



SQLJava

Yellowfin

	100%
1-9	1-91
10	10
N	N
N	N
9-1	9-11
	-/
	2
	2
	2
%	%
	2
	e
Null	Null
%	%
%	
%	%
%	%
%	%
%	%
	2

10	10
N	N
N	NN
/N%	N%
	N

	2
	2
	2

	10-1
Deviation	
Linear Regression	
Mean	
Median	
Mode	
Moving Average	
	22
Moving Total	N
Naïve Forecasting	1tt-1
Polynomial Regression	23
Quartile	4-1
Standard Deviation	
Standard Score	
Stepped Regression	
Trend	
Triple Exponential Smoothing	
Variance	
Weighted Moving Average	

	2
--	---

1,000

-R

R

Rserve R

YellowfinR

1. **Yellowfin-R jar**Yellowfin/appserver/webapps/ROOT/WEB-INF/lib
2. **R**
3. **Rserve**RserveTCP/IPRRR install.packages("Rserve")
install.packages("magrittr")
install.packages("rattle") (linux requires special installation)
4. **Rserve**YellowfinRRserveR
library(Rserve)
Rserve()
5. **RRSCRIPT_PATH**RRSCRIPT_PATH
RSCRIPT_PATH=/home/foo/Rscripts

R

YellowfinR

<R_file_name>.RYellowfin<R_file_name>.R.input.csvR<R_file_name>.R.result.csv

Neural NetworksRR

Sample R-Script : neural-net-script.R

```
setwd("C:/R/R-3.2.3/bin/x64")
library(rattle) #
To access the weather dataset and utility commands.
library(magrittr) # For the
%>% and %<>% operators.
building <- TRUE
scoring <- ! building
# A pre-defined value is used
to reset the random seed so that results are repeatable.
crv$seed <- 42
# Load the data.
rPATH <-
Sys.getenv("RSCRIPT_PATH")
rINPUT <- paste0(rPATH, "/neural-net-script.r.input.csv")
rOUTPUT <- paste0(rPATH
, "/neural-net-script.r.result.csv")
dataset <-
read.csv(file=rINPUT, header=FALSE, sep=",")
# Note the user
selections.
# Build the
training/validate/test datasets.
set.seed(crv$seed)
crs$nobs <- nrow(dataset) #
366 observations
crs$sample <- crs$train
<- sample(nrow(dataset), 0.7*crs$nobs) # 256 observations
crs$validate <-
sample(setdiff(seq_len(nrow(dataset)), crs$train), 0.15*crs$nobs) # 54
observations
crs$test <-
setdiff(setdiff(seq_len(nrow(dataset)), crs$train), crs$validate) # 56
observations
# The following variable
selections have been noted.
crs$input <-
c("V1", "V2", "V3", "V4", "V5")
crs$target
<- "V6"
#=====
# Neural Network
#=====
# Build a neural network model
using the nnet package.
library(nnet, quietly=TRUE)
# Build the NNet model.

set.seed(199)
crs$nnet <-
nnet(as.factor(V6) ~ ., data=dataset[crs$sample, c(crs$input, crs$target)], size=10,
skip=TRUE, MaxNWts=10000, trace=FALSE, maxit=100)
#=====
# Score a dataset.
#=====
# Obtain probability scores for
the Neural Net model on weather.csv [validate].
#crs$pr <- predict(crs$nnet,
newdata=dataset[crs$validate, c(crs$input)], type="class")
#crs$pr <- predict(crs$nnet,
newdata=dataset[crs$validate, c(crs$input)], type="class")
crs$pr <- predict(crs$nnet,
newdata=dataset, type="class")
write.table(crs$pr,
file=rOUTPUT, row.names=FALSE, col.names = FALSE)
```

R

Advanced Metrics

Select Function:
Rserve R

Search

Apply R-Script

Apply R-Script

This function will invoke an R-script (in a separate file) which is pointed to by a parameter. The R-script will return a result value which will be included in the report

Attribute	Setting	User Prompt
R Script File Name	scriptName.r	Name Of The R Script to Invoke
Value For MinTemp	MinTemp	First Parameter
Value For MaxTemp	MaxTemp	Second Parameter
Value For Rainfall	Rainfall	Third Parameter
Value For Evaporation	Evaporation	Fourth Parameter
Rain Today	RainToday	Fifth Parameter
Value For RainTomorrow	RainTomorrow	Sixth Parameter

Step 1: Use "Apply R-Script" to convert to

-

210

1.

a. /

カラム(列)	Year	Σ Sum Invoiced ...
ロウ(行)		

b.

Year	Sum Invoiced Amount
2009	\$17,633,473
2010	\$8,611,470
2011	\$11,012,244
2012	\$81,690,100
2013	\$158,353,519
2014	\$152,912,577
2015	\$28,207,858
2016	\$12,522,605

2.

3.

高度な関数

高度な関数データ変換

保存キャンセル

Invoiced Amount (数値)

Σ

#

#!

+

%

-

⊗

●

○

○

○

○

○

○

グラフにのみ表示

○

関数の選択:

分析

▼

検索

昇順(1-9)ラック

最上位から 10位

最上位から N位

最上位から N位(同順位を含む)

最下位から 10位

最下位から N位

最上位から N位

選択したフィールドの最上位の値を返します

属性

設定

ユーザープロンプト

最上位

10

○

表示数

ステップ 1: カラムに対する変化百分率(%) を使用し 数値 へ変換

- a.
- b.
- c.
- d.

4.

-

1. +

高度な関数

+

Year

Sum Invoiced Amount

2.
3.

- a.
- b.
- c.
- d.

4.

-

1.

Year ▼	Sum Invoiced Amount ▼
2009	\$17,633,473
2010	\$8,611,470
2011	\$11,012,244
2012	\$81,690,100
2013	\$158,353,519
2014	\$152,912,577
2015	\$28,207,858
2016	\$12,522,605

2.

3.

- a.
- b.
- c.
- d.

4.

1.

2.

3.

4. Java1,000

5.

6.