Microsoft R Server and SQL Server 2016

Tomaž Kaštrun

e: tomaz.kastrun@gmail.com
t: @tomaz_tsql
b: https://tomaztsql.wordpress.com

R consortium

BI CONSULTING

andego

Morgan Stanley

Shift R

Data Solutions

hiflylabs

KOGENTUM
egyszerűen komplex

PRECOGNOX

quanopt

rapidminer

SYNETIQ

USTREAM
an IBM Company

About

- BI Developer and data analyst
- 15years experience with MSSQL Server
- 15years experience data analysis and DM
- Working: Spar ICS Austria, Spar Slovenija
- MCT, MCPT, MCSE SQL Server
- tomaz.kastrun@gmail.com
- @tomaz_tsql

https://tomaztsql.wordpress.com

- Frequent community speaker at SQL and Microsoft events
- Blogger, Avid Coffee Lover, Bicycle junkie

Analytical Barriers

Common Challenges

Addressing Challanges with R from Microsoft

Uncertain total cost of ownership

Inadequate access to important business data

Limited business agility

Limited business value

Peace of mind

Efficiency

Speed and scalability

Flexibility and agility

What is R?

- A Language Platform
  - A Procedural Language optimized for Statistics and data science (and much more)
  - A Data Visualization framework
  - Provided as Free Software

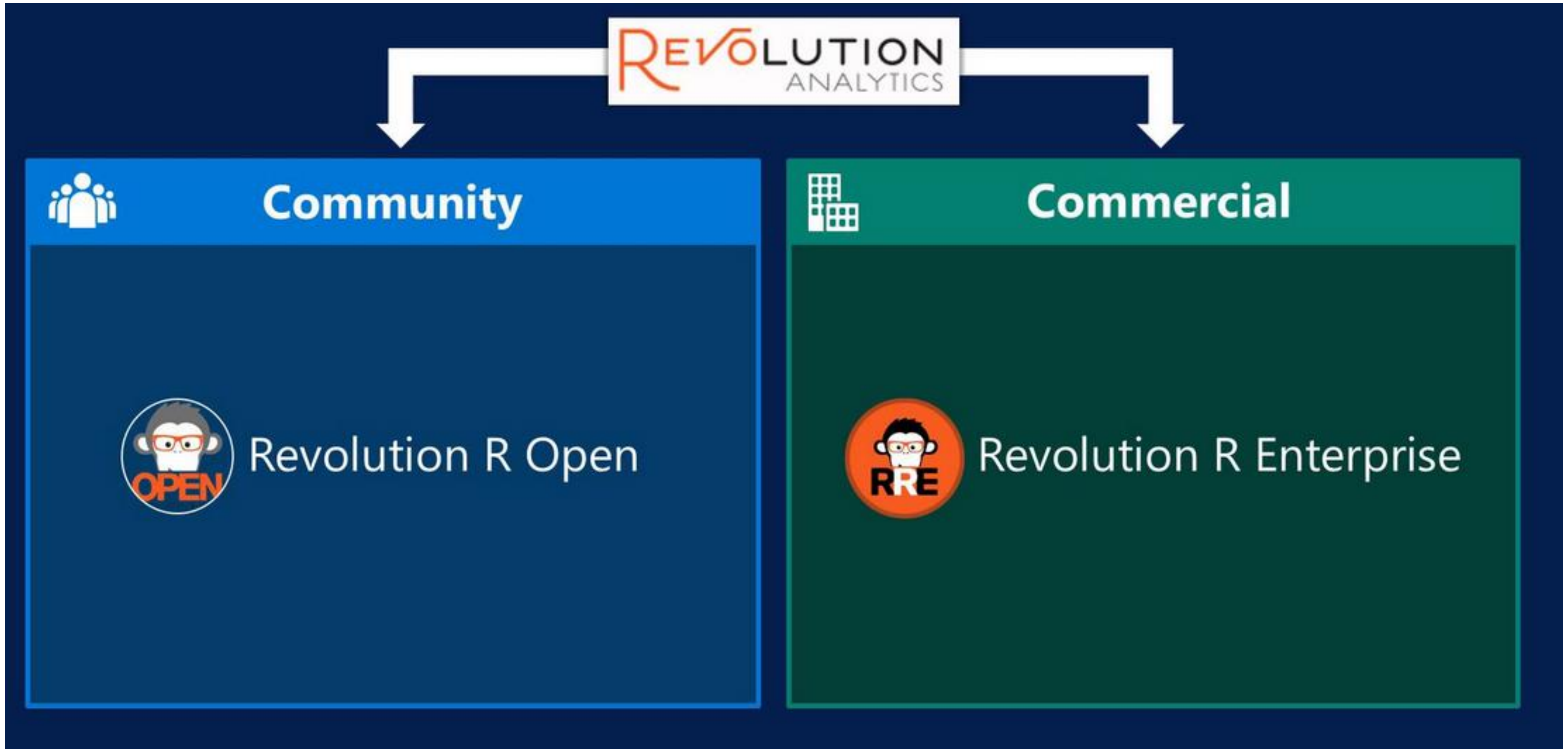
- A Community and a system
  - Taught on universitieis and many active user groups across the world
  - Estimated 3Mio Users
  - Repositories (CRAN, BioConductor, Github,…)

In fact, R is a movement!

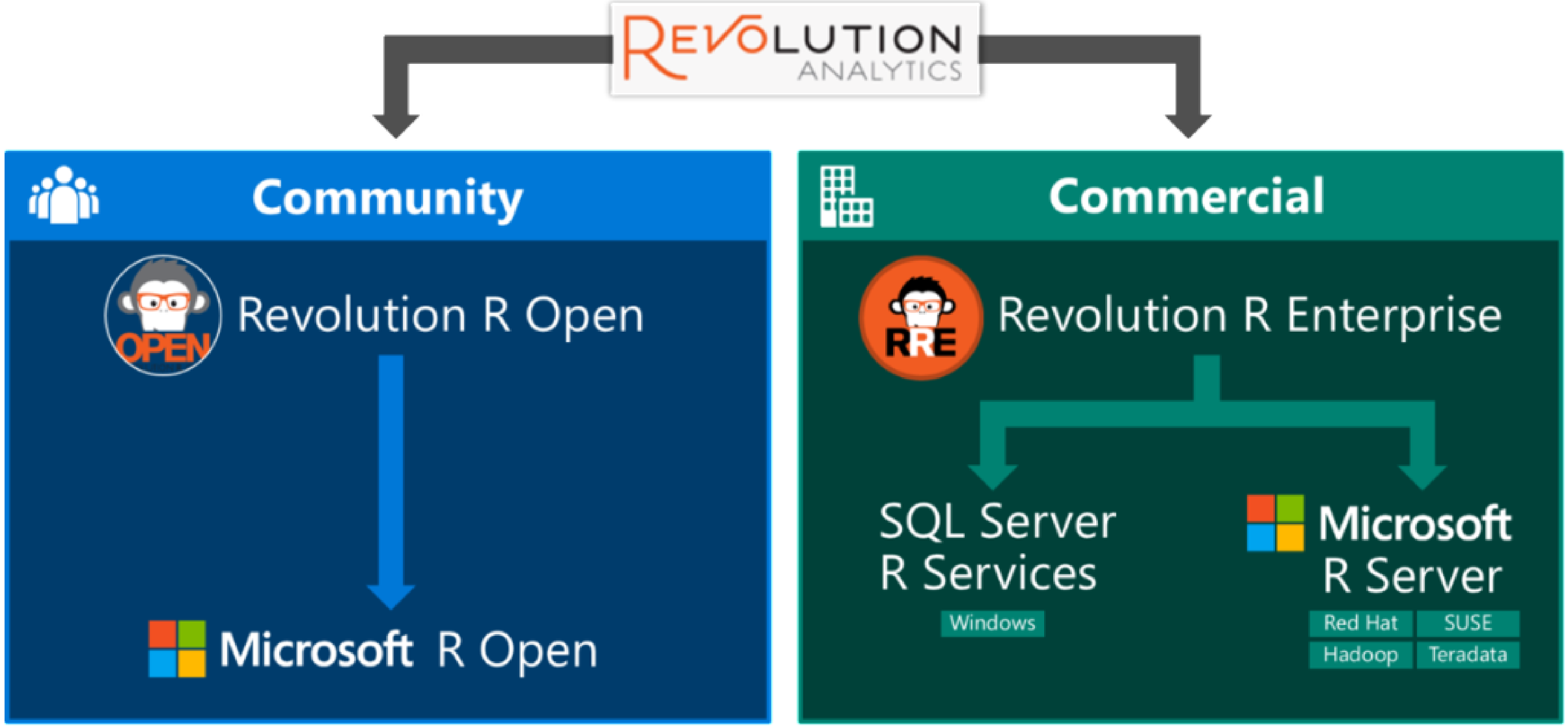
Limitations of R as a free software

- Memory Based Data access model
- Interpreted vs. Compiled Performance
- Lack of parallel computation
- Data movement & Duiplication Costs
- Governance and providence oversight
- Community support vs. Enterprise utilization

Revolution Analytics Product Integration



Microsoft R SQL Server platform



-> Free and open Source R distribution
-> Enhanced and distributed by Revolution analytics

-> Built in Advanced Analytics and Standalone Server Capability
-> Leverages the benefits of SQL Server 2016EE

Microsoft R Platform

Microsoft R Open

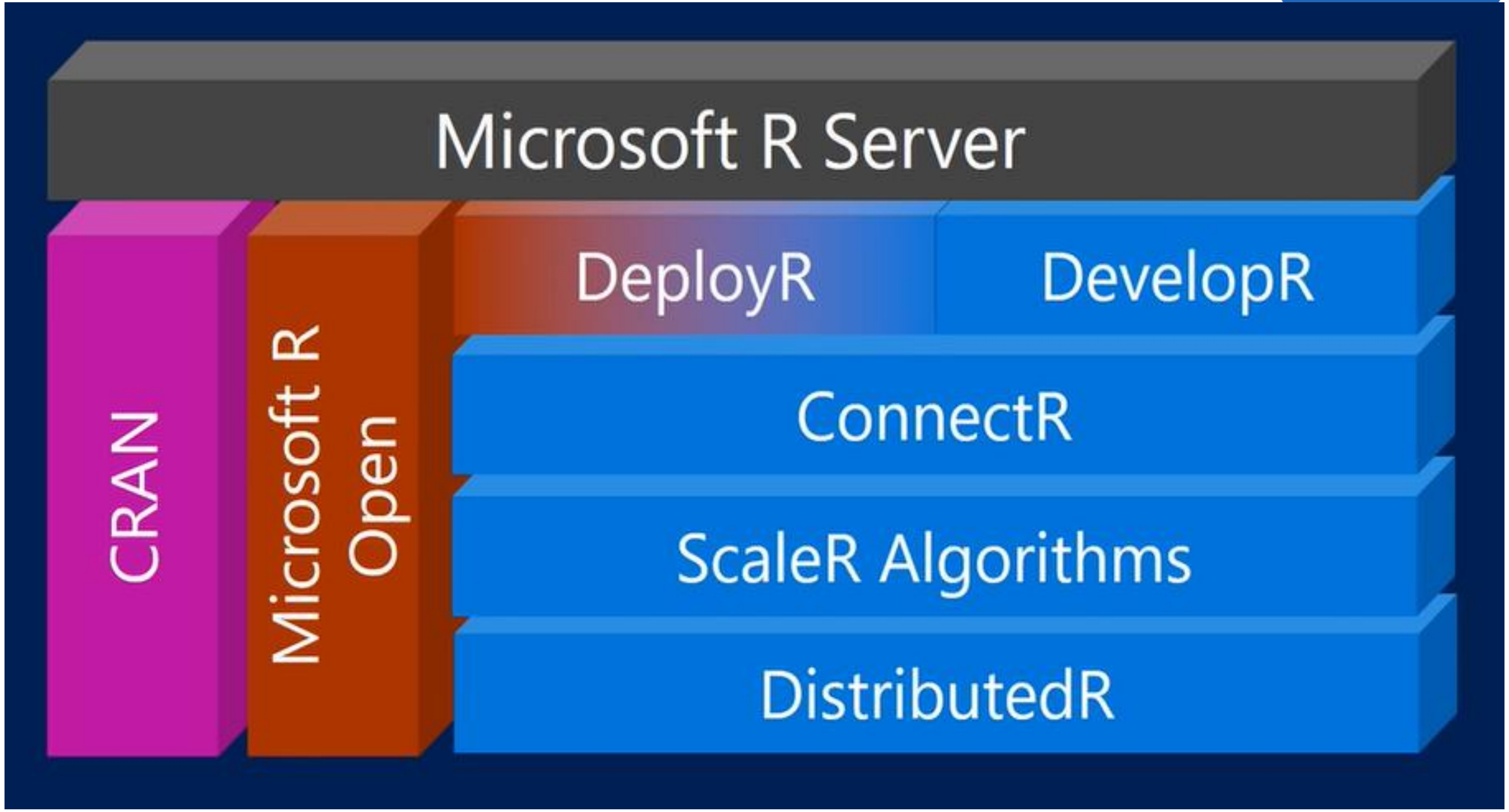
Microsoft R Client

Microsoft SQL R Services

Microsoft R Server

Different flavors:

- Microsoft R server for Linux,
- Microsoft R Server for Teradata,
- Microsoft R Server for Hadoop,
- Microsoft R HDInsight

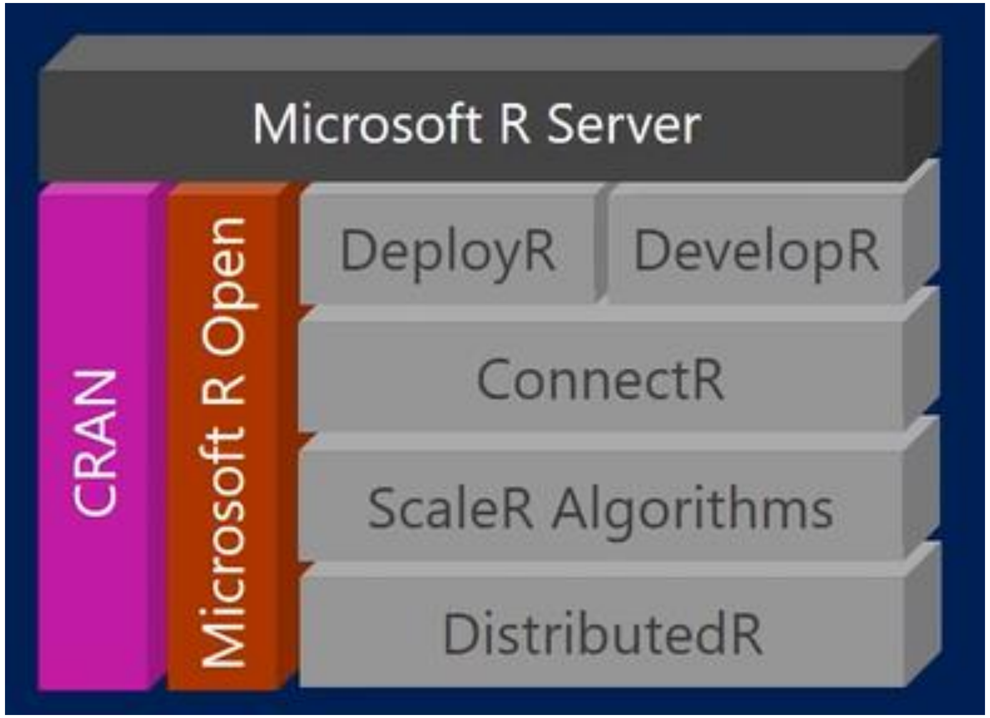


Microsoft R Server

- Evolved from Revolution R Enterprise
- Based on open Source R
- Adapted for Enterprise Scale
- For multiple platforms
  - Hadoop
  - Teradata
  - LinuX
  - Azure
  - Windows
- Interoperable
- On-premises + Cloud + Hybrid
- Operationzalize analytics for Big scale datasets and big data

Based on Open Source R

- Open source based
- Runs your normal R Script
- MetaCran / CRAN / Github / Bioconductor



Microsoft R Server

CRAN

Microsoft R Open

DeployR

DevelopR

ConnectR

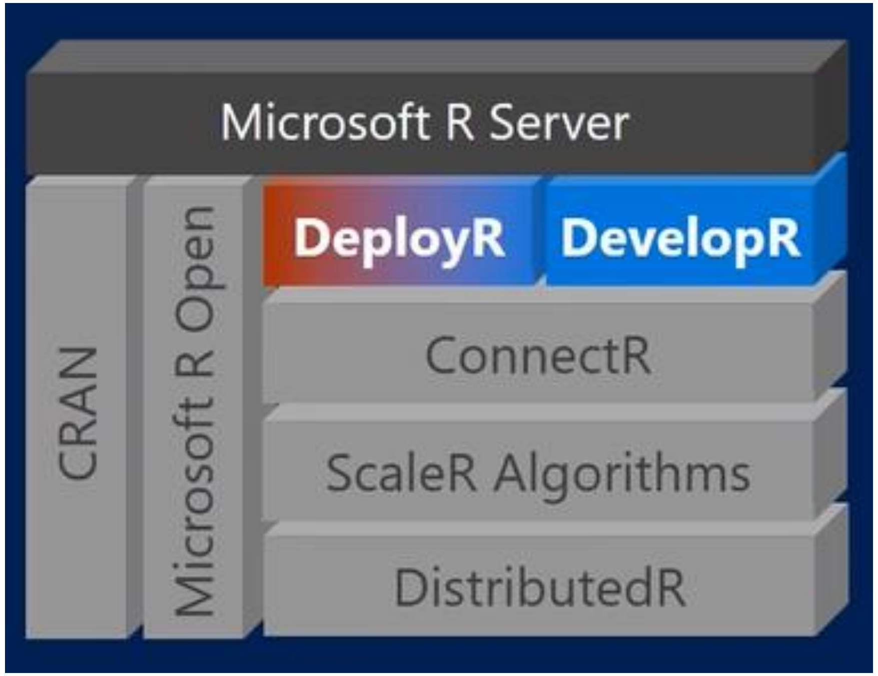
ScaleR Algorithms

DistributedR

Microsoft R Server

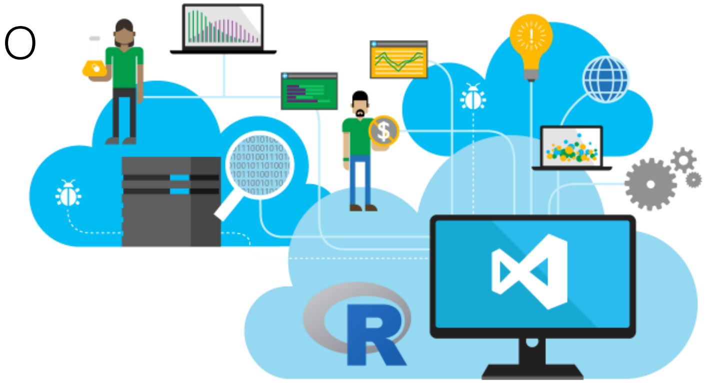
DeployR

- Web service - API integration
- Compatible with array of tools
- Abstract usage of R without knowing it



DevelopR

- R IDE based on Visual studio
- Rstudio for linux Users
- Client Based



Microsoft R Server

ConnectR

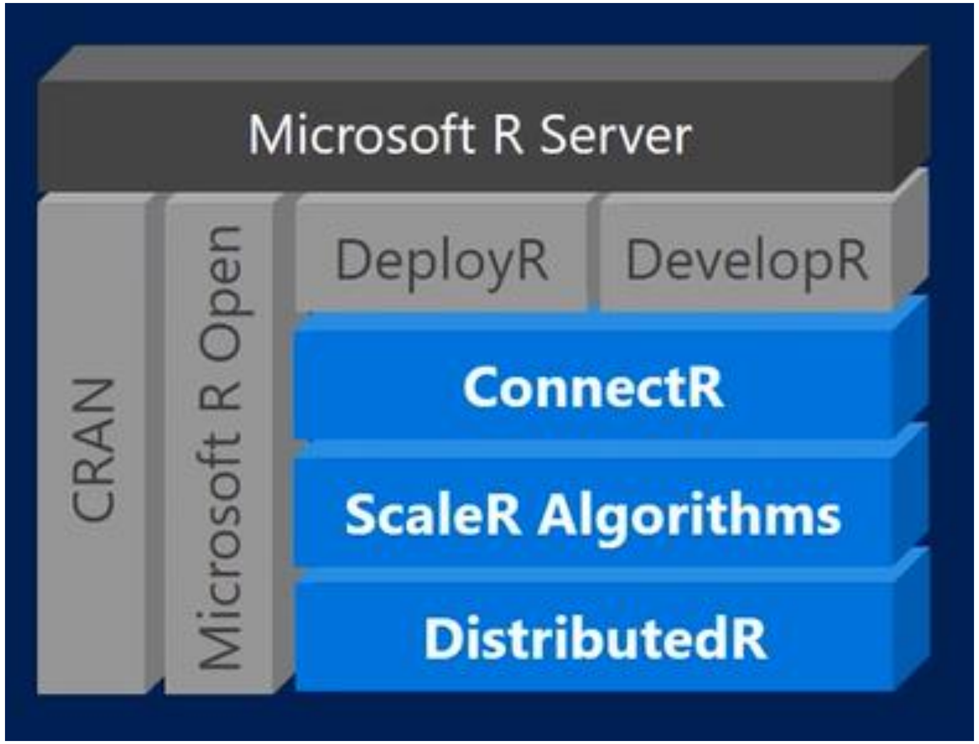
- Serier of connectors for consistent access to scaleR algorithms

DistributedR

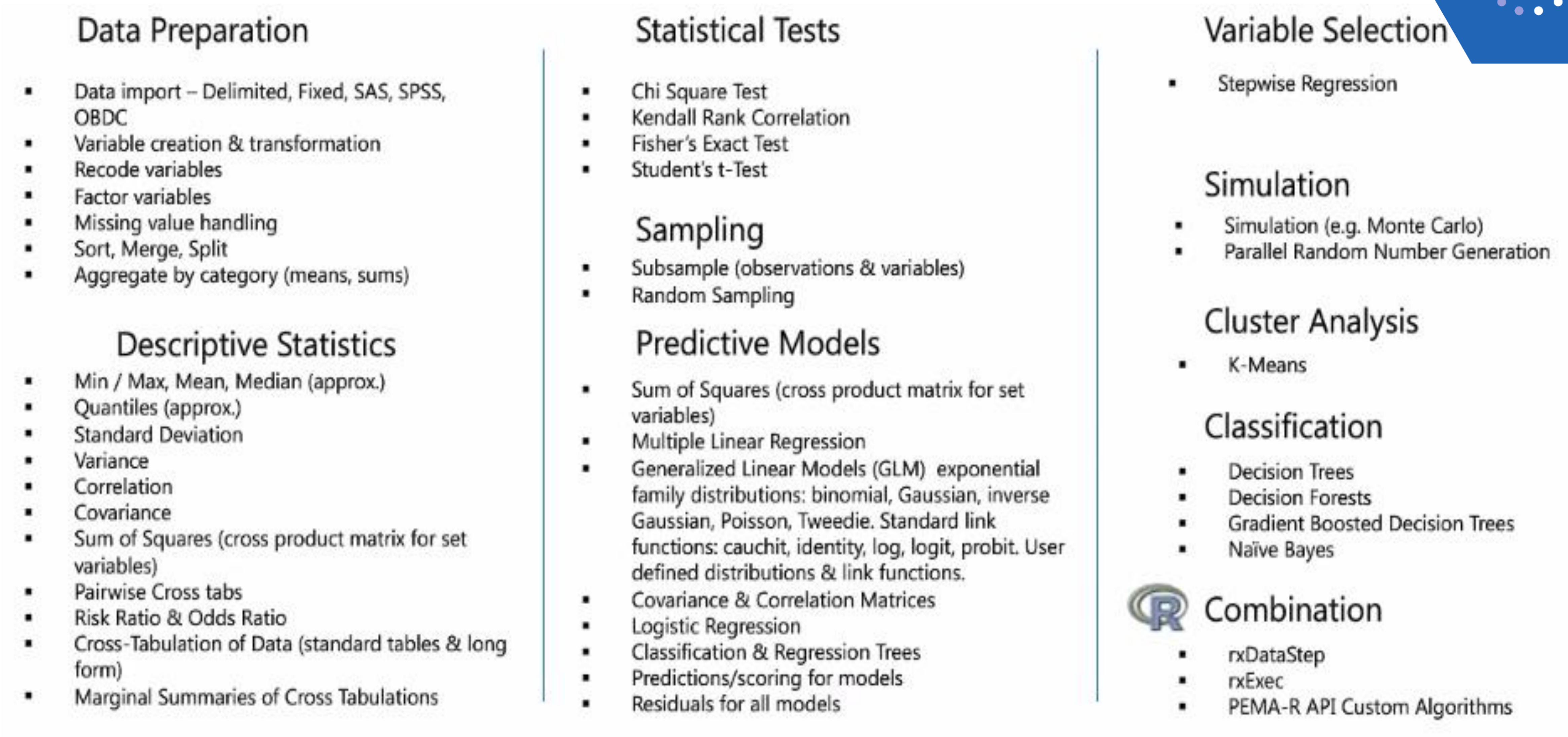
- Normalization layer for ScaleR algoritms (SQLServer,Win, Lin, TeraData, Hadoop, HDI)

ScaleR

- Typical statistical approaches refactored for parallel computation
- Block-wise computation; No In-Memory constraints



ScaleR algorithms



Data Preparation

- Data import – Delimited, Fixed, SAS, SPSS, OBDC
- Variable creation & transformation
- Recode variables
- Factor variables
- Missing value handling
- Sort, Merge, Split
- Aggregate by category (means, sums)

Descriptive Statistics

- Min / Max, Mean, Median (approx.)
- Quantiles (approx.)
- Standard Deviation
- Variance
- Correlation
- Covariance
- Sum of Squares (cross product matrix for set variables)
- Pairwise Cross tabs
- Risk Ratio & Odds Ratio
- Cross-Tabulation of Data (standard tables & long form)
- Marginal Summaries of Cross Tabulations

Statistical Tests

- Chi Square Test
- Kendall Rank Correlation
- Fisher's Exact Test
- Student's t-Test

Sampling

- Subsample (observations & variables)
- Random Sampling

Predictive Models

- Sum of Squares (cross product matrix for set variables)
- Multiple Linear Regression
- Generalized Linear Models (GLM) exponential family distributions: binomial, Gaussian, inverse Gaussian, Poisson, Tweedie. Standard link functions: cauchit, identity, log, logit, probit. User defined distributions & link functions.
- Covariance & Correlation Matrices
- Logistic Regression
- Classification & Regression Trees
- Predictions/scoring for models
- Residuals for all models

Variable Selection

- Stepwise Regression

Simulation

- Simulation (e.g. Monte Carlo)
- Parallel Random Number Generation

Cluster Analysis

- K-Means

Classification

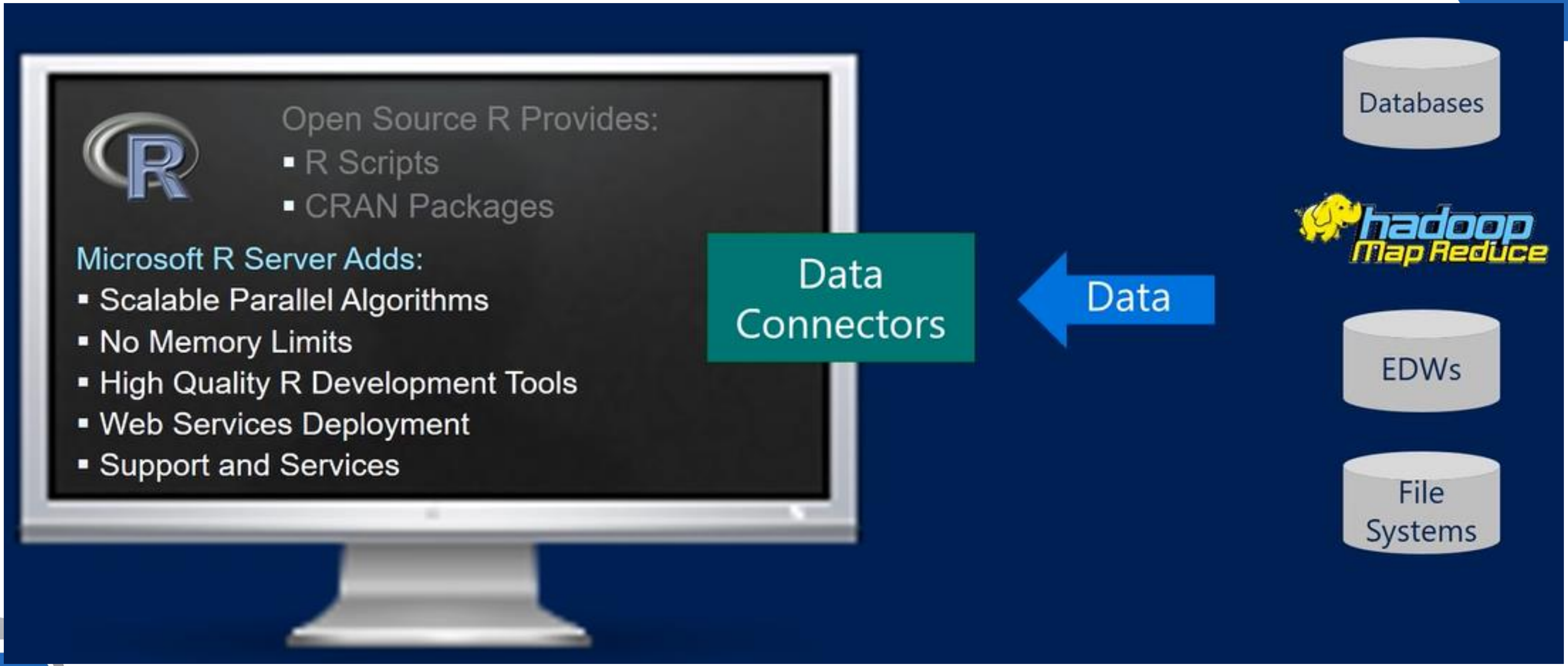
- Decision Trees
- Decision Forests
- Gradient Boosted Decision Trees
- Naïve Bayes

Combination

- rxDataStep
- rxExec
- PEMA-R API Custom Algorithms

Microsoft

Microsoft R Server - architecture



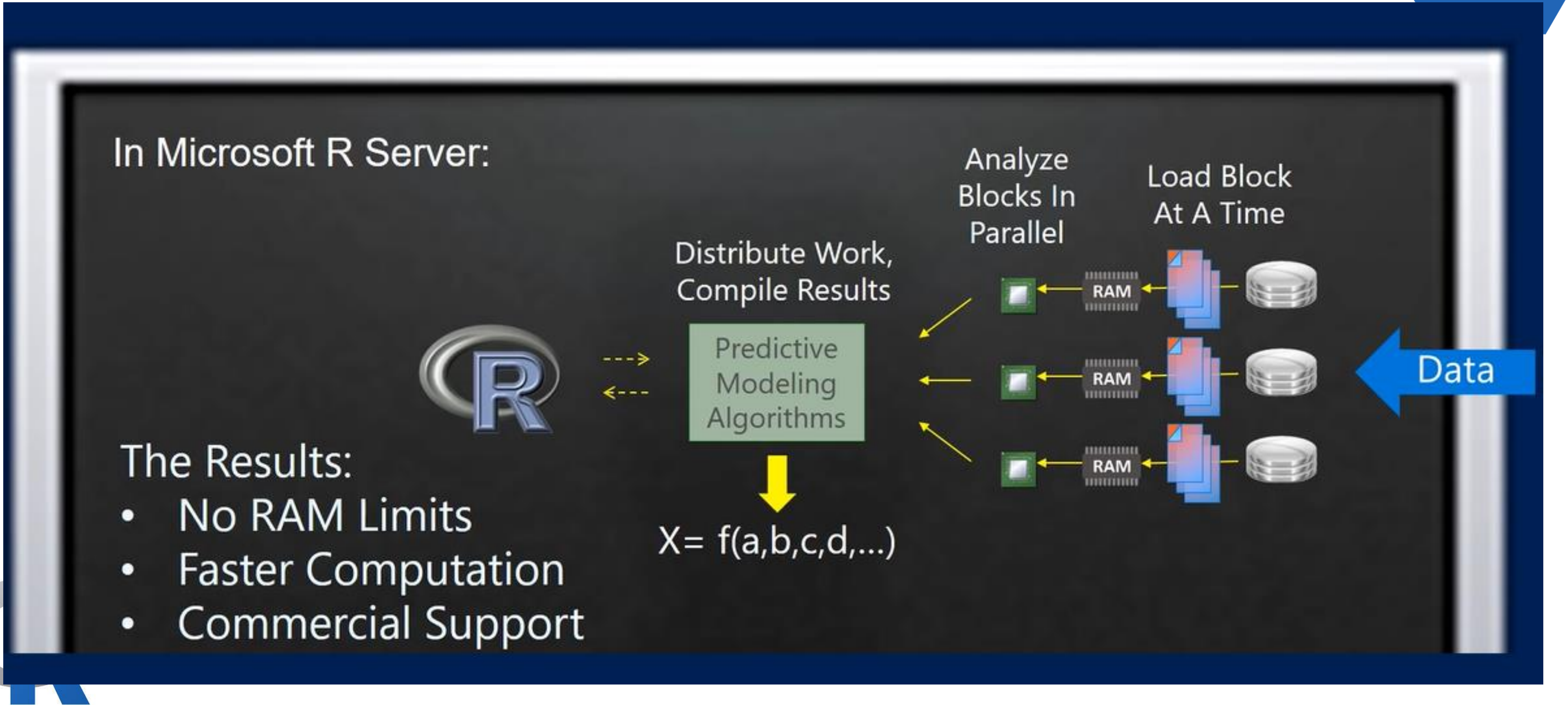
Open Source R Provides:

- R Scripts
- CRAN Packages

Microsoft R Server Adds:

- Scalable Parallel Algorithms
- No Memory Limits
- High Quality R Development Tools
- Web Services Deployment
- Support and Services

Parallelizing data process



In Microsoft R Server:

Distribute Work, Compile Results
Predictive Modeling Algorithms
X= f(a,b,c,d,...)
Analyze Blocks In Parallel
Load Block At A Time
RAM
RAM
RAM
Data

The Results:

- No RAM Limits
- Faster Computation
- Commercial Support

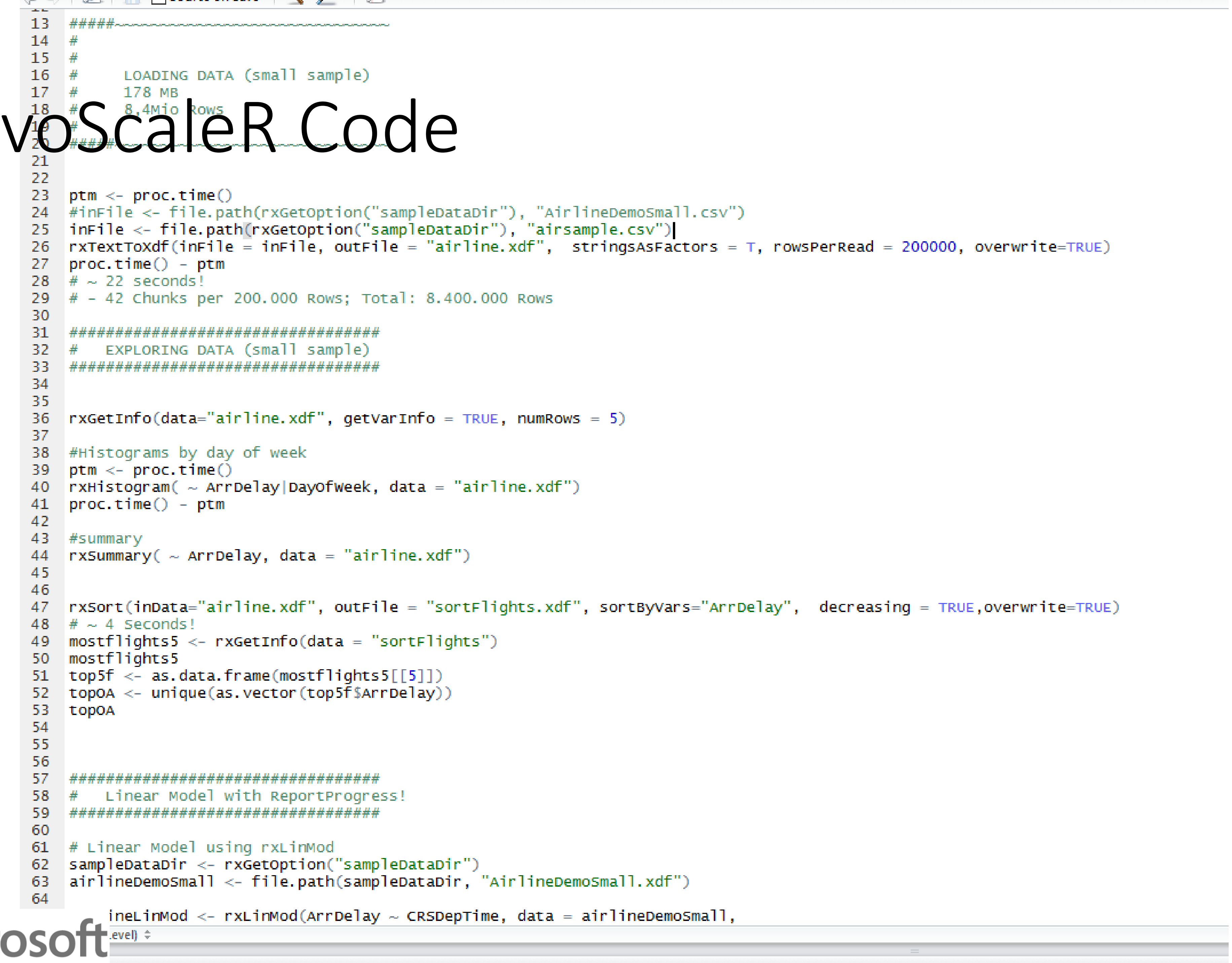
R code in SQL Server as T-SQL

```
EXECUTE sp_execute_external_script
   @language = N'R'
  ,@script = N'
               library(e1071);
               irismodel <-naiveBayes(iris_data[,1:4], iris_data[,5]);
               trained_model <- data.frame(payload = as.raw(serialize(irismodel, connection=NULL)));'
  ,@input_data_1 = N'select "Sepal.Length", "Sepal.Width","Petal.Length","Petal.Width","Species" from iris_data'
  ,@input_data_1_name = N'iris_data'
  ,@output_data_1_name = N'trained_model'

WITH RESULT SETS ((model VARBINARY(MAX)));
```

R code in SQL Server using Scale R algorithms

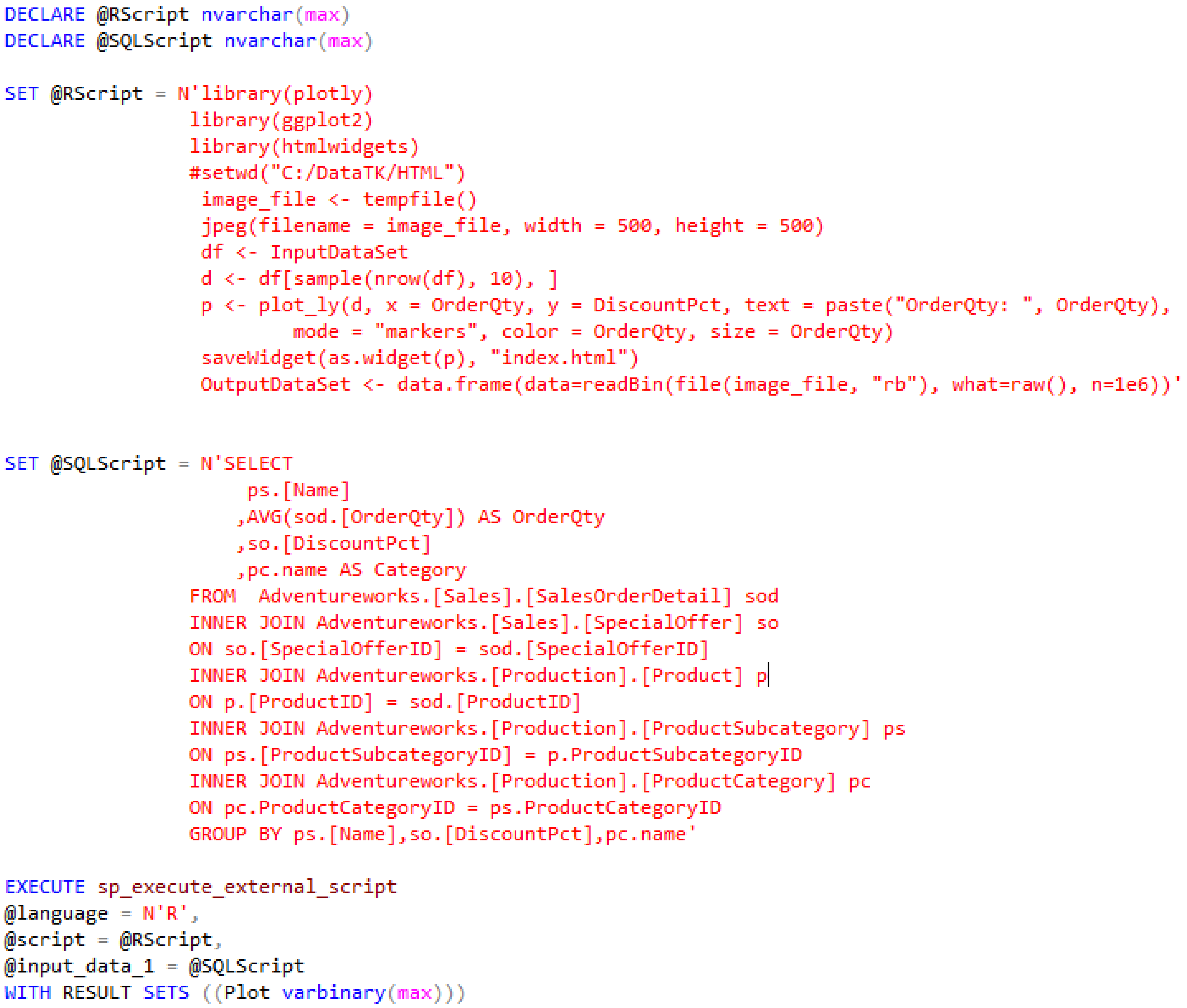
```
EXECUTE sp_execute_external_script
     @language = N'R'
    ,@script = N'require("RevoScaleR");
                 irisLinMod <- rxLinMod(Sepal.Length ~ Sepal.Width + Petal.Length + Petal.Width + Species,
                         data = iris_rx_data);
                 trained_model <- data.frame(payload = as.raw(serialize(irisLinMod, connection=NULL)));'
    ,@input_data_1 = N'select "Sepal.Length", "Sepal.Width", "Petal.Length", "Petal.Width", "Species" from
                          iris_rx_data'
    ,@input_data_1_name = N'iris_rx_data'
    ,@output_data_1_name = N'trained_model'

WITH result SETS ((model VARBINARY(MAX)));
```

RevoScaleR Code



```
#####~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#
#
#      LOADING DATA (small sample)
#      178 MB
#      8.4Mio Rows
#
#####~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


ptm <- proc.time()
#inFile <- file.path(rxGetOption("sampleDataDir"), "AirlineDemoSmall.csv")
inFile <- file.path(rxGetOption("sampleDataDir"), "airsample.csv")
rxTextToXdf(inFile = inFile, outFile = "airline.xdf",  stringsAsFactors = T, rowsPerRead = 200000, overwrite=TRUE)
proc.time() - ptm
# ~ 22 seconds!
# - 42 Chunks per 200.000 Rows; Total: 8.400.000 Rows

##################################
#   EXPLORING DATA (small sample)
##################################


rxGetInfo(data="airline.xdf", getVarInfo = TRUE, numRows = 5)

#Histograms by day of week
ptm <- proc.time()
rxHistogram( ~ ArrDelay|DayOfWeek, data = "airline.xdf")
proc.time() - ptm

#summary
rxSummary( ~ ArrDelay, data = "airline.xdf")


rxSort(inData="airline.xdf", outFile = "sortFlights.xdf", sortByVars="ArrDelay",  decreasing = TRUE,overwrite=TRUE)
# ~ 4 Seconds!
mostflights5 <- rxGetInfo(data = "sortFlights")
mostflights5
top5f <- as.data.frame(mostflights5[[5]])
topOA <- unique(as.vector(top5f$ArrDelay))
topOA



##################################
#   Linear Model with ReportProgress!
##################################

# Linear Model using rxLinMod
sampleDataDir <- rxGetOption("sampleDataDir")
airlineDemoSmall <- file.path(sampleDataDir, "AirlineDemoSmall.xdf")

    ineLinMod <- rxLinMod(ArrDelay ~ CRSDepTime, data = airlineDemoSmall,
```

R code in SQL Server as T-SQL to generate graphs



```
DECLARE @RScript nvarchar(max)
DECLARE @SQLScript nvarchar(max)

SET @RScript = N'library(plotly)
                library(ggplot2)
                library(htmlwidgets)
                #setwd("C:/DataTK/HTML")
                 image_file <- tempfile()
                 jpeg(filename = image_file, width = 500, height = 500)
                 df <- InputDataSet
                 d <- df[sample(nrow(df), 10), ]
                 p <- plot_ly(d, x = OrderQty, y = DiscountPct, text = paste("OrderQty: ", OrderQty),
                         mode = "markers", color = OrderQty, size = OrderQty)
                 saveWidget(as.widget(p), "index.html")
                 OutputDataSet <- data.frame(data=readBin(file(image_file, "rb"), what=raw(), n=1e6))'


SET @SQLScript = N'SELECT
                     ps.[Name]
                    ,AVG(sod.[OrderQty]) AS OrderQty
                    ,so.[DiscountPct]
                    ,pc.name AS Category
                FROM  Adventureworks.[Sales].[SalesOrderDetail] sod
                INNER JOIN Adventureworks.[Sales].[SpecialOffer] so
                ON so.[SpecialOfferID] = sod.[SpecialOfferID]
                INNER JOIN Adventureworks.[Production].[Product] p
                ON p.[ProductID] = sod.[ProductID]
                INNER JOIN Adventureworks.[Production].[ProductSubcategory] ps
                ON ps.[ProductSubcategoryID] = p.ProductSubcategoryID
                INNER JOIN Adventureworks.[Production].[ProductCategory] pc
                ON pc.ProductCategoryID = ps.ProductCategoryID
                GROUP BY ps.[Name],so.[DiscountPct],pc.name'

EXECUTE sp_execute_external_script
@language = N'R',
@script = @RScript,
@input_data_1 = @SQLScript
WITH RESULT SETS ((Plot varbinary(max)))
```

Benefits of R integration

- Based on Open source R
- Different versions available (Open, Client and Server)
- Distributed workloads, multi-threading and parallelization
- Interoperable (Windows, Linux, MacOS) with different flavors (Hadoop, Teradata, HDInsight)
- Faster model prediction and model deployment
- No „in-memory" constraints, less data movement, less bottlenecks in performance, no data size limitations
- Hybrid topologies, agile development, stable platform for data operationalization, investment protection (SLA, Terms and agreements)
- R Code is available in SSMS environment
- Community and commercial support
- R Language is growing in popularity

Questions?

Contacts:

Email: tomaz.kastrun@gmail.com
Twitter: @tomaz_tsql
Blog: https://tomaztsql.wordpress.com