwrite=TRUE)
rxGetVarInfo(data=mortData)

## Var 1: creditScore, Type: integer, Low/High: (470, 925)
## Var 2: houseAge, Type: integer, Low/High: (0, 40)
## Var 3: yearsEmploy, Type: integer, Low/High: (0, 14)
## Var 4: ccDebt, Type: integer, Low/High: (0, 14094)
## Var 5: year, Type: integer, Low/High: (2000, 2009)
## Var 6: default, Type: integer, Low/High: (0, 1)
## Var 7: experiencedWorker, Type: logical, Low/High: (0, 1)
## Var 8: sclCreditScore, Type: numeric, Low/High: (0.0000, 1.0000)
## Var 9: moreThanHalf, Type: logical, Low/High: (0, 1)
## Var 10: newConst, Type: numeric, Low/High: (23.0000, 23.0000)
```

Transforms Example 2

Using `varsToKeep` saves time (and potentially disk space)

```
rxDataStep(inData=mortData, outFile="hello2.xdf", varsToKeep="yearsEmploy",
           transforms=list(experiencedWorker=yearsEmploy>=5),
           append="none", overwrite=TRUE)
rxGetVarInfo(data="hello2.xdf")

## Var 1: yearsEmploy, Type: integer, Low/High: (0, 14)
## Var 2: experiencedWorker, Type: logical, Low/High: (0, 1)
```

Simple Transform Examples

- Simple Combinations of variables (e.g. `a + b`)
- Logical Categorizations

Transform functions

Used for more elaborate transformations

- Complicated recoding and rules
- Merging in data (even though we have a function for that)
- Reshaping data
- Any thing that can be calculated on the data one chunk at a time

Global Min and Max of Credit Score

```
rxGetVarInfo(mortData)$creditScore

## Type: integer, Low/High: (470, 925)

names(rxGetVarInfo(mortData)$creditScore)

## [1] "varType" "storage" "low"     "high"

minCS <- rxGetVarInfo(mortData)$creditScore$low
maxCS <- rxGetVarInfo(mortData)$creditScore$high
```

Define a `transformFunc`:

```
simplyScale <- function(dataList) {
  dataList[["sclCreditScore"]] <- as.numeric((dataList[["creditScore"]] -
    minCreditScore)/(maxCreditScore - minCreditScore))
  dataList[["moreThanHalf"]] <- dataList[["sclCreditScore"]] >
    0.5
  dataList[["newConst"]] <- rep(23, length.out = length(dataList[[1]]))
  return(dataList)
}
```

Applying the transformFunc

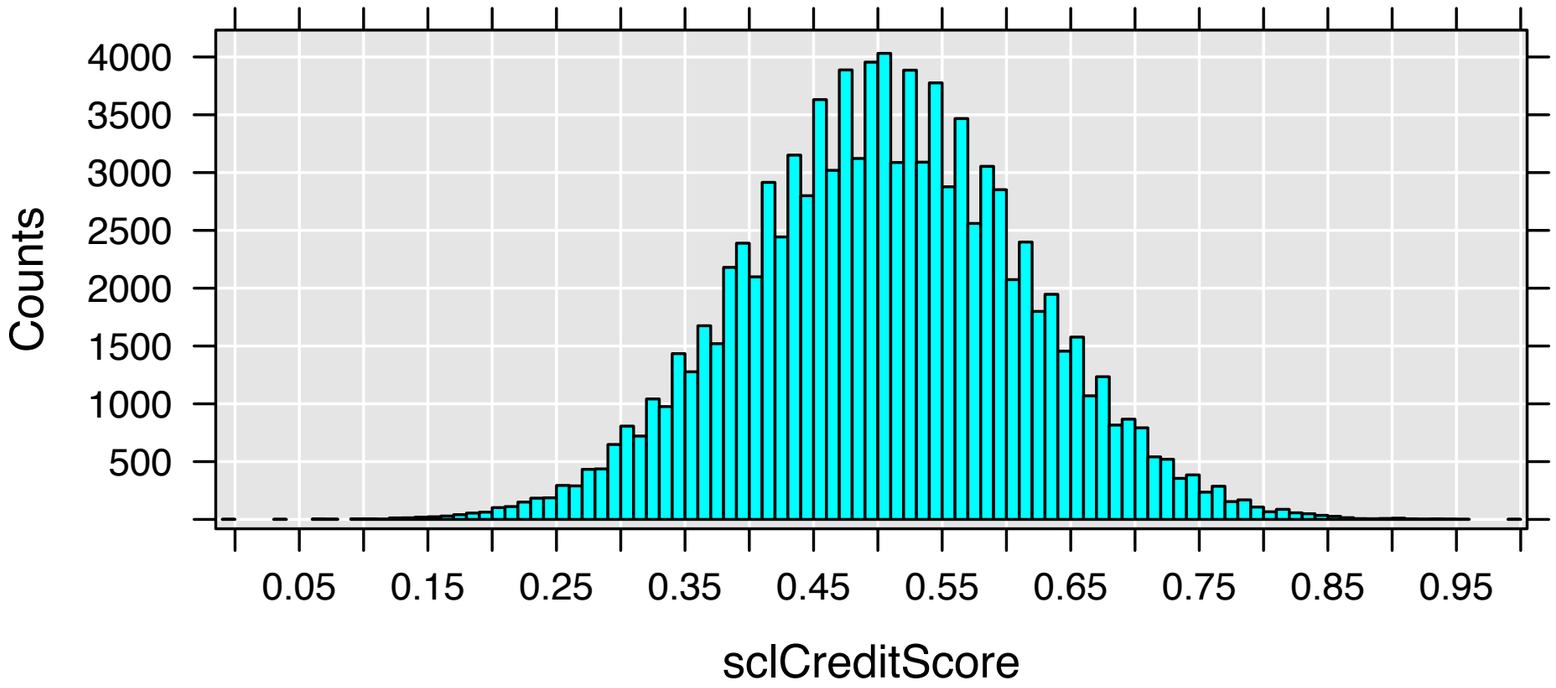
```
rxDataStep(inData = mortData, outFile = mortData, transformFunc = simplyScale,
  transformObjects = list(minCreditScore = minCS, maxCreditScore = maxCS),
  append = "cols", overwrite = TRUE)
```

View the Transformed Variable

```
rxGetInfo(data = mortData, getVarInfo = TRUE, varsToKeep = c("sclCreditScore",
  "moreThanHalf", "newConst"))

## File name: E:\BigData\DataCamp_DataManipulation\xdf\mortgages.xdf
## Number of observations: 100000
## Number of variables: 10
## Number of blocks: 10
## Compression type: zlib
## Variable information:
## Var 1: sclCreditScore, Type: numeric, Low/High: (0.0000, 1.0000)
## Var 2: moreThanHalf, Type: logical, Low/High: (0, 1)
## Var 3: newConst, Type: numeric, Low/High: (23.0000, 23.0000)
```

Histogram of the Transformed Variable

```
rxHistogram(~sclCreditScore, data = mortData, xNumTicks = 15)
```



Bad transformFunc

```
wrongScale <- function(dataList) {
  dataList[["sclCreditScore2"]] <- (dataList[["creditScore"]] -
    min(dataList[["creditScore"]]))/(max(dataList[["creditScore"]],
    na.rm = TRUE) - min(dataList[["creditScore"]], na.rm = TRUE))
  dataList[["sclDiff"]] <- dataList[["sclCreditScore2"]] -
    dataList[["sclCreditScore"]]
  return(dataList)
}
```

REVOLUTION ANALYTICS

Apply the Wrong Scaling

```
mortData2 <- file.path(workDir, "mortgagesTemp.xdf")
rxDataStep(inData = mortData, outFile = mortData2, transformFunc = wrongScale,
  overwrite = TRUE)
rxGetInfo(mortData2, getVarInfo = TRUE, varsToKeep = c("sclCreditScore",
  "sclCreditScore2"))

## File name: E:\BigData\DataCamp_DataManipulation\xdf\mortgagesTemp.xdf
## Number of observations: 100000
## Number of variables: 12
## Number of blocks: 10
## Compression type: zlib
## Variable information:
## Var 1: sclCreditScore, Type: numeric, Low/High: (0.0000, 1.0000)
## Var 2: sclCreditScore2, Type: numeric, Low/High: (0.0000, 1.0000)
```

Summarize the Wrong Scaling

```
print(rxSummary(~sclCreditScore + sclCreditScore2 + sclDiff,
  data = mortData2), header = FALSE)
```

```
##  Name             Mean          StdDev    Min
##  sclCreditScore    0.505242725 0.1102388  0.0000000
##  sclCreditScore2   0.504419083 0.1318656  0.0000000
##  sclDiff          -0.000823642 0.0321998 -0.1252747
##  Max        ValidObs MissingObs
##  1.0000000 100000    0
##  1.0000000 100000    0
##  0.1164835 100000    0
```

Not the same summary statistics!

REVOLUTION ANALYTICS

Thank you

Revolution Analytics is the leading commercial provider of software and support for the popular open source R statistics language.

www.revolutionanalytics.com
1.855.GET.REVO
Twitter: @RevolutionR

REVOLUTION ANALYTICS

Big data analytics with RRE: Modeling

Sample Data

```
sampleDataDir <- rxGetOption("sampleDataDir")
list.files(path = sampleDataDir, pattern = "*.csv")
```

```
##  [1] "AirlineDemo1kNoMissing.csv" "AirlineDemoSmall.csv"
##  [3] "mortDefaultSmall2000.csv"   "mortDefaultSmall2001.csv"
##  [5] "mortDefaultSmall2002.csv"   "mortDefaultSmall2003.csv"
##  [7] "mortDefaultSmall2004.csv"   "mortDefaultSmall2005.csv"
##  [9] "mortDefaultSmall2006.csv"   "mortDefaultSmall2007.csv"
## [11] "mortDefaultSmall2008.csv"   "mortDefaultSmall2009.csv"
```

REVOLUTION ANALYTICS

Import Setup

```
# identify input file
inputFile <- file.path(sampleDataDir, "AirlineDemoSmall.csv")
# identify output file
airDS <- "xdf/ADS.xdf"
# Make a factor variable while reading in the file
colInfo <- list(DayOfWeek = list(type = "factor", levels = c("Monday", "Tuesday", "Wednesday",
  "Thursday", "Friday", "Saturday", "Sunday")))
```

Import the Data

```
# Import the file
system.time(rxImport(inData = inputFile, outFile = airDS, colInfo = colInfo, missingValueString = "M",
  overwrite = TRUE))

##    user  system elapsed
##    1.63    0.08    1.78
```

REVOLUTION ANALYTICS

Examine the data

```
rxGetInfo(airDS, getVarInfo = TRUE)

## File name: E:\GitHub\Revolution_Course_Materials\modules\BigData\DataCamp_Analysis\doc\xdf\ADS.xdf
## Number of observations: 600000
## Number of variables: 3
## Number of blocks: 2
## Compression type: zlib
## Variable information:
## Var 1: ArrDelay, Type: integer, Low/High: (-86, 1490)
## Var 2: CRSDepTime, Type: numeric, Storage: float32, Low/High: (0.0167, 23.9833)
## Var 3: DayOfWeek
##        7 factor levels: Monday Tuesday Wednesday Thursday Friday Saturday Sunday
```

REVOLUTION ANALYTICS

Summarize Each Variable

```
print(rxSummary(~., data = airDS), header = FALSE)
```

```
##  Name        Mean     StdDev    Min         Max         ValidObs MissingObs
##  ArrDelay    11.31794 40.688536 -86.000000 1490.00000 582628   17372
##  CRSDepTime 13.48227  4.697566   0.016667   23.98333 600000        0
##
## Category Counts for DayOfWeek
## Number of categories: 7
## Number of valid observations: 600000
## Number of missing observations: 0
##
##  DayOfWeek Counts
##  Monday    97975
##  Tuesday   77725
##  Wednesday 78875
##  Thursday  81304
##  Friday    82987
##  Saturday  86159
##  Sunday    94975
```

REVOLUTION ANALYTICS

Summarize by Levels

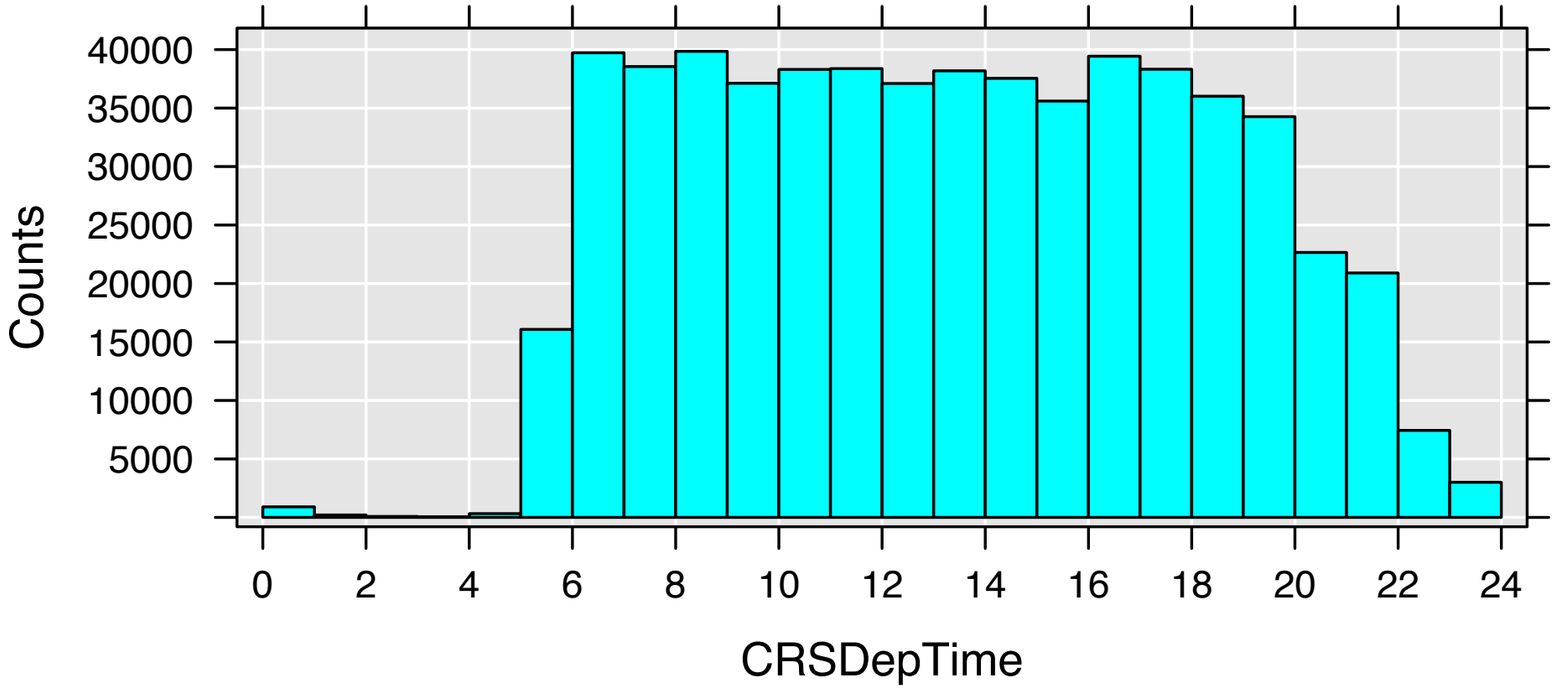
```
print(delaySumm <- rxSummary(ArrDelay ~ DayOfWeek, data = airDS), header = FALSE)

##  Name                Mean     StdDev   Min Max  ValidObs MissingObs
##  ArrDelay:DayOfWeek 11.31794 40.68854 -86 1490 582628   17372
##
## Statistics by category (7 categories):
##
##  Category                           DayOfWeek Means     StdDev   Min Max  ValidObs
##  ArrDelay for DayOfWeek=Monday      Monday    12.025604 40.02463 -76 1017 95298
##  ArrDelay for DayOfWeek=Tuesday     Tuesday   11.293808 43.66269 -70 1143 74011
##  ArrDelay for DayOfWeek=Wednesday   Wednesday 10.156539 39.58803 -81 1166 76786
##  ArrDelay for DayOfWeek=Thursday    Thursday   8.658007 36.74724 -58 1053 79145
##  ArrDelay for DayOfWeek=Friday      Friday    14.804335 41.79260 -78 1490 80142
##  ArrDelay for DayOfWeek=Saturday    Saturday  11.875326 45.24540 -73 1370 83851
##  ArrDelay for DayOfWeek=Sunday      Sunday    10.331806 37.33348 -86 1202 93395
```

Compute Means

```
delayCube <- rxCube(ArrDelay ~ DayOfWeek, data = airDS, means = TRUE)
data.frame(delaySumm$categorical[[1]][, c("DayOfWeek", "Means")], delayCube$ArrDelay)

##   DayOfWeek     Means delayCube.ArrDelay
## 1    Monday 12.025604          12.025604
## 2   Tuesday 11.293808          11.293808
## 3 Wednesday 10.156539          10.156539
## 4  Thursday  8.658007           8.658007
## 5    Friday 14.804335          14.804335
## 6  Saturday 11.875326          11.875326
## 7    Sunday 10.331806          10.331806
```

Examine a Distribution of Departure Time

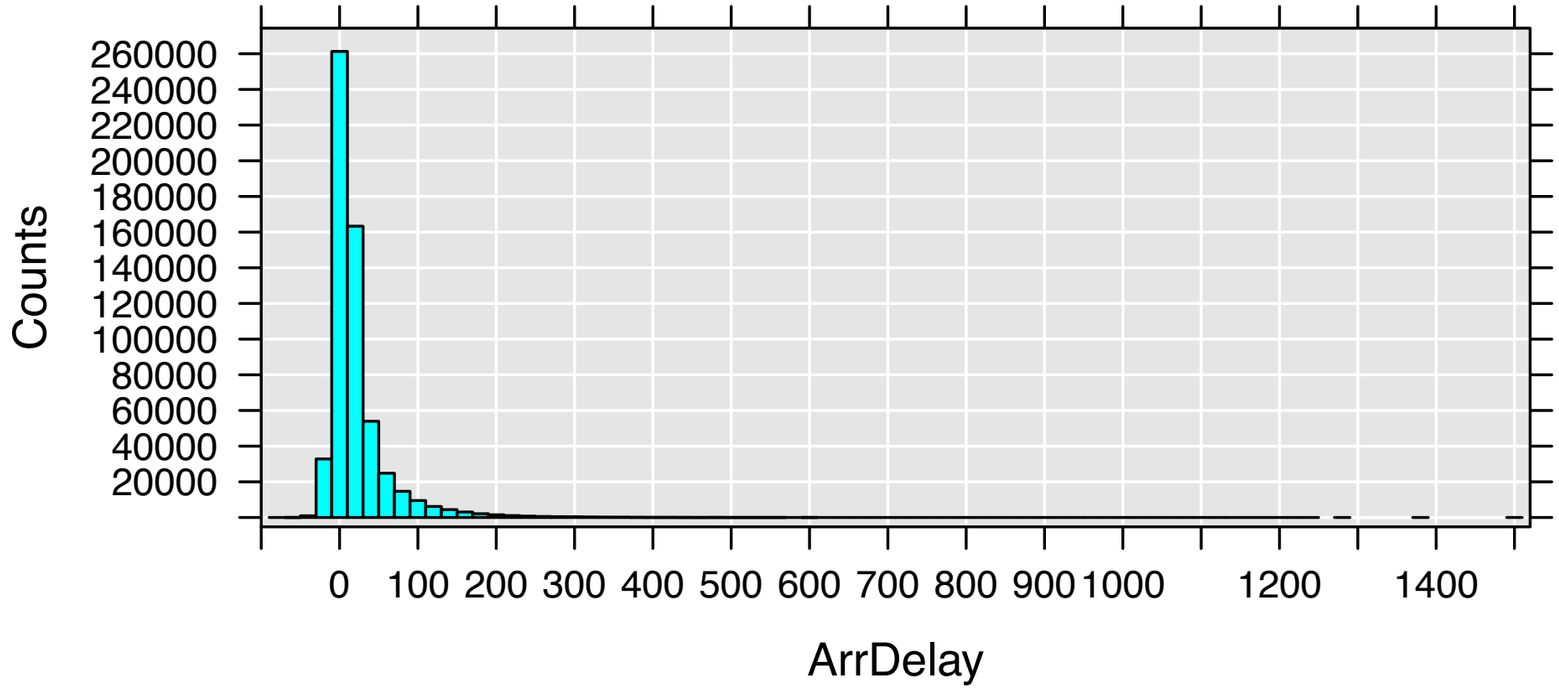
```
rxHistogram(~CRSDepTime, data = airDS, numBreaks = 25, xNumTicks = 13)
```



REVOLUTION ANALYTICS

Examine a Distribution of Arrival Delay

```
rxHistogram(~ArrDelay, data = airDS, xNumTicks = 16)
```



REVOLUTION ANALYTICS

Predict Arrival Delay by Day of Week

```
system.time(arrDelayLm1 <- rxLinMod(ArrDelay ~ DayOfWeek, data = airDS))

##    user  system elapsed
##    0.06    0.00    0.11
```

REVOLUTION ANALYTICS

Model Summary

```
summary(arrDelayLm1)

## Call:
## rxLinMod(formula = ArrDelay ~ DayOfWeek, data = airDS)
##
## Linear Regression Results for: ArrDelay ~ DayOfWeek
## Data: airDS (RxXdfData Data Source)
## File name: xdf/ADS.xdf
## Dependent variable(s): ArrDelay
## Total independent variables: 8 (Including number dropped: 1)
## Number of valid observations: 582628
## Number of missing observations: 17372
##
## Coefficients: (1 not defined because of singularities)
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          10.3318     0.1330  77.673 2.22e-16 ***
## DayOfWeek=Monday      1.6938     0.1872   9.049 2.22e-16 ***
## DayOfWeek=Tuesday     0.9620     0.2001   4.809 1.52e-06 ***
## DayOfWeek=Wednesday  -0.1753     0.1980  -0.885    0.376
## DayOfWeek=Thursday   -1.6738     0.1964  -8.522 2.22e-16 ***
## DayOfWeek=Friday      4.4725     0.1957  22.850 2.22e-16 ***
## DayOfWeek=Saturday    1.5435     0.1934   7.981 2.22e-16 ***
## DayOfWeek=Sunday     Dropped    Dropped Dropped  Dropped
## ---
```

Model Summary (continued)

```
print(summary(arrDelayLm1), header = FALSE)

## Coefficients: (1 not defined because of singularities)
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          10.3318     0.1330  77.673 2.22e-16 ***
## DayOfWeek=Monday      1.6938     0.1872   9.049 2.22e-16 ***
## DayOfWeek=Tuesday     0.9620     0.2001   4.809 1.52e-06 ***
## DayOfWeek=Wednesday  -0.1753     0.1980  -0.885    0.376
## DayOfWeek=Thursday   -1.6738     0.1964  -8.522 2.22e-16 ***
## DayOfWeek=Friday      4.4725     0.1957  22.850 2.22e-16 ***
## DayOfWeek=Saturday    1.5435     0.1934   7.981 2.22e-16 ***
## DayOfWeek=Sunday     Dropped    Dropped Dropped  Dropped
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 40.65 on 582621 degrees of freedom
## Multiple R-squared: 0.001869
## Adjusted R-squared: 0.001858
## F-statistic: 181.8 on 6 and 582621 DF,  p-value: < 2.2e-16
## Condition number: 10.5595
```

Argument: dropFirst=TRUE

```
arrDelayLm1df <- rxLinMod(ArrDelay ~ DayOfWeek, data = airDS, dropFirst = TRUE)
```

REVOLUTION ANALYTICS

Results: dropFirst=TRUE

```
coef(summary(arrDelayLm1df)[[1]])
```

```
##                          Estimate Std. Error     t value     Pr(>|t|)
## (Intercept)          12.0256039  0.1316820  91.3230913 2.220446e-16
## DayOfWeek=Monday             NA         NA          NA           NA
## DayOfWeek=Tuesday    -0.7317962  0.1991674  -3.6742778 2.385445e-04
## DayOfWeek=Wednesday -1.8690649  0.1971313  -9.4813216 2.220446e-16
## DayOfWeek=Thursday  -3.3675964  0.1954976 -17.2257730 2.220446e-16
## DayOfWeek=Friday     2.7787309  0.1948321  14.2621822 2.220446e-16
## DayOfWeek=Saturday  -0.1502774  0.1924772  -0.7807543 4.349473e-01
## DayOfWeek=Sunday    -1.6937981  0.1871726  -9.0493894 2.220446e-16
```

REVOLUTION ANALYTICS

Argument: cube=TRUE

```
arrDelayLm1cube <- rxLinMod(ArrDelay ~ DayOfWeek, data = airDS, cube = TRUE)
```

REVOLUTION ANALYTICS

Results: cube=TRUE

```
coef(summary(arrDelayLm1cube)[[1]])
```

```
##                       Estimate Std. Error   t value     Pr(>|t|) Counts
## DayOfWeek=Monday     12.025604  0.1316820  91.32309 2.220446e-16  95298
## DayOfWeek=Tuesday    11.293808  0.1494239  75.58234 2.220446e-16  74011
## DayOfWeek=Wednesday 10.156539  0.1466990  69.23386 2.220446e-16  76786
## DayOfWeek=Thursday    8.658007  0.1444962  59.91858 2.220446e-16  79145
## DayOfWeek=Friday     14.804335  0.1435946 103.09813 2.220446e-16  80142
## DayOfWeek=Saturday   11.875326  0.1403829  84.59243 2.220446e-16  83851
## DayOfWeek=Sunday     10.331806  0.1330168  77.67296 2.220446e-16  93395
```

REVOLUTION ANALYTICS

Compare Results to delayCube

```
lm1est <- coef(arrDelayLm1)["(Intercept)"] + c(coef(arrDelayLm1)[-c(1, 8)], 0)
lm1dfest <- coef(arrDelayLm1df)["(Intercept)"] + c(0, coef(arrDelayLm1df)[-c(1, 2)])
lm1cubeest <- coef(arrDelayLm1cube)
data.frame(delayCube[, "ArrDelay", drop = FALSE], lm1est, lm1dfest, lm1cubeest, row.names = delayCube[,
  "DayOfWeek"])

##             ArrDelay    lm1est  lm1dfest lm1cubeest
## Monday     12.025604 12.025604 12.025604  12.025604
## Tuesday    11.293808 11.293808 11.293808  11.293808
## Wednesday 10.156539 10.156539 10.156539  10.156539
## Thursday    8.658007  8.658007  8.658007   8.658007
## Friday     14.804335 14.804335 14.804335  14.804335
## Saturday   11.875326 11.875326 11.875326  11.875326
## Sunday     10.331806 10.331806 10.331806  10.331806
```

REVOLUTION ANALYTICS

Elements of rxLinMod Objects

```
names(arrDelayLm1)
```

```
##  [1] "coefficients"      "residual.squares"  "condition.number"  "rank"
##  [5] "aliased"           "coef.std.error"    "coef.t.value"      "coef.p.value"
##  [9] "total.squares"     "y.var"             "sigma"             "residual.variance"
## [13] "r.squared"         "f.pvalue"          "df"                "y.names"
## [17] "deviance"          "aic"               "params"            "formula"
## [21] "call"              "fstatistics"       "adj.r.squared"     "nValidObs"
## [25] "nMissingObs"       "coefLabelStyle"
```

Additive Models

```
arrDelayLm3 <- rxLinMod(ArrDelay ~ DayOfWeek + CRSDepTime, data = airDS, cube = TRUE)
```

REVOLUTION ANALYTICS

Additive Models: Results

```
coef(summary(arrDelayLm3)[[1]])
```

```
##                         Estimate Std. Error    t value     Pr(>|t|) Counts
## DayOfWeek=Monday    -1.1135823 0.19988456  -5.571127 2.532079e-08  95298
## DayOfWeek=Tuesday   -1.8544247 0.21191691  -8.750716 2.220446e-16  74011
## DayOfWeek=Wednesday -3.0430310 0.21045458 -14.459325 2.220446e-16  76786
## DayOfWeek=Thursday  -4.5175933 0.20874452 -21.641733 2.220446e-16  79145
## DayOfWeek=Friday     1.6058429 0.20832119   7.708495 2.220446e-16  80142
## DayOfWeek=Saturday  -1.0049295 0.20346881  -4.938985 7.855173e-07  83851
## DayOfWeek=Sunday    -3.1945789 0.20412829 -15.649859 2.220446e-16  93395
## CRSDepTime           0.9786211 0.01125522  86.948171 2.220446e-16    NaN
```

Interactive Models

Only the interaction:

```
arrDelayLm4 <- rxLinMod(ArrDelay ~ DayOfWeek:CRSDepTime, data = airDS)
```

Main Effects and Interactions:

```
arrDelayLm4b <- rxLinMod(ArrDelay ~ DayOfWeek * CRSDepTime, data = airDS)
```

REVOLUTION ANALYTICS

Inline Transformations

```
arrDelayLm5 <- rxLinMod(log(abs(ArrDelay)) ~ DayOfWeek:F(CRSDepTime), data = airDS)
```

REVOLUTION ANALYTICS

More Complex Transforms

```
system.time(
  arrDelayLm6 <-
    rxLinMod(ArrDelay ~ DayOfWeek:depTimeCat,
             data = airDS,
               transforms=list(depTimeCat =
                                 cut(CRSDepTime,
                                     breaks=seq(from = 5, to = 23, by = 2)
                                     )
                                 )
             )
)

##    user  system elapsed
##    0.27    0.02    0.28
```

REVOLUTION ANALYTICS

rxLinMod and Text Files: Setup

```
inputDataSource <- RxTextData(file = inputFile, colInfo = colInfo, missingValueString = "M")
```

REVOLUTION ANALYTICS

rxLinMod and Text Files: Run

```
system.time(
    arrDelayLm6b <-
        rxLinMod(ArrDelay ~ DayOfWeek:depTimeCat,
                 data = inputDataSource,
                   transforms=list(depTimeCat =
                                     cut(CRSDepTime,
                                         breaks=seq(from = 5,to = 23, by = 2)
                                         )
                                   )
                 )
)

##    user  system elapsed
##    1.83    0.05    2.16
```

Same procedure afterwards

```
coef(summary(arrDelayLm6b)[[1]])
```

```
##                                              Estimate Std. Error        t value
## (Intercept)                                14.3121890547  0.5812290  24.6240114461
## DayOfWeek=Monday, depTimeCat=(5,7]        -10.6937623767  0.7155505 -14.9448052737
## DayOfWeek=Tuesday, depTimeCat=(5,7]       -10.6104011470  0.7484972 -14.1756058396
## DayOfWeek=Wednesday, depTimeCat=(5,7]     -12.8586943580  0.7479515 -17.1918827866
## DayOfWeek=Thursday, depTimeCat=(5,7]      -14.1515975383  0.7410982 -19.0954418535
## DayOfWeek=Friday, depTimeCat=(5,7]        -10.7489603552  0.7393418 -14.5385526380
## DayOfWeek=Saturday, depTimeCat=(5,7]       -8.5746353524  0.7373944 -11.6282885727
## DayOfWeek=Sunday, depTimeCat=(5,7]        -12.2679527435  0.7567371 -16.2116441288
## DayOfWeek=Monday, depTimeCat=(7,9]         -8.4609244864  0.6833984 -12.3806621672
## DayOfWeek=Tuesday, depTimeCat=(7,9]        -7.4557272978  0.7093253 -10.5110124827
## DayOfWeek=Wednesday, depTimeCat=(7,9]      -9.9322050836  0.7078763 -14.0309904391
## DayOfWeek=Thursday, depTimeCat=(7,9]      -12.1323997827  0.7027991 -17.2629702965
## DayOfWeek=Friday, depTimeCat=(7,9]         -7.6800706037  0.7017865 -10.9436003770
## DayOfWeek=Saturday, depTimeCat=(7,9]       -6.6614208629  0.6894919  -9.6613469219
## DayOfWeek=Sunday, depTimeCat=(7,9]        -12.0232803688  0.6949892 -17.2999521053
## DayOfWeek=Monday, depTimeCat=(9,11]        -4.6174917687  0.6884164  -6.7074109563
## DayOfWeek=Tuesday, depTimeCat=(9,11]       -5.4519443727  0.7193369  -7.5791252249
## DayOfWeek=Wednesday, depTimeCat=(9,11]     -8.6766525561  0.7129235 -12.1705243282
## DayOfWeek=Thursday, depTimeCat=(9,11]     -10.3031073191  0.7087136 -14.5377587653
## DayOfWeek=Friday, depTimeCat=(9,11]        -4.9864834096  0.7085966  -7.0371254536
...
```

REVOLUTION ANALYTICS

rxPredict()

```
args(RevoScaleR:::rxPredict.default)

## function (modelObject, data = NULL, outData = NULL, computeStdErrors = FALSE,
##     interval = "none", confLevel = 0.95, computeResiduals = FALSE,
##     type = c("response", "link"), writeModelVars = FALSE, removeMissings = FALSE,
##     overwrite = FALSE, checkFactorLevels = TRUE, predVarNames = NULL,
##     residVarNames = NULL, intervalVarNames = NULL, stdErrorsVarNames = NULL,
##     predNames = NULL, blocksPerRead = rxGetOption("blocksPerRead"),
##     reportProgress = rxGetOption("reportProgress"), verbose = 0,
##     xdfCompressionLevel = rxGetOption("xdfCompressionLevel"),
##     ...)
## NULL
```

Review the Model

```
system.time(
  arrDelayLm <- rxLinMod(
    ArrDelay ~ DayOfWeek:depTimeCat,
    data = airDS, cube=TRUE,
    transforms=list(depTimeCat =
                      cut(CRSDepTime, breaks=seq(from=5, to=23, by=2))
                    )
    )
  )

##    user  system elapsed
##    0.31    0.00    0.32
```

Model Results

```
summary(arrDelayLm)

## Call:
## rxLinMod(formula = ArrDelay ~ DayOfWeek:depTimeCat, data = airDS,
##     cube = TRUE, transforms = list(depTimeCat = cut(CRSDepTime,
##         breaks = seq(from = 5, to = 23, by = 2))))
##
## Cube Linear Regression Results for: ArrDelay ~ DayOfWeek:depTimeCat
## Data: airDS (RxXdfData Data Source)
## File name: xdf/ADS.xdf
## Dependent variable(s): ArrDelay
## Total independent variables: 63
## Number of valid observations: 578128
## Number of missing observations: 21872
##
## Coefficients:
##                                          Estimate Std. Error t value Pr(>|t|)     | Counts
## DayOfWeek=Monday, depTimeCat=(5,7]         3.6184     0.4174   8.670 2.22e-16 *** |   9356
## DayOfWeek=Tuesday, depTimeCat=(5,7]        3.7018     0.4716   7.849 2.22e-16 *** |   7327
## DayOfWeek=Wednesday, depTimeCat=(5,7]      1.4535     0.4707   3.088  0.00202 **  |   7354
## DayOfWeek=Thursday, depTimeCat=(5,7]       0.1606     0.4598   0.349  0.72688     |   7709
## DayOfWeek=Friday, depTimeCat=(5,7]         3.5632     0.4569   7.798 2.22e-16 *** |   7805
## DayOfWeek=Saturday, depTimeCat=(5,7]       5.7376     0.4538  12.644 2.22e-16 *** |   7914
...
```

Predicting Values

```
rxPredict(modelObject = arrDelayLm, data = airDS, outData = airDS, overwrite = TRUE)
```

REVOLUTION ANALYTICS

View Predictions

```
rxGetVarInfo(airDS)

## Var 1: ArrDelay, Type: integer, Low/High: (-86, 1490)
## Var 2: CRSDepTime, Type: numeric, Storage: float32, Low/High: (0.0167, 23.9833)
## Var 3: DayOfWeek
##        7 factor levels: Monday Tuesday Wednesday Thursday Friday Saturday Sunday
## Var 4: urv, Type: numeric, Low/High: (0.0000, 1.0000)
## Var 5: TrainTest
##        2 factor levels: Train Test
## Var 6: ArrDelay_Pred, Type: numeric, Low/High: (0.1606, 25.1872)
```

REVOLUTION ANALYTICS

Compute Residuals

```
rxPredict(modelObject = arrDelayLm, data = airDS, outData = airDS, computeResiduals = TRUE,
  overwrite = TRUE)
```

REVOLUTION ANALYTICS

View Residuals

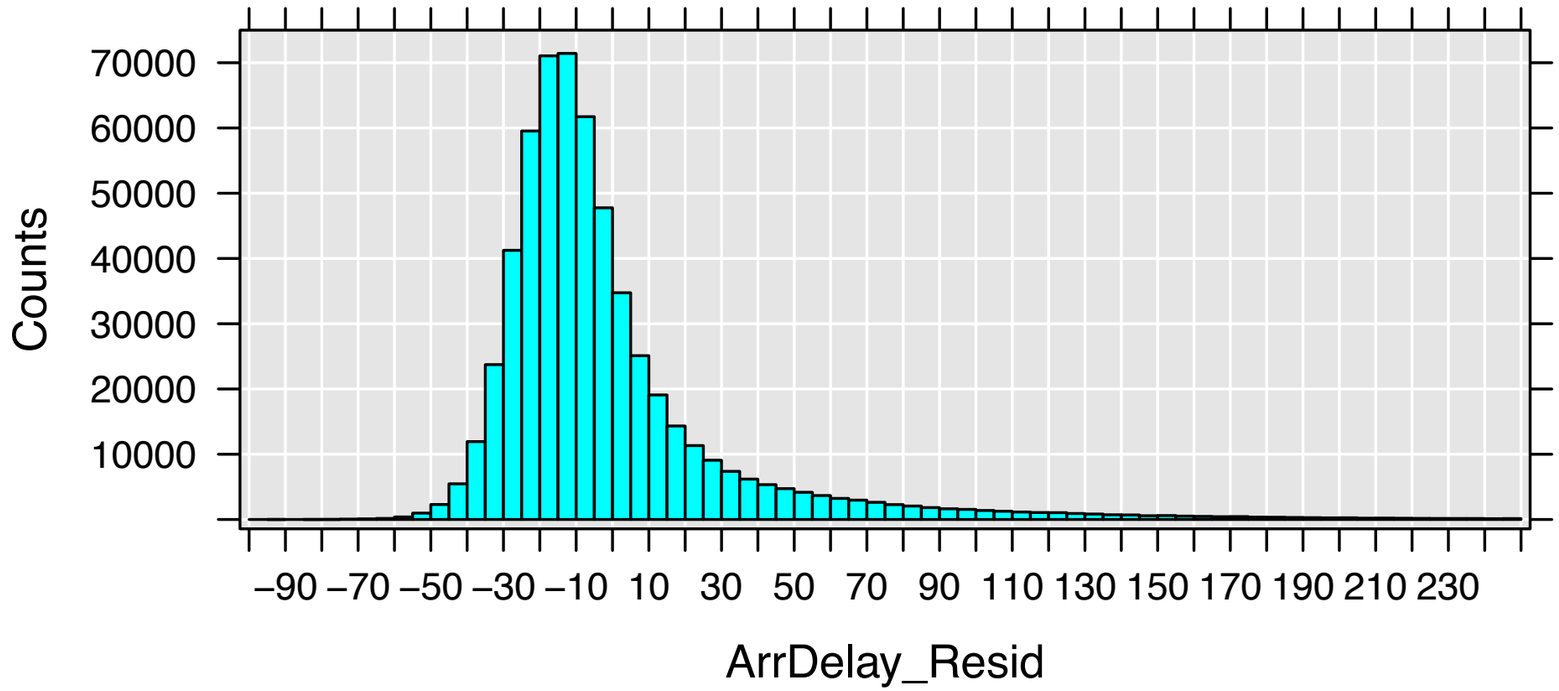
```
rxGetVarInfo(airDS)

## Var 1: ArrDelay, Type: integer, Low/High: (-86, 1490)
## Var 2: CRSDepTime, Type: numeric, Storage: float32, Low/High: (0.0167, 23.9833)
## Var 3: DayOfWeek
##        7 factor levels: Monday Tuesday Wednesday Thursday Friday Saturday Sunday
## Var 4: urv, Type: numeric, Low/High: (0.0000, 1.0000)
## Var 5: TrainTest
##        2 factor levels: Train Test
## Var 6: ArrDelay_Pred, Type: numeric, Low/High: (0.1606, 25.1872)
## Var 7: ArrDelay_Resid, Type: numeric, Low/High: (-97.6677, 1486.4368)
```

Examine the Distribution of Residuals

```
rxHistogram(~ArrDelay_Resid, data = airDS)
```

Examine the Distribution in More Detail

```
rxHistogram(~ArrDelay_Resid, data = airDS, numBreaks = 100, startVal = -100, endVal = 250)
```



REVOLUTION ANALYTICS

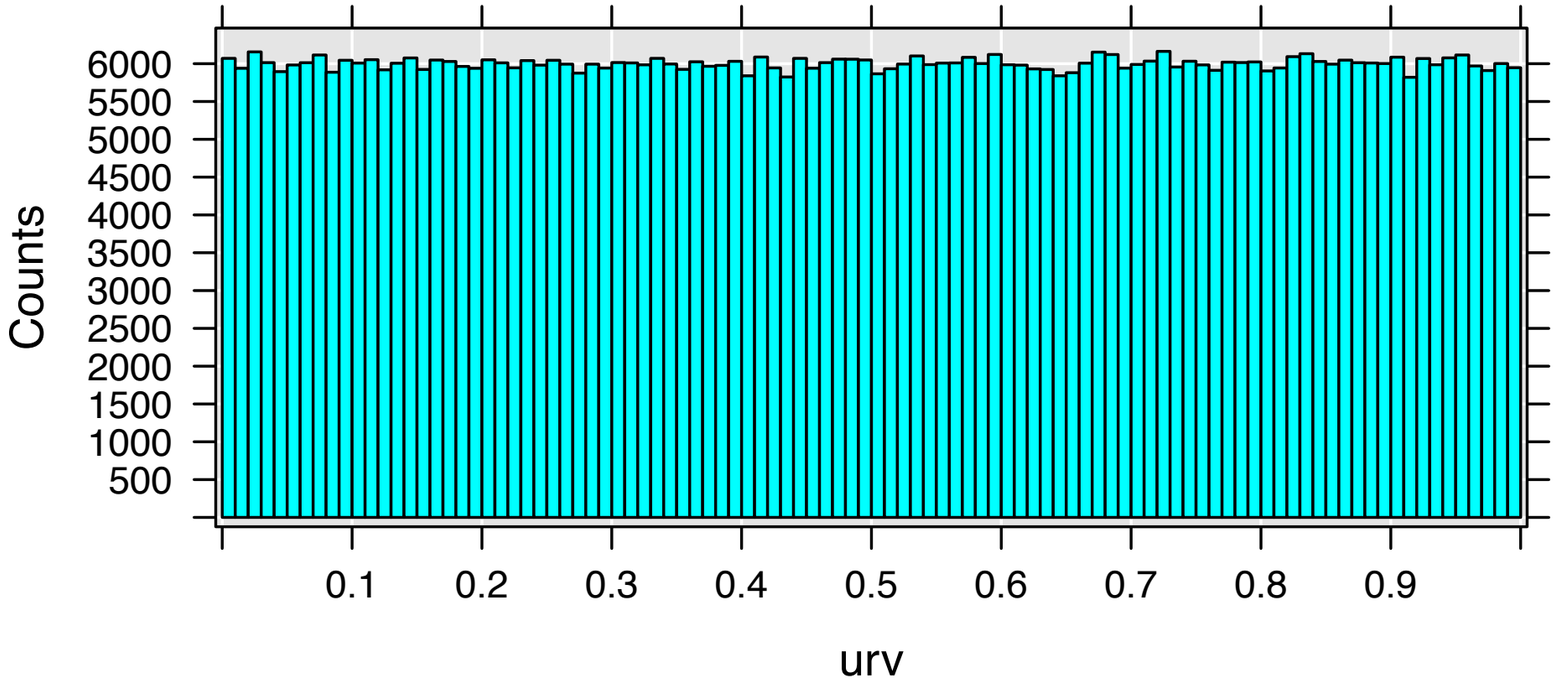
Training and Validation Setup

```
rxDataStep(inData = airDS, outFile = airDS, transforms = list(urv = runif(length(DayOfWeek))),
  append = "cols", overwrite = TRUE)
```

REVOLUTION ANALYTICS

URV Visualization

```
rxHistogram(~urv, airDS, xNumTicks = 10)
```



Converting urv to a Factor Variable

```
rxDataStep(inData = airDS, outFile = airDS,
           transforms=list(TrainTest = cut(urv, breaks = c(0,0.8,1),
                                           labels = c("Train", "Test"))),
           append = "cols", overwrite = TRUE)
```

REVOLUTION ANALYTICS

View TrainTest

```
rxSummary(~TrainTest, data = airDS)
```

```
## Call:
## rxSummary(formula = ~TrainTest, data = airDS)
##
## Summary Statistics Results for: ~TrainTest
## Data: airDS (RxXdfData Data Source)
## File name: xdf/ADS.xdf
## Number of valid observations: 600000
##
##
## Category Counts for TrainTest
## Number of categories: 2
## Number of valid observations: 600000
## Number of missing observations: 0
##
##  TrainTest Counts
##  Train     479854
##  Test      120146
```

Splitting the dataset

```
mySplit <- rxSplit(inData = airDS, splitByFactor = "TrainTest",
                   rowSelection = ArrDelay < 450,
                   overwrite = TRUE)
```

REVOLUTION ANALYTICS

View the New File Names

```
names(mySplit)

## [1] "E:\\BigData\\DataCamp_Analysis\\doc\\ADS.TrainTest.Train.xdf"
## [2] "E:\\BigData\\DataCamp_Analysis\\doc\\ADS.TrainTest.Test.xdf"


trainDS <- mySplit[[1]]
testDS <- mySplit[[2]]
```

Create Training Model

```
system.time(
  arrDelayLm.train <- rxLinMod(
    ArrDelay ~ DayOfWeek:depTimeCat,
    data = trainDS,
    cube=TRUE,
    transforms=list(
      depTimeCat = cut(CRSDepTime, breaks=seq(from=5, to=23, by=2)))
    )
  )

##    user  system elapsed
##    0.24    0.02    0.25
```

REVOLUTION ANALYTICS

Cross-Validate Model

```
rxPredict(modelObject = arrDelayLm.train, data = testDS, outData = testDS, computeResiduals = TRUE,
  overwrite = TRUE)
rxGetVarInfo(testDS)

## Var 1: ArrDelay, Type: integer, Low/High: (-86, 1490)
## Var 2: CRSDepTime, Type: numeric, Low/High: (0.0167, 23.9833)
## Var 3: DayOfWeek
##        7 factor levels: Monday Tuesday Wednesday Thursday Friday Saturday Sunday
## Var 4: urv, Type: numeric, Low/High: (0.0000, 1.0000)
## Var 5: TrainTest
##        2 factor levels: Train Test
## Var 6: ArrDelay_Pred, Type: numeric, Low/High: (0.0812, 25.1872)
## Var 7: ArrDelay_Resid, Type: numeric, Low/High: (-97.6677, 1486.4368)
```

Correlate Predicted and Actual Values

```
(pred.act.cors <- rxCor(formula = ~ArrDelay_Pred + ArrDelay, data = testDS))
```

```
##                ArrDelay_Pred  ArrDelay
## ArrDelay_Pred      1.0000000 0.1358719
## ArrDelay           0.1358719 1.0000000
```

```
arrDelayLm.train$r.squared
```

```
## [1] 0.02078747
```

```
pred.act.cors[1, 2]^2
```

```
## [1] 0.01846116
```

REVOLUTION ANALYTICS

Computing Std. Errors

```
try(rxPredict(modelObject = arrDelayLm.train, data = testDS, outData = testDS, computeResiduals = TRUE,
  computeStdErrors = TRUE, overwrite = TRUE))

##
## The estimated variance-covariance matrix of the coefficients must be available in order to compute prediction standard errors
```

REVOLUTION ANALYTICS

Argument: covCoef

```
system.time(arrDelayLm.train <- rxLinMod(ArrDelay ~ DayOfWeek:depTimeCat, data = trainDS,
  transforms = list(depTimeCat = cut(CRSDepTime, breaks = seq(from = 5, to = 23, by = 2))),
  covCoef = TRUE))

##    user  system elapsed
##    0.23    0.03    0.26
```

REVOLUTION ANALYTICS

Generating Std. Errors

```
rxPredict(modelObject = arrDelayLm.train, data = testDS, outData = testDS, computeResiduals = TRUE,
  computeStdErrors = TRUE, interval = "prediction", overwrite = TRUE)
rxGetVarInfo(testDS)

## Var 1: ArrDelay, Type: integer, Low/High: (-86, 1490)
## Var 2: CRSDepTime, Type: numeric, Low/High: (0.0167, 23.9833)
## Var 3: DayOfWeek
##        7 factor levels: Monday Tuesday Wednesday Thursday Friday Saturday Sunday
## Var 4: urv, Type: numeric, Low/High: (0.0000, 1.0000)
## Var 5: TrainTest
##        2 factor levels: Train Test
## Var 6: ArrDelay_Pred, Type: numeric, Low/High: (0.0812, 25.1872)
## Var 7: ArrDelay_Resid, Type: numeric, Low/High: (-97.6677, 1486.4368)
## Var 8: ArrDelay_StdErr, Type: numeric, Low/High: (0.3784, 0.8077)
## Var 9: ArrDelay_Lower, Type: numeric, Low/High: (-74.5331, -49.5667)
## Var 10: ArrDelay_Upper, Type: numeric, Low/High: (74.6955, 99.6713)
```

Mortgage Data

Set the path:

```
mortXdf <- file.path(rxGetOption("sampleDataDir"), "mortDefaultSmall.xdf")
```

Review the Data

```
rxGetInfo(data = mortXdf, getVarInfo = TRUE)
```

```
## File name: C:\Revolution\R-Enterprise-7.2\R-3.0.3\library\RevoScaleR\SampleData\mortDefaultSmall.xdf
## Number of observations: 100000
## Number of variables: 6
## Number of blocks: 10
## Compression type: zlib
## Variable information:
## Var 1: creditScore, Type: integer, Low/High: (470, 925)
## Var 2: houseAge, Type: integer, Low/High: (0, 40)
## Var 3: yearsEmploy, Type: integer, Low/High: (0, 14)
## Var 4: ccDebt, Type: integer, Low/High: (0, 14094)
## Var 5: year, Type: integer, Low/High: (2000, 2009)
## Var 6: default, Type: integer, Low/High: (0, 1)
```

rxLogit() Arguments

```
args(rxLogit)

## function (formula, data, pweights = NULL, fweights = NULL, cube = FALSE,
##     cubePredictions = FALSE, variableSelection = list(), rowSelection = NULL,
##     transforms = NULL, transformObjects = NULL, transformFunc = NULL,
##     transformVars = NULL, transformPackages = NULL, transformEnvir = NULL,
##     dropFirst = FALSE, dropMain = rxGetOption("dropMain"), covCoef = FALSE,
##     covData = FALSE, covariance = FALSE, initialValues = NULL,
##     coefLabelStyle = rxGetOption("coefLabelStyle"), blocksPerRead = rxGetOption("blocksPerRead"),
##     maxIterations = 25, coeffTolerance = 0.000001, gradientTolerance = 0.000001,
##     objectiveFunctionTolerance = 0.00000001, reportProgress = rxGetOption("reportProgress"),
##     verbose = 0, computeContext = rxGetOption("computeContext"),
##     ...)
## NULL
```

Logistic Regression: Estimation

```
logitModel1 <-
  rxLogit(
    default ~ F(year) + ccDebt + creditScore + houseAge + yearsEmploy,
    data = mortXdf,
    dropFirst = TRUE
    )
```

REVOLUTION ANALYTICS

Logistic Regression: Summary

```
summary(logitModel1)

## Call:
## rxLogit(formula = default ~ F(year) + ccDebt + creditScore +
##     houseAge + yearsEmploy, data = mortXdf, dropFirst = TRUE)
##
## Logistic Regression Results for: default ~ F(year) + ccDebt + creditScore +
##     houseAge + yearsEmploy
## Data: mortXdf (RxXdfData Data Source)
## File name:
##     C:/Revolution/R-Enterprise-7.2/R-3.0.3/library/RevoScaleR/SampleData/mortDefaultSmall.xdf
## Dependent variable(s): default
## Total independent variables: 15 (Including number dropped: 1)
## Number of valid observations: 100000
## Number of missing observations: 0
## -2*LogLikelihood: 2946.1416 (Residual deviance on 99986 degrees of freedom)
##
...
```

Summary (continued)

```
##                  Estimate    Std. Error  z value Pr(>|z|)
## (Intercept) -10.676300691   0.868163310 -12.2976 2.22e-16 ***
## F_year=2000            NA            NA       NA       NA
## F_year=2001   1.036735208   0.377298214   2.7478   0.0060 **
## F_year=2002  -0.059887745   0.451414000  -0.1327   0.8945
## F_year=2003  -0.906620754   0.595801034  -1.5217   0.1281
## F_year=2004  -0.754256100   0.530412342  -1.4220   0.1550
## F_year=2005  -0.795193515   0.553608298  -1.4364   0.1509
## F_year=2006  -0.805887276   0.563593927  -1.4299   0.1527
## F_year=2007   0.607389800   0.398804738   1.5230   0.1278
## F_year=2008   3.168104457   0.330552944   9.5843 2.22e-16 ***
## F_year=2009   3.737629855   0.327482184  11.4132 2.22e-16 ***
## ccDebt        0.001319981   0.000039315  33.5745 2.22e-16 ***
## creditScore  -0.007823851   0.001080576  -7.2404 2.22e-16 ***
## houseAge      0.028769237   0.007035559   4.0891 4.33e-05 ***
## yearsEmploy  -0.267533265   0.027405533  -9.7620 2.22e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Prediction Data

```
(newData <- data.frame(
    year   =       rep(c(2006, 2009), each=4),
    ccDebt =       rep(c(1000, 10000), 4),
    creditScore = rep(c(700, 800), 4),
    houseAge =    rep(c(1, 5, 10, 20), 2),
    yearsEmploy = 7
  ))
```

```
##   year ccDebt creditScore houseAge yearsEmploy
## 1 2006   1000         700        1           7
## 2 2006  10000         800        5           7
## 3 2006   1000         700       10           7
## 4 2006  10000         800       20           7
## 5 2009   1000         700        1           7
## 6 2009  10000         800        5           7
## 7 2009   1000         700       10           7
## 8 2009  10000         800       20           7
```

REVOLUTION ANALYTICS

Predicted Values

```
(dataWithPredctions <- rxPredict(
  modelObject = logitModel1,
  data = newData,
  outData = newData,
  type = "response"))
```

```
##   year ccDebt creditScore houseAge yearsEmploy     default_Pred
## 1 2006   1000         700        1           7 0.00000002554393
## 2 2006  10000         800        5           7 0.00188800972037
## 3 2006   1000         700       10           7 0.00000003309304
## 4 2006  10000         800       20           7 0.00290386751027
## 5 2009   1000         700        1           7 0.00000240165789
## 6 2009  10000         800        5           7 0.15099413007025
## 7 2009   1000         700       10           7 0.00000311142776
## 8 2009  10000         800       20           7 0.21495934590121
```

Census Data

```
dataPath = "../../../../Data/"
bigCensusData <- file.path(dataPath, "Census5PCT2000.xdf")
rxGetInfo(bigCensusData)

## File name: E:\GitHub\Revolution_Course_Materials\Data\Census5PCT2000.xdf
## Number of observations: 14058983
## Number of variables: 265
## Number of blocks: 98
## Compression type: none
```

REVOLUTION ANALYTICS

Subset Data

```
propinFile <- file.path("xdf/CensusPropertyIns.xdf")
rxDataStep(inData = bigCensusData,
  rowSelection = (related == "Head/Householder") & (age > 20) & (age < 90),
  varsToKeep = c("propinsr", "age", "sex", "region", "perwt"),
  outFile = propinFile,
  blocksPerRead = 10,
  overwrite = TRUE
)
```

REVOLUTION ANALYTICS

View Subset Data

```
rxGetInfoXdf(propinFile, getVarInfo = TRUE)
```

```
## File name: E:\GitHub\Revolution_Course_Materials\modules\BigData\DataCamp_Analysis\doc\xdf\CensusPropertyIns.xdf
## Number of observations: 5175270
## Number of variables: 5
## Number of blocks: 10
## Compression type: zlib
## Variable information:
## Var 1: propinsr, Annual property insurance cost
##        Type: integer, Low/High: (0, 3700)
## Var 2: age, Age
##        Type: integer, Low/High: (21, 89)
## Var 3: sex, Sex
##        2 factor levels: Male Female
## Var 4: region, Census region and division
##        17 factor levels: New England Division Middle Atlantic Division Mixed Northeast Divisions (1970 Metro) East North Centr
## Var 5: perwt, Type: integer, Low/High: (2, 193)
```

Summarize Factor

```
print(rxSummary(~region, data = propinFile), header = FALSE)
```

```
##
## Category Counts for region
## Number of categories: 17
## Number of valid observations: 5175270
## Number of missing observations: 0
##
##  region                                        Counts
##  New England Division                          265372
##  Middle Atlantic Division                      734585
##  Mixed Northeast Divisions (1970 Metro)             0
##  East North Central Div.                       847367
##  West North Central Div.                       366417
##  Mixed Midwest Divisions (1970 Metro)               0
##  South Atlantic Division                       981614
##  East South Central Div.                       324003
##  West South Central Div.                       553425
##  Mixed Southern Divisions (1970 Metro)              0
##  Mountain Division                             328940
##  Pacific Division                              773547
##  Mixed Western Divisions (1970 Metro)               0
##  Military/Military reservations                     0
##  PUMA boundaries cross state lines-1% sample        0
```

Clean Factor Variables

```
regionLevels <- list(`New England` = "New England Division", `Middle Atlantic` = "Middle Atlantic Division",
  `East North Central` = "East North Central Div.", `West North Central` = "West North Central Div.",
  `South Atlantic` = "South Atlantic Division", `East South Central` = "East South Central Div.",
  `West South Central` = "West South Central Div.", Mountain = "Mountain Division",
  Pacific = "Pacific Division")

rxFactors(inData = propinFile, outFile = propinFile, factorInfo = list(region = list(newLevels = regionLevels,
  otherLevel = "Other")), overwrite = TRUE)
```

REVOLUTION ANALYTICS

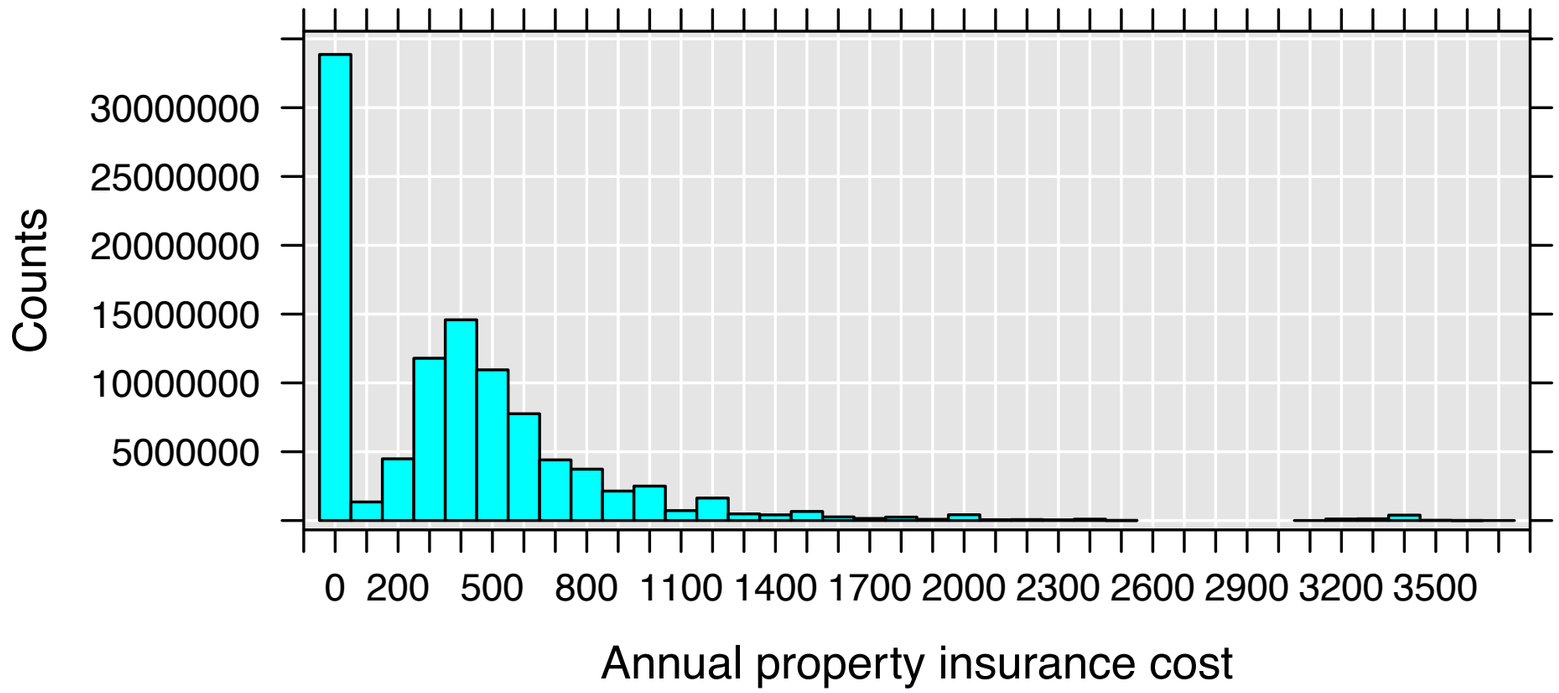
Re-Summarize Factor

```
print(rxSummary(~region, data = propinFile), header = FALSE)
```

```
##
## Category Counts for region
## Number of categories: 10
## Number of valid observations: 5175270
## Number of missing observations: 0
##
##  region             Counts
##  New England        265372
##  Middle Atlantic    734585
##  East North Central 847367
##  West North Central 366417
##  South Atlantic     981614
##  East South Central 324003
##  West South Central 553425
##  Mountain           328940
##  Pacific            773547
##  Other                   0
```

Property Insurance: Histogram

```
rxHistogram(~propinsr, data = propinFile, pweights = "perwt", numBreaks = 50)
```



rxGlm() Arguments

```
args(rxGlm)

## function (formula, data, family = gaussian(), pweights = NULL,
##     fweights = NULL, offset = NULL, cube = FALSE, variableSelection = list(),
##     rowSelection = NULL, transforms = NULL, transformObjects = NULL,
##     transformFunc = NULL, transformVars = NULL, transformPackages = NULL,
##     transformEnvir = NULL, dropFirst = FALSE, dropMain = rxGetOption("dropMain"),
##     covCoef = FALSE, computeAIC = FALSE, initialValues = NA,
##     coefLabelStyle = rxGetOption("coefLabelStyle"), blocksPerRead = rxGetOption("blocksPerRead"),
##     maxIterations = 25, coeffTolerance = 0.000001, objectiveFunctionTolerance = 0.00000001,
##     reportProgress = rxGetOption("reportProgress"), verbose = 0,
##     computeContext = rxGetOption("computeContext"), ...)
## NULL
```

GLM with Tweedie Family

```
system.time(
propinGlm <-
  rxGlm(propinsr~sex + F(age) + region,
   pweights = "perwt",
   data = propinFile,
   family = rxTweedie(var.power = 1.5),
   dropFirst = TRUE
  )
)

##    user  system elapsed
##   32.40    6.40   39.58
```

GLM with Tweedie Family: Output

```
printCoefmat(coef(summary(propinGlm)))
```

```
##                          Estimate   Std. Error   t value  Pr(>|t|)
## (Intercept)             0.123131629  0.000589255  208.9614 2.220e-16 ***
## sex=Male                         NA           NA        NA        NA
## sex=Female              0.009026344  0.000031637  285.3054 2.220e-16 ***
## F_age=21                         NA           NA        NA        NA
## F_age=22               -0.009208145  0.000752297  -12.2400 2.220e-16 ***
## F_age=23               -0.019803733  0.000696578  -28.4300 2.220e-16 ***
## F_age=24               -0.028556094  0.000664796  -42.9546 2.220e-16 ***
## F_age=25               -0.036518683  0.000643205  -56.7761 2.220e-16 ***
## F_age=26               -0.043705574  0.000628859  -69.4998 2.220e-16 ***
## F_age=27               -0.048937559  0.000618199  -79.1616 2.220e-16 ***
## F_age=28               -0.053979330  0.000609898  -88.5055 2.220e-16 ***
## F_age=29               -0.057865092  0.000604339  -95.7494 2.220e-16 ***
## F_age=30               -0.060635022  0.000602040 -100.7160 2.220e-16 ***
## F_age=31               -0.063358444  0.000600426 -105.5224 2.220e-16 ***
## F_age=32               -0.065257338  0.000599059 -108.9332 2.220e-16 ***
## F_age=33               -0.067210867  0.000597490 -112.4887 2.220e-16 ***
## F_age=34               -0.068536775  0.000596242 -114.9478 2.220e-16 ***
## F_age=35               -0.069415252  0.000594879 -116.6879 2.220e-16 ***
## F_age=36               -0.070896305  0.000594058 -119.3425 2.220e-16 ***
## F_age=37               -0.071837352  0.000593582 -121.0234 2.220e-16 ***
...
```

REVOLUTION ANALYTICS

Data To Predict

```
Ages <- 21:89
predData <-
  data.frame(age = Ages,
   sex = gl(2, length(Ages),labels = c("Male", "Female")),
   region = factor(rep(c(5, 2), each =
   length(Ages)*2),
   levels = 1:10, labels = rxGetVarInfo(propinFile)$region$levels)
   )
nrow(predData)

## [1] 276
```

REVOLUTION ANALYTICS

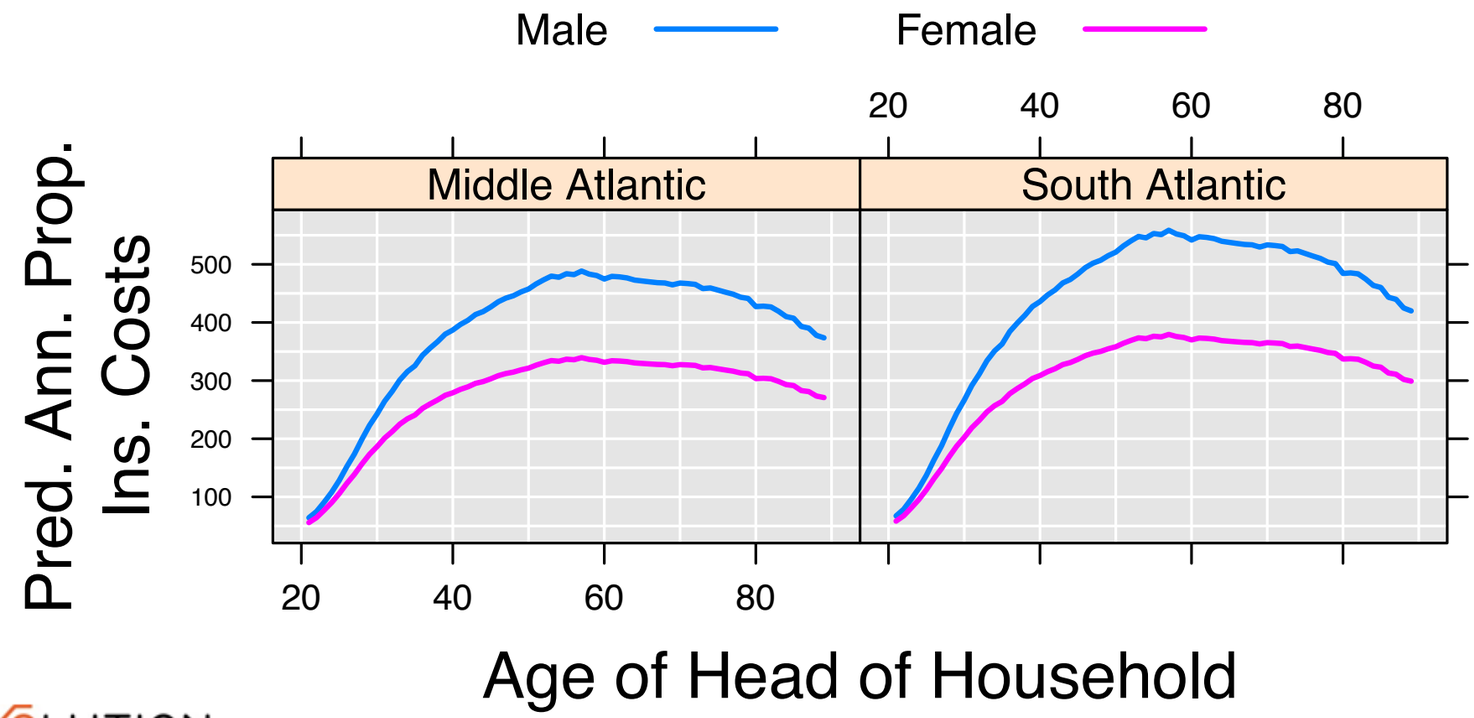
Predicted Data for GLM

```
predData <- rxPredict(propinGlm,
  data = predData,
  outData=predData,
  predVarNames=c("predicted"),
  overwrite=TRUE)
```

REVOLUTION ANALYTICS

Plot Predictions

```
rxLinePlot(predicted ~age|region, group = sex, data = predData,
 auto.key=list(columns=2, points=FALSE, lines=TRUE),
 xTitle = list("Age of Head of Household", cex=1.5),
 yTitle = list("Pred. Ann. Prop.\nIns. Costs",cex=1.5),
 scales=list(y=list(cex=0.6))
)
```



Data

```
sampleDataDir <- rxGetOption("sampleDataDir")
mdata <- file.path(sampleDataDir, "mortDefaultSmall.xdf")
mortDefault2 <- file.path("xdf/mortDefault2.xdf")
```

REVOLUTION ANALYTICS

Create New Variables

```
rxDataStep(inData = mdata, outFile = mortDefault2,
           transforms = list(RandomSample = sample(10, size = .rxNumRows, replace = TRUE)),
           overwrite = TRUE)
rxGetVarInfo(mortDefault2)

## Var 1: creditScore, Type: integer, Low/High: (470, 925)
## Var 2: houseAge, Type: integer, Low/High: (0, 40)
## Var 3: yearsEmploy, Type: integer, Low/High: (0, 14)
## Var 4: ccDebt, Type: integer, Low/High: (0, 14094)
## Var 5: year, Type: integer, Low/High: (2000, 2009)
## Var 6: default, Type: integer, Low/High: (0, 1)
## Var 7: RandomSample, Type: integer, Low/High: (1, 10)
```

rxKmeans()

```
args(rxKmeans)

## function (formula, data, outFile = NULL, outColName = ".rxCluster",
##     writeModelVars = FALSE, overwrite = FALSE, numClusters = NULL,
##     centers = NULL, algorithm = "Lloyd", numStartRows = 0, maxIterations = 1000,
##     numStarts = 1, rowSelection = NULL, transforms = NULL, transformObjects = NULL,
##     transformFunc = NULL, transformVars = NULL, transformPackages = NULL,
##     transformEnvir = NULL, blocksPerRead = rxGetOption("blocksPerRead"),
##     reportProgress = rxGetOption("reportProgress"), verbose = 0,
##     computeContext = rxGetOption("computeContext"), xdfCompressionLevel = rxGetOption("xdfCompressionLevel"),
##     ...)
## NULL
```

Estimate the Clusters

```
md.km <- rxKmeans(formula = ~ creditScore + houseAge + yearsEmploy + ccDebt + year,
                  data = mortDefault2,
                  numClusters = 3,
                  outFile = mortDefault2,
                  algorithm = "lloyd",
                  overwrite = TRUE)
```

View the Output

```
print(md.km)

## Call:
## rxKmeans(formula = ~creditScore + houseAge + yearsEmploy + ccDebt +
##     year, data = mortDefault2, outFile = mortDefault2, overwrite = TRUE,
##     numClusters = 3, algorithm = "lloyd")
##
## Data: mortDefault2
## Number of valid observations: 100000
## Number of missing observations: 0
## Clustering algorithm:
##
## K-means clustering with 3 clusters of sizes 26816, 27364, 45820
##
## Cluster means:
##   creditScore houseAge yearsEmploy   ccDebt     year
## 1    699.8421 19.95693    5.013723 2555.351 2004.494
## 2    699.5454 19.96609    4.996199 7433.085 2004.520
## 3    700.1138 19.95853    4.999956 4987.269 2004.492
##
## Within cluster sum of squares by cluster:
##           1           2           3
## 24302543771 27215486100 21700772048
##
```

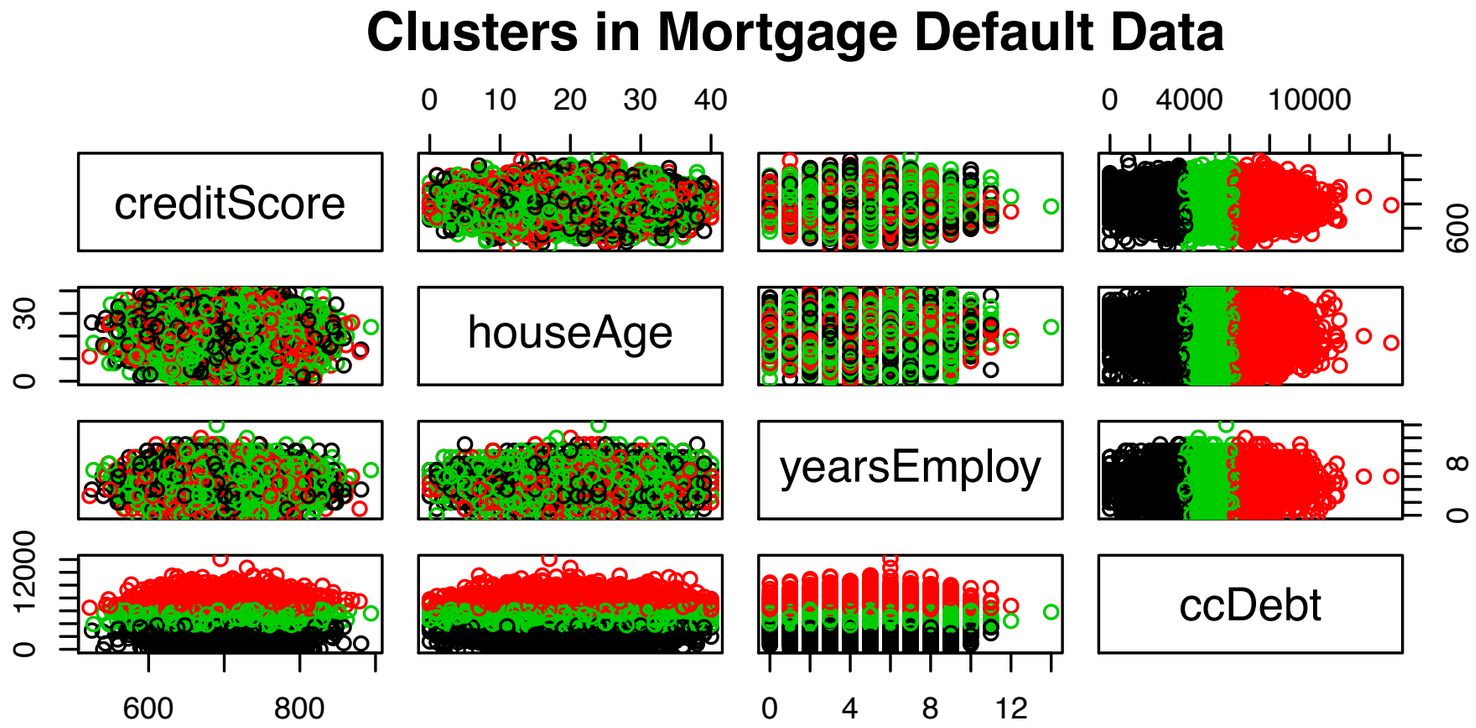
REVOLUTION ANALYTICS

Clusters

```
rxGetVarInfo(mortDefault2)

## Var 1: creditScore, Type: integer, Low/High: (470, 925)
## Var 2: houseAge, Type: integer, Low/High: (0, 40)
## Var 3: yearsEmploy, Type: integer, Low/High: (0, 14)
## Var 4: ccDebt, Type: integer, Low/High: (0, 14094)
## Var 5: year, Type: integer, Low/High: (2000, 2009)
## Var 6: default, Type: integer, Low/High: (0, 1)
## Var 7: RandomSample, Type: integer, Low/High: (1, 10)
## Var 8: .rxCluster, Type: integer, Low/High: (1, 3)
```

Plot Clusters

```
mdDf <- rxXdfToDataFrame(file=mortDefault2,
                         rowSelection = RandomSample == 5)
plot(mdDf[,1:4],col=mdDf$.rxCluster)
```



REVOLUTION ANALYTICS

Decision Trees: Data

`mortDefault2` was created in the last session

```
mortDefault2 <- file.path("xdf/mortDefault2.xdf")
rxGetInfo(mortDefault2, getVarInfo = TRUE)

## File name: E:\GitHub\Revolution_Course_Materials\modules\BigData\DataCamp_Analysis\doc\xdf\mortDefault2.xdf
## Number of observations: 100000
## Number of variables: 8
## Number of blocks: 10
## Compression type: zlib
## Variable information:
## Var 1: creditScore, Type: integer, Low/High: (470, 925)
## Var 2: houseAge, Type: integer, Low/High: (0, 40)
## Var 3: yearsEmploy, Type: integer, Low/High: (0, 14)
## Var 4: ccDebt, Type: integer, Low/High: (0, 14094)
## Var 5: year, Type: integer, Low/High: (2000, 2009)
## Var 6: default, Type: integer, Low/High: (0, 1)
## Var 7: RandomSample, Type: integer, Low/High: (1, 10)
## Var 8: .rxCluster, Type: integer, Low/High: (1, 3)
```

Split Files into Training and Testing

```
trainTestFiles <- rxSplit(
  inData = mortDefault2,
  transforms = list(TrainTest=factor(ifelse(RandomSample < 9,"Train","Test"))),
  splitByFactor="TrainTest", overwrite=TRUE
)
```

REVOLUTION ANALYTICS

rxDTree Arguments

```
args(rxDTree)

## function (formula, data, outFile = NULL, outColName = ".rxNode",
##     writeModelVars = FALSE, overwrite = FALSE, pweights = NULL,
##     fweights = NULL, method = NULL, parms = NULL, cost = NULL,
##     minSplit = max(20, sqrt(numObs)), minBucket = round(minSplit/3),
##     maxDepth = 10, cp = 0, maxCompete = 0, maxSurrogate = 0,
##     useSurrogate = 2, surrogateStyle = 0, xVal = 2, maxNumBins = NULL,
##     maxUnorderedLevels = 32, removeMissings = FALSE, pruneCp = 0,
##     rowSelection = NULL, transforms = NULL, transformObjects = NULL,
##     transformFunc = NULL, transformVars = NULL, transformPackages = NULL,
##     transformEnvir = NULL, blocksPerRead = rxGetOption("blocksPerRead"),
##     reportProgress = rxGetOption("reportProgress"), verbose = 0,
##     computeContext = rxGetOption("computeContext"), xdfCompressionLevel = rxGetOption("xdfCompressionLevel"),
##     ...)
## NULL
```

Estimate a Regression Tree

```
treeR <- rxDTree(
  formula = default ~ houseAge + year + creditScore + yearsEmploy + ccDebt,
  data=trainTestFiles[[1]],
  maxdepth=5
  )
```

REVOLUTION ANALYTICS

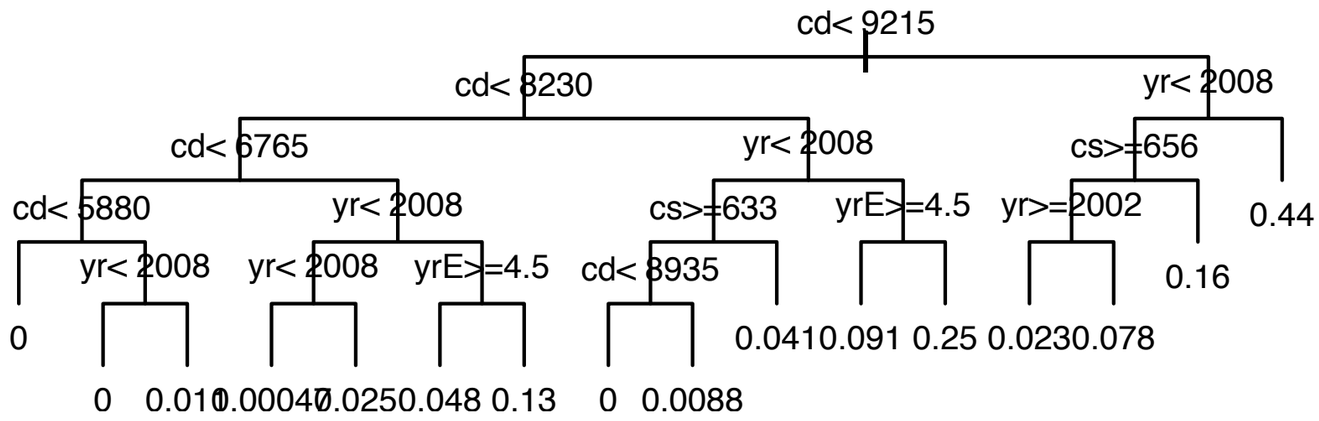
Print the Tree

```
treeR

## Call:
## rxDTree(formula = default ~ houseAge + year + creditScore + yearsEmploy +
##     ccDebt, data = trainTestFiles[[1]], maxdepth = 5)
## File:  E:\GitHub\Revolution_Course_Materials\modules\BigData\DataCamp_Analysis\doc\mortDefault2.TrainTest.Test.xdf
## Number of valid observations:  20241
## Number of missing observations:  0
##
## Tree representation:
## n= 20241
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 20241 97.5255200 0.0048416580
##    2) ccDebt< 9215 19955 59.8195900 0.0030067650
##      4) ccDebt< 8230 19237 34.9363200 0.0018194110
##        8) ccDebt< 6765 16536  5.9978230 0.0003628447
##         16) ccDebt< 5880 13706  0.0000000 0.0000000000 *
##         17) ccDebt>=5880 2830  5.9872790 0.0021201410
##           34) year< 2007.5 2287  0.0000000 0.0000000000 *
##           35) year>=2007.5 543  5.9337020 0.0110497200 *
...
```

REVOLUTION ANALYTICS

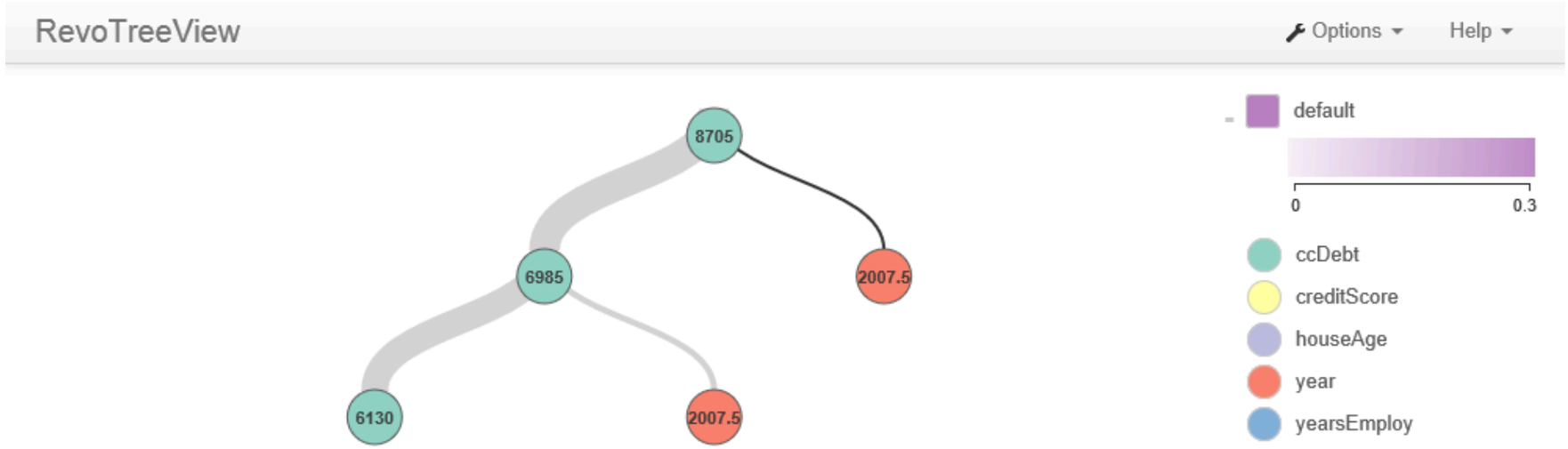
Plot Regression Tree

```
plot(rxAddInheritance(treeR), uniform = TRUE, margin = 0.1)
text(rxAddInheritance(treeR), digits = 2, cex = 0.6)
title(main = "Regression Tree for Mortgage Data")
```

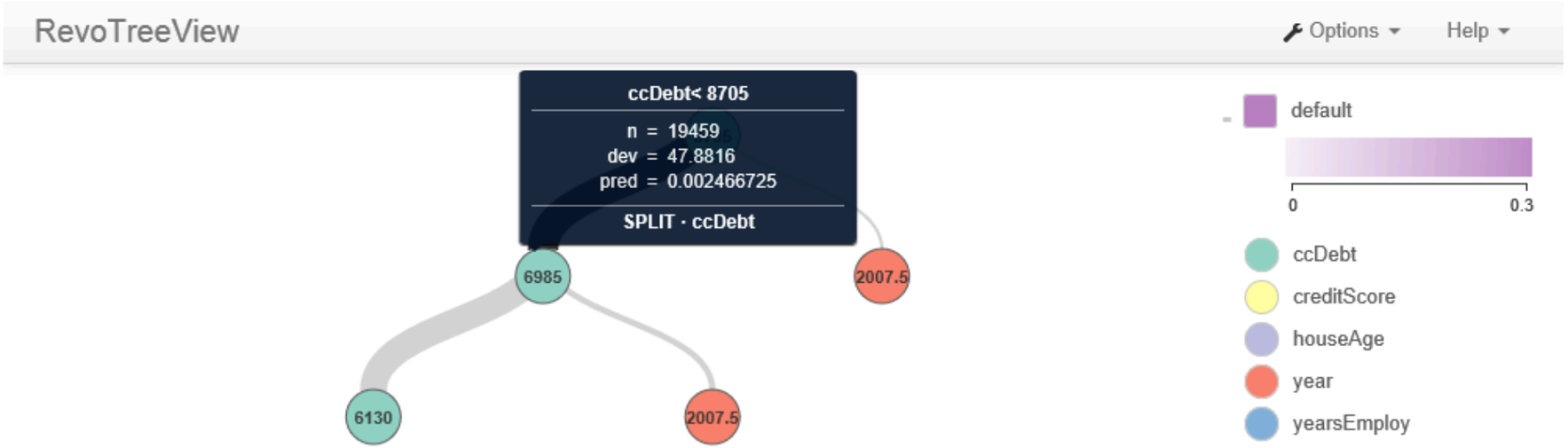


RevoTreeView

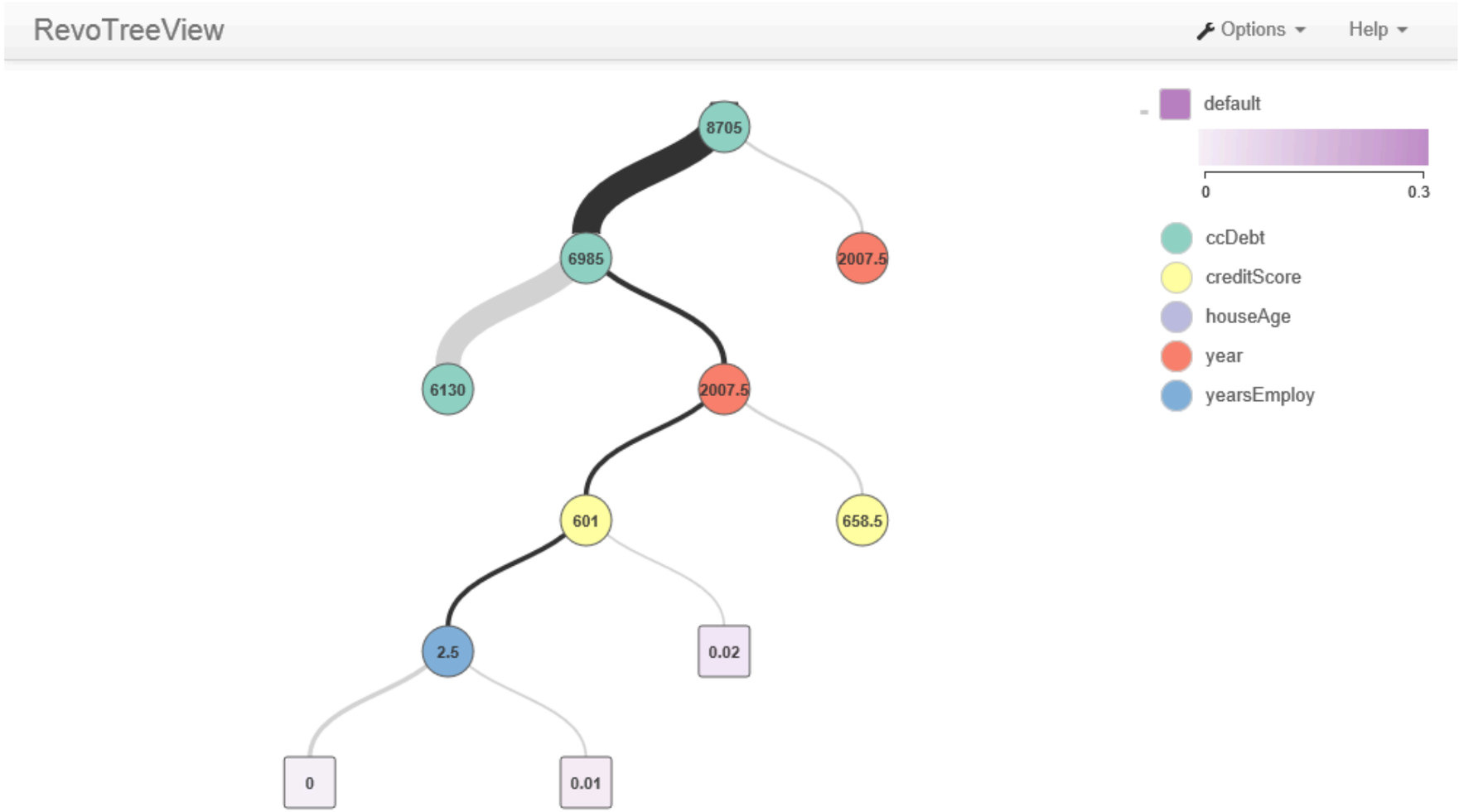
```
library(RevoTreeView)
plot(createTreeView(treeR))
```



RevoTreeView (Mouseover)



RevoTreeView (Deep)



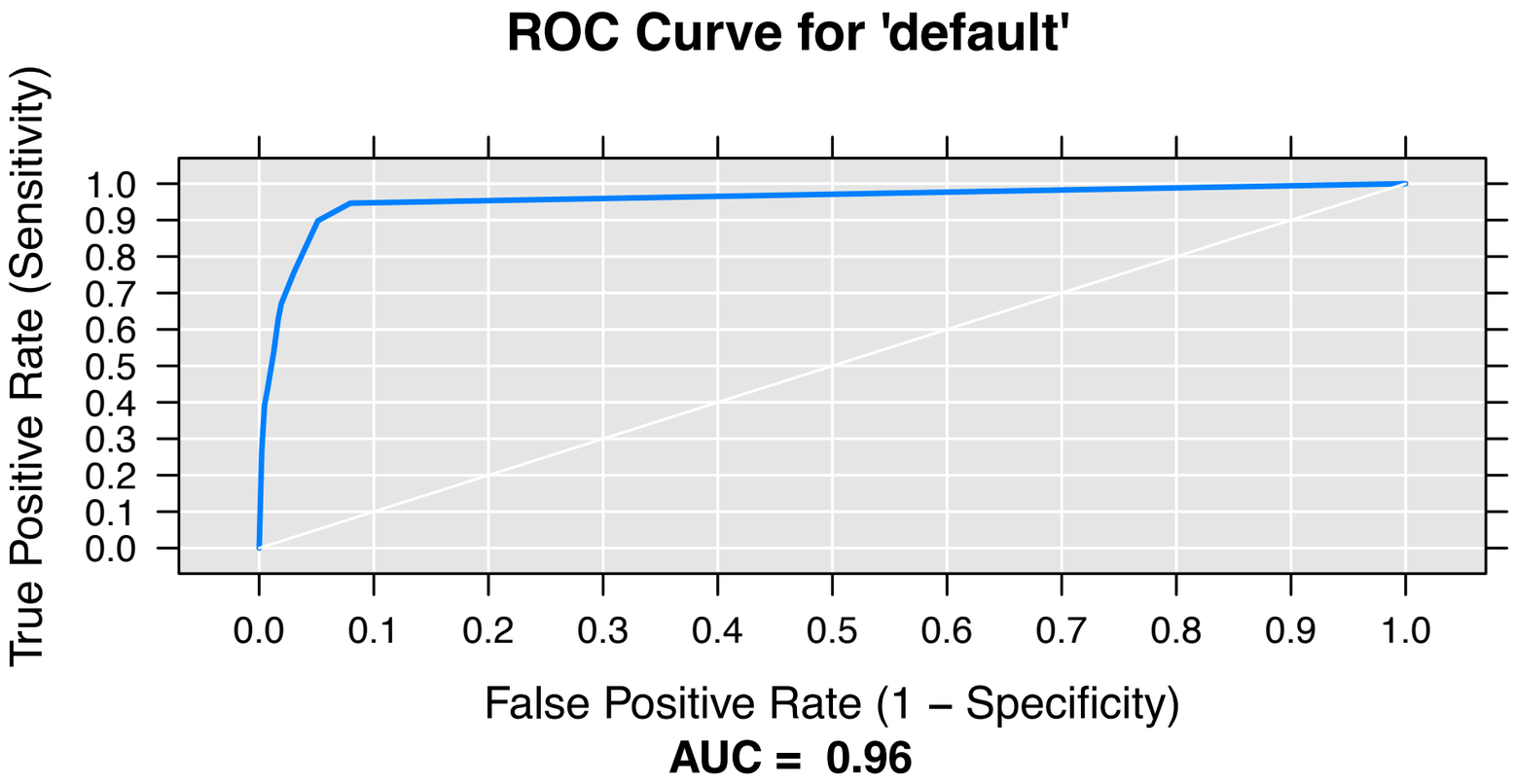
Tree Predictions

```
rxPredict(treeR, data = trainTestFiles[[2]], outData = trainTestFiles[[2]], overwrite = TRUE,
  predVarNames = "Pred_R")
```

REVOLUTION ANALYTICS

ROC Curves

```
rxRocCurve(actualVarName = "default", predVarNames = c("Pred_R"), data = trainTestFiles[[2]])
```



Build factor for Classification Tree

Training Dataset:

```
rxFactors(inData = trainTestFiles[[1]], factorInfo = list(defaultFactor = list(varName = "default")),
  outFile = trainTestFiles[[1]], overwrite = TRUE)
```

Testing Dataset:

```
rxFactors(inData = trainTestFiles[[2]], factorInfo = list(defaultFactor = list(varName = "default")),
  outFile = trainTestFiles[[2]], overwrite = TRUE)
```

View defaultFactor

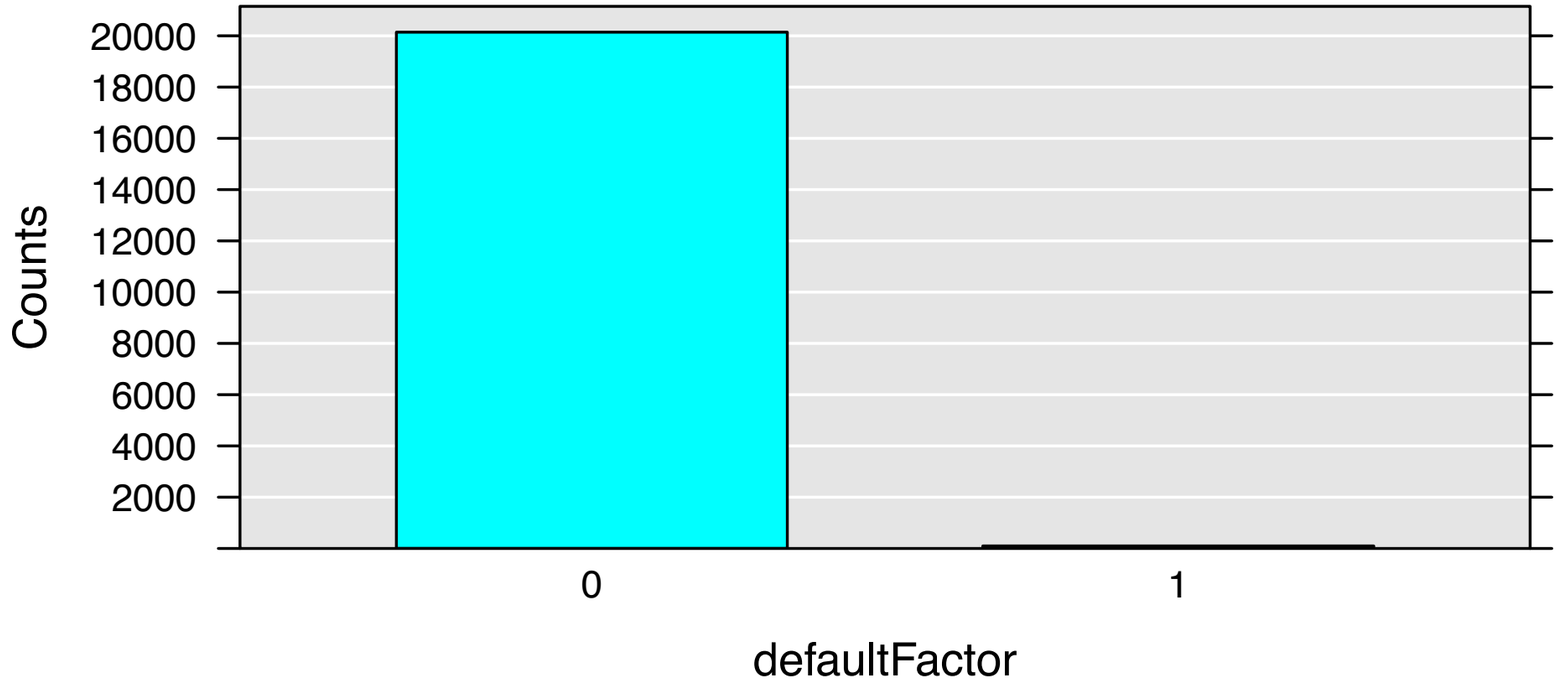
```
rxGetInfo(trainTestFiles[[1]], getVarInfo = TRUE, varsToKeep = c("default", "defaultFactor"))
```

```
## File name: E:\GitHub\Revolution_Course_Materials\modules\BigData\DataCamp_Analysis\doc\mortDefault2.TrainTest.Test.xdf
## Number of observations: 20241
## Number of variables: 15
## Number of blocks: 10
## Compression type: zlib
## Variable information:
## Var 1: default, Type: integer, Low/High: (0, 1)
## Var 2: defaultFactor
##        2 factor levels: 0 1
```

REVOLUTION ANALYTICS

Histogram of Default Factors

```
rxHistogram(~defaultFactor, data = trainTestFiles[[1]])
```



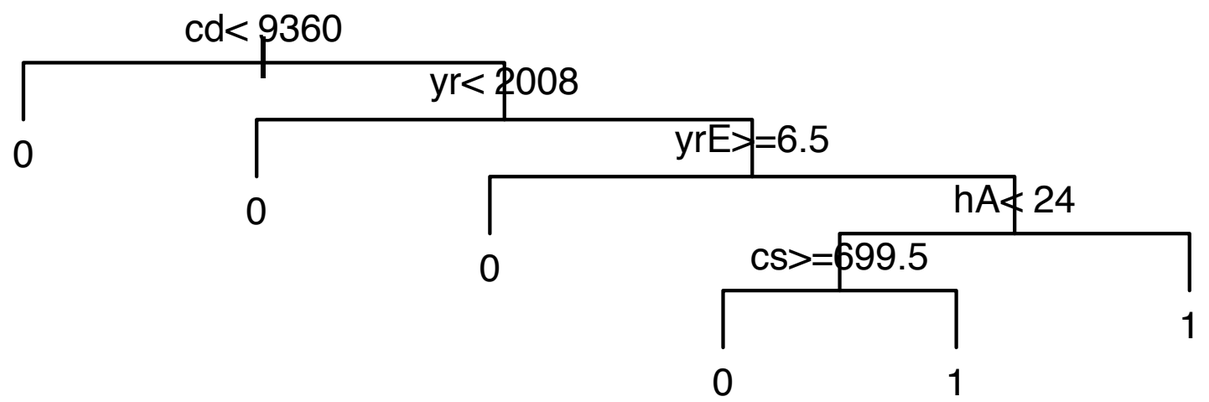
Build a Classification Tree

```
control <- list(minsplit = 20, cp = 0.01, xval = 2, maxdepth = 5, maxcompete = 0, maxsurrogate = 0,
  usesurrogate = 2, surrogatestyle = 0)
treeC <- rxDTree(formula = defaultFactor ~ houseAge + year + creditScore + yearsEmploy +
  ccDebt, data = trainTestFiles[[1]], control = control, maxNumBins = 15000)
```

REVOLUTION ANALYTICS

Plot the Classification Tree

```
plot(rxAddInheritance(treeC), uniform = TRUE, margin = 0.15)
text(rxAddInheritance(treeC), digits = 2, cex = 0.7)
title(main = "Classification Tree for Mortgage Data")
```



Predict by Classification Tree

```
rxPredict(treeC, data = trainTestFiles[[2]], outData = trainTestFiles[[2]], overwrite = TRUE,
  predVarNames = "Pred_C", type = "vector")
rxGetInfo(trainTestFiles[[2]], getVarInfo = TRUE, varsToKeep = c("defaultFactor", "Pred_C"),
  numRows = 5)

## File name: E:\GitHub\Revolution_Course_Materials\modules\BigData\DataCamp_Analysis\doc\mortDefault2.TrainTest.Train.xdf
## Number of observations: 79759
## Number of variables: 13
## Number of blocks: 10
## Compression type: zlib
## Variable information:
## Var 1: defaultFactor
##        2 factor levels: 0 1
## Var 2: Pred_C, Type: numeric, Low/High: (1.0000, 2.0000)
## Data (5 rows starting with row 1):
##   defaultFactor Pred_C
## 1             0      1
## 2             0      1
## 3             0      1
## 4             0      1
## 5             0      1
```

Relabel Predictions

```
rxFactors(inData = trainTestFiles[[2]], outFile = trainTestFiles[[2]], overwrite = TRUE,
  factorInfo = list(Pred_CF = list(newLevels = c("0", "1"), levels = c("1", "2"), varName = "Pred_C")))
```

View the Relabeled Predictions

```
rxGetInfo(trainTestFiles[[2]], getVarInfo = TRUE, varsToKeep = c("defaultFactor", "Pred_C",
  "Pred_CF"), numRows = 5)

## File name: E:\GitHub\Revolution_Course_Materials\modules\BigData\DataCamp_Analysis\doc\mortDefault2.TrainTest.Train.xdf
## Number of observations: 79759
## Number of variables: 13
## Number of blocks: 10
## Compression type: zlib
## Variable information:
## Var 1: defaultFactor
##        2 factor levels: 0 1
## Var 2: Pred_C, Type: numeric, Low/High: (1.0000, 2.0000)
## Var 3: Pred_CF
##        2 factor levels: 0 1
## Data (5 rows starting with row 1):
##   defaultFactor Pred_C Pred_CF
## 1             0      1       0
## 2             0      1       0
## 3             0      1       0
## 4             0      1       0
## 5             0      1       0
```

Confusion Matrix

```
conf.mat <- rxCrossTabs(~defaultFactor:Pred_CF, data = trainTestFiles[[2]])
conf.mat$counts[[1]]
```

```
##              Pred_CF
## defaultFactor     0  1
##             0 79316 70
##             1   330 43
```

```
prop.table(conf.mat$counts[[1]])
```

```
##              Pred_CF
## defaultFactor           0            1
##             0 0.994445768 0.0008776439
##             1 0.004137464 0.0005391241
```

REVOLUTION ANALYTICS

Thank you

Revolution Analytics is the leading commercial provider of software and support for the popular open source R statistics language.

www.revolutionanalytics.com
1.855.GET.REVO
Twitter: @RevolutionR