GIVE ME SOME CREDIT

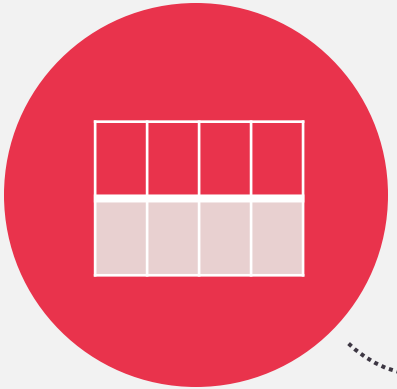
Machine Learning Application

By
Princewill Eneh
Xiaozhao Song
Yalei Yan
Suraj Jois
Ruogu Wang

Project Overview

Objective : This project requires us to improve on the state of the art in credit scoring, by predicting the probability that somebody will experience financial distress in the next two years.

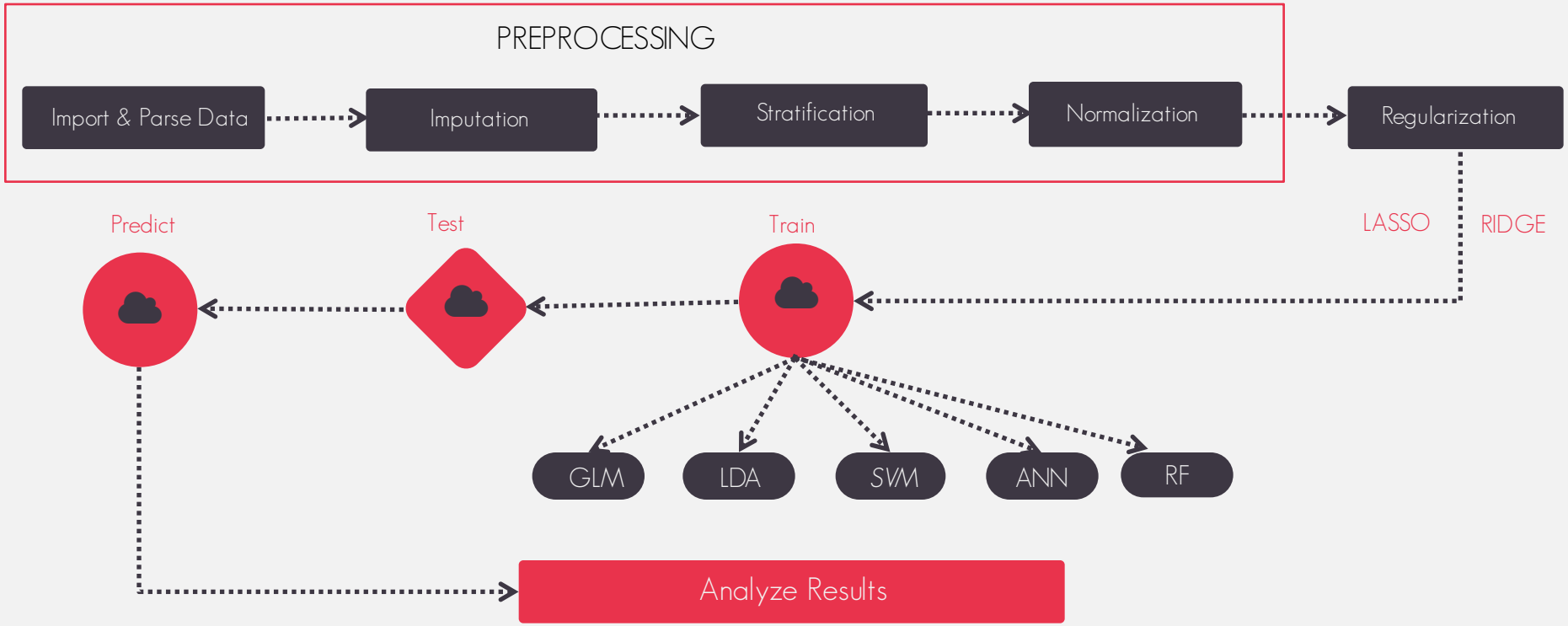
DATASET



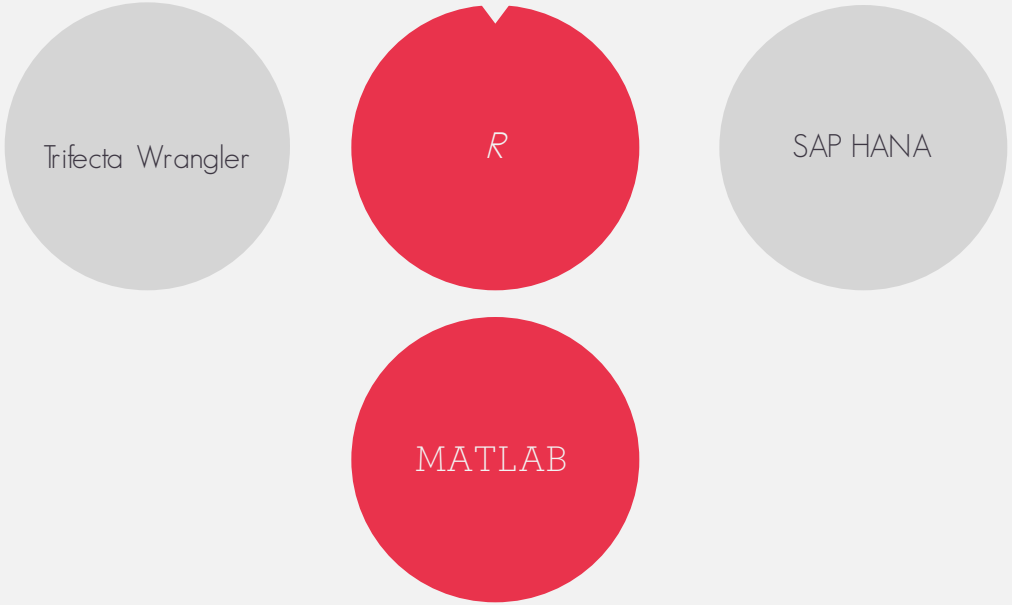
ANALYSIS

PREDICTION

Workflow


PREPROCESSING
Import & Parse Data
Imputation
Stratification
Normalization
Regularization
LASSO
RIDGE
Train
Test
Predict
GLM
LDA
SVM
ANN
RF
Analyze Results

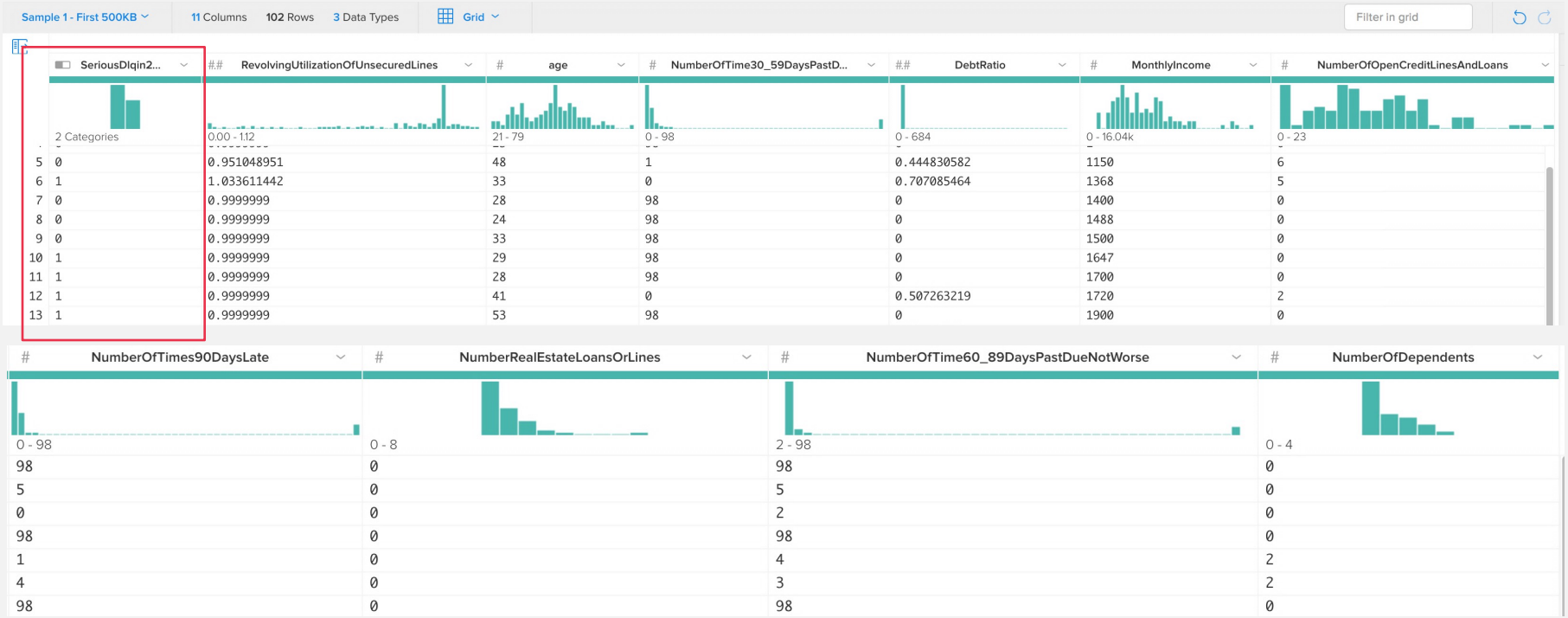
Tools



Preliminary Analysis & Wrangling

Dataset

Import And Parsing

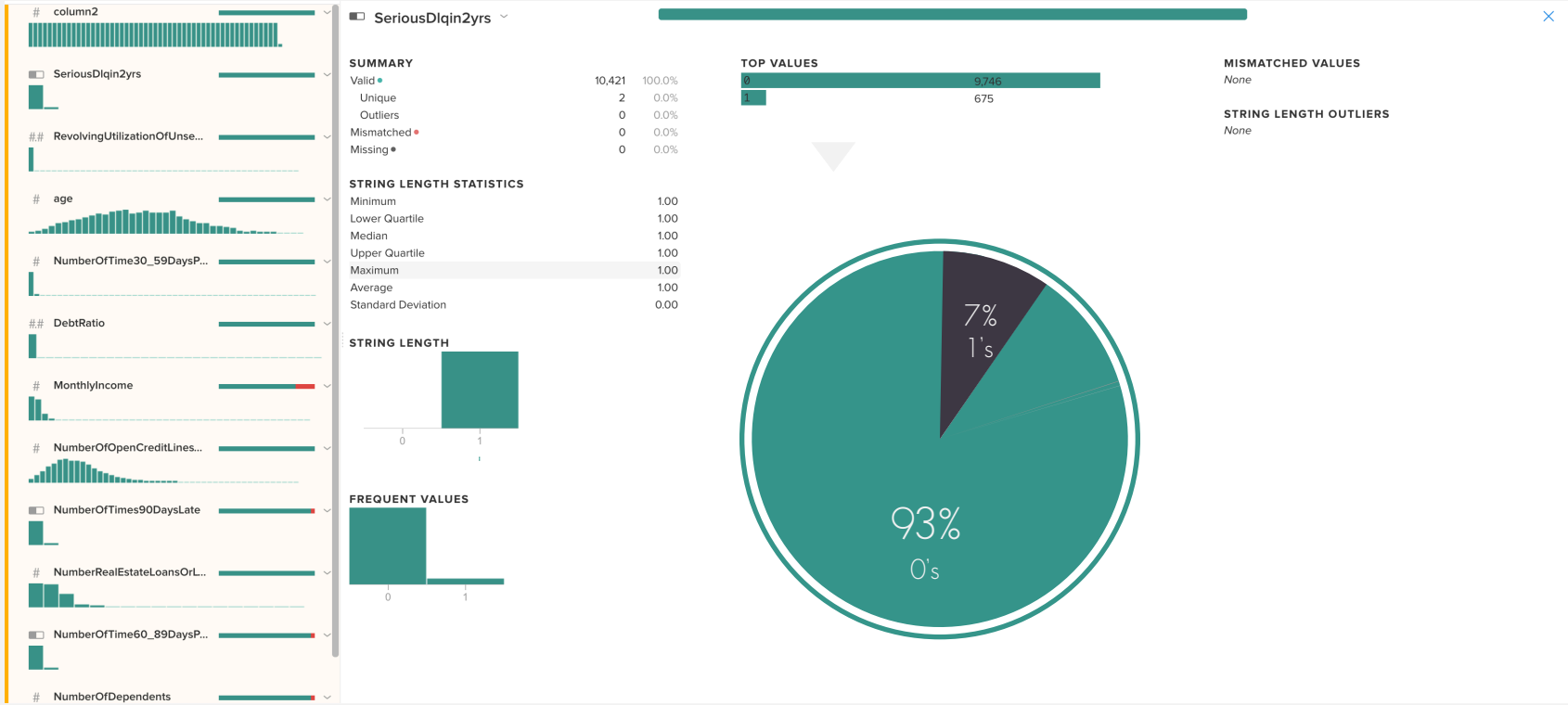


Imputation

Sample 1 - First 500KB | 12 Columns | 10,421 Rows | 3 Data Types | Grid | Filter in grid

Lines	# age	# NumberOfTime30_59DaysPastDueNotWorse	## DebtRatio	# MonthlyIncome	# NumberOfOpenCreditLinesAndLoans	Nu
	21 - 101	0 - 98	0 - 168,835	0 - 208.33k	0 - 46	2 Categories
357	50	0	2720	NA	7	0
358	38	0	0.563328034	1886	3	0
359	35	0	3388	NA	1	0
360	59	0	0.033982276	14330	12	0
361	67	0	0.119463561	5666	9	0
362	30	0	0.698120752	2500	7	0
363	56	0	0.170164956	19580	12	0
364	51	0	0.333583302	8000	16	0
365	40	1	3878	NA	10	0
366	28	0	0.467171717	1187	4	0
367	54	0	0.017655818	4700	7	0
368	46	1	4480	NA	6	0
369	57	0	6381	NA	5	0
370	35	2	0.314566116	3871	8	0
371	79	0	0.009596162	2500	11	0
372	46	0	0.267146571	5000	6	0
373	79	0	1063	NA	7	0
374	64	0	0.214426305	8636	29	0
375	74	0	0.164917541	2000	3	0
376	64	0	0.101514142	10500	20	0
377	63	0	0.387956894	9000	6	0
378	63	1	0.285156885	6150	12	0
379	54	0	0.655589124	3640	7	0
380	63	0	0.278226307	8050	13	0
381	45	0	0.290229092	14491	13	0
382	22	0	130	NA	0	0
383	66	0	870	NA	12	0

Stratification



Normalization

```
train<-train %>% mutate_each_(funs(scale),vars=c("RevolvingUtilizationOfUnsecuredLines","age","NumberOfTime30.59DaysPastDueNotWorse",
                                                 "DebtRatio","MonthlyIncome","NumberOfOpenCreditLinesAndLoans","NumberOfTimes90DaysLate",
                                                 "NumberRealEstateLoansOrLines","NumberOfTime60.89DaysPastDueNotWorse","NumberOfDependents")) # Here you chose what colums you want to normalize
```

RESCALING

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Sample 1 - First 500KB | 12 Columns | 2,593 Rows | 3 Data Types | Grid | Filter in grid

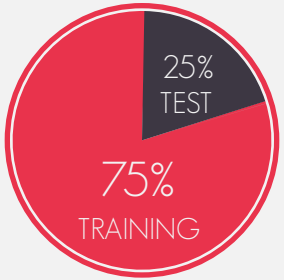
	SeriousDlqin2yrs	RevolvingUtilizationOfUnsecuredLines	age	NumberOfTime30_59DaysPastDueNotWorse	DebtRatio	MonthlyIncome
	2 Categories	-0 - 65	-2.0 - 3.4	-0.2 - 13.4	-0 - 42	-0.9 - 9.8
2	0	-0.0304573083111435	-0.486080352338568	-0.155613210143838	-0.0680294440321644	-0.16397253834217
3	0	-0.0291232339637546	-0.558036449028647	-0.155613210143838	-0.0685594935335708	0.379940962917594
4	1	-0.0241857611321987	-0.0543437721980901	-0.155613210143838	-0.0649046800186877	1.11149612104844
5	0	-0.0311726679362547	1.16890987153326	-0.155613210143838	-0.0687788890341409	-0.281287215084472
6	1	-0.0285634220158335	0.0895684211820691	0.398953782253626	-0.0682647996611607	0.859091816542279
7	1	-0.0247053195915838	-0.558036449028647	0.26031203415426	-0.0688311189951879	0.333571525470044
8	0	-0.0310916880922699	1.16890987153326	-0.155613210143838	-0.0686653950547134	-0.176492286453008
9	0	-0.0303769515386091	-0.629992545718727	-0.155613210143838	-0.0692529801887371	0.982743649735746
10	0	-0.0307899016043629	1.024976781531	-0.155613210143838	-0.0691581490190793	-0.176492286453008
11	1	-0.0245240780511359	-0.12629986888817	0.26031203415426	-0.0685842957924426	-0.516534827735043
12	0	-0.0308696791482916	-0.701948642408807	-0.155613210143838	-0.0684456353111107	-0.0219274949611745
13	0	-0.0311051703084635	0.377392807942388	-0.155613210143838	-0.0690101482889886	-0.210651105372704
14	1	-0.0241857611321987	-1.27759741592944	-0.0169714620444719	-0.0688500567796704	-0.468928871955558
15	1	-0.0241857611321987	-1.27759741592944	-0.155613210143838	-0.0680818985212596	-0.717469056674427
16	1	-0.0294501872226175	0.521305001322547	-0.155613210143838	-0.0680302572442554	-0.269231161348109
17	0	-0.0311609852959052	1.31282206491342	-0.155613210143838	-0.069426362692836	0.743168222923404
18	1	-0.0248864931983301	-1.49346570599968	-0.155613210143838	-0.068210501615343	-0.331057077944842
19	0	-0.0275355759011369	0.233480614562228	-0.155613210143838	-0.0684576001219134	0.548880280018168
20	1	-0.0297149444142985	1.024976781531	-0.155613210143838	-0.0683526502521959	-0.0477398151403108

Training & Test Dataset

SeriousDlqin2yrs	RevolvingUtilizationOfUnsecuredLines	age	NumberOfTime30.59DaysPastDueNotWorse	DebtRatio	MonthlyIncome
0	-0.02418576	0.08956842	-0.15561321	-0.06940961	-0.436161136
1	-0.02418576	-1.27759742	0.12167029	-0.06887661	-0.586398113
1	-0.02877733	-0.12629987	-0.01697146	-0.06872429	-0.603400241
0	-0.02499887	-1.13368522	-0.15561321	-0.06858692	-0.567541209
1	-0.02418576	-0.84586084	-0.15561321	-0.06943009	-0.680682636
1	-0.02881836	-0.84586084	-0.15561321	-0.06907226	-0.763838494
1	-0.02832809	0.08956842	-0.01697146	-0.06819264	0.725084142
1	-0.02418576	-1.56542180	0.12167029	-0.06942334	-0.575887708
1	-0.02432378	-0.62999255	-0.01697146	-0.06804322	-0.238318203
0	-0.03117267	-0.62999255	-0.15561321	-0.06704838	-0.562904265
1	-0.03041684	-0.91781693	-0.15561321	-0.06762905	0.503129102
1	-0.02493384	0.80912939	0.26031203	-0.06834274	0.789228531
0	-0.03090297	-0.12629987	-0.15561321	-0.06858992	0.209919692
0	-0.02524946	-1.06172913	-0.15561321	-0.06916498	-0.009098617
1	-0.02524091	-0.05434377	-0.15561321	-0.06757785	4.202791951
0	-0.02937535	1.45673426	-0.01697146	-0.06864426	0.016713703
0	-0.03090376	-0.48608035	-0.15561321	-0.06609814	-0.598454167
1	-0.02425029	-0.91781693	0.12167029	-0.06844727	-0.312972997
0	-0.03117267	1.09695377	-0.15561321	-0.06910493	-0.566768385
0	-0.03088944	-1.06172913	-0.15561321	-0.06767820	-0.385773014
1	-0.02396430	-0.70194864	-0.01697146	-0.06829033	0.403125682
1	-0.02799688	-0.34216816	-0.15561321	-0.06650553	-0.331057078
0	-0.03046536	-0.12629987	-0.15561321	-0.06867505	0.673614067
0	-0.02999025	-0.12629987	-0.15561321	-0.06880952	-0.176492286
1	-0.02485098	0.08956842	0.12167029	-0.06556223	-0.060259563
1	-0.02200017	-1.42150961	-0.15561321	-0.06869171	-0.681300895
0	-0.03022052	0.52130500	-0.15561321	-0.06814216	0.106824976
1	-0.02474492	0.01761232	-0.01697146	-0.06551404	-0.671099619
0	-0.02991334	0.44934890	-0.15561321	-0.06892262	1.304702110
1	-0.02948031	0.52130500	-0.01697146	-0.06561821	-0.843593927
1	-0.02902450	-1.56542180	-0.15561321	-0.06924146	0.256289130
1	-0.03117267	-1.70933400	-0.15561321	-0.06943009	-0.810671626

1 to 33 of 8,357 entries

Training Data



SeriousDlqin2yrs	RevolvingUtilizationOfUnsecuredLines	age	NumberOfTime30.59DaysPastDueNotWorse	D
	-0.03045731	-0.48608035	-0.15561321	-
	-0.02418576	-0.05434377	-0.15561321	-
	-0.02856342	0.08956842	0.39895378	-
	-0.02470532	-0.55803645	0.26031203	-
	-0.02418576	-1.27759742	-0.01697146	-
	-0.02945019	0.52130500	-0.15561321	-
	-0.03116099	1.31282206	-0.15561321	-
	-0.02824325	-0.62999255	-0.15561321	-
	-0.03117267	0.95304158	-0.15561321	-
	-0.03087064	1.24086597	-0.15561321	-
	-0.03113750	1.88847084	-0.15561321	-
	-0.03085230	2.03238303	-0.15561321	-
	-0.02798376	-0.27021206	-0.15561321	-
	-0.02389485	0.80912939	-0.01697146	-
	-0.02459547	-0.34216816	0.12167029	-
	-0.02928255	-0.55803645	-0.15561321	-
	-0.02418576	-0.05434377	-0.01697146	-
	-0.03117267	0.16152452	0.12167029	-
	-0.03064567	-1.20564132	-0.15561321	-
	-0.03108630	1.81651474	0.12167029	-
	-0.03074099	1.09695377	-0.15561321	-
	-0.03111532	-0.55803645	-0.01697146	-
	-0.03109881	0.73717329	-0.15561321	-
	-0.02510237	-1.27759742	0.39895378	-
	-0.02426012	-0.05434377	-0.15561321	-
	-0.03062240	-1.13368522	-0.15561321	-
	-0.03085472	0.66521719	-0.15561321	-
	-0.03117267	-1.92520229	-0.15561321	-
	-0.03117267	1.31282206	-0.15561321	-
	-0.03106270	0.23348061	-0.15561321	-
	-0.03111868	1.67260255	-0.15561321	-
	-0.02551226	-0.27021206	-0.01697146	-

ing 1 to 33 of 8,357 entries

Test Data

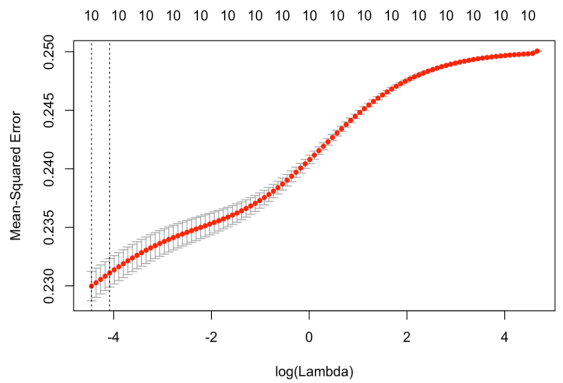
Regularization

RIDGE

```
x.tr <- model.matrix(SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines  + age + NumberOfTime30.59DaysPastDu
eNotWorse
                     + DebtRatio  + MonthlyIncome +NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
                       NumberRealEstateLoansOrLines +  NumberOfTime60.89DaysPastDueNotWorse
                     + NumberOfDependents, data = training_data)[, -1]
y.tr <- training_data$SeriousDlqin2yrs


x.val <- model.matrix(SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines  + age + NumberOfTime30.59DaysPastD
ueNotWorse
                      + DebtRatio  + MonthlyIncome +NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
                        NumberRealEstateLoansOrLines +  NumberOfTime60.89DaysPastDueNotWorse
                      + NumberOfDependents, data = testing_data)[, -1]
y.val <- testing_data$SeriousDlqin2yrs
```

Cross Validation to obtain best lambda

```
set.seed(10)
rr.cv <- cv.glmnet(x.tr, y.tr, alpha = 0)
```

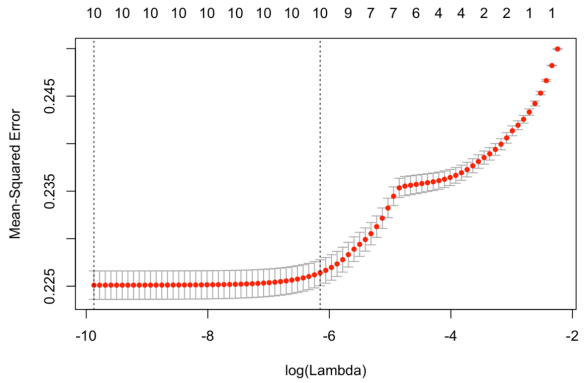


LASSO

```
x.tr <- model.matrix(SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines  + age + NumberOfTime30.59DaysPastDu
eNotWorse
                     + DebtRatio  + MonthlyIncome +NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
                       NumberRealEstateLoansOrLines +  NumberOfTime60.89DaysPastDueNotWorse
                     + NumberOfDependents, data = training_data)[, -1]
y.tr <- training_data$SeriousDlqin2yrs


x.val <- model.matrix(SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines  + age + NumberOfTime30.59DaysPastD
ueNotWorse
                      + DebtRatio  + MonthlyIncome +NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
                        NumberRealEstateLoansOrLines +  NumberOfTime60.89DaysPastDueNotWorse
                      + NumberOfDependents, data = testing_data)[, -1]
y.val <- testing_data$SeriousDlqin2yrs
```

CV to obtain best lambda

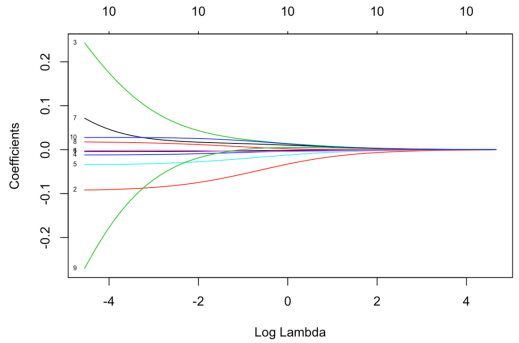
```
set.seed(10)
las.cv <- cv.glmnet(x.tr, y.tr, alpha = 1)
```



Regularization Contd.

RIDGE

```
rr.fit <- glmnet(x.tr, y.tr, alpha = 0)
```




```
rr.pred <- predict(rr.fit, s = rr.bestlam, newx = x.v
rr.pred  <- ifelse(rr.pred  > 0.5,1,0)
```

```
table(rr.pred ,testing_data[,1 ])
```

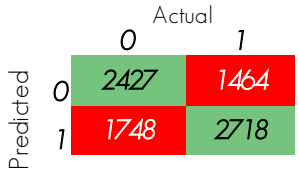
```
##
## rr.pred    0    1
##        0 2427 1464
##        1 1748 2718
```

```
MSE <-mean(rr.pred != testing_SeriousDlqin2yrs)
MSE
```

```
## [1] 0.3843485
```

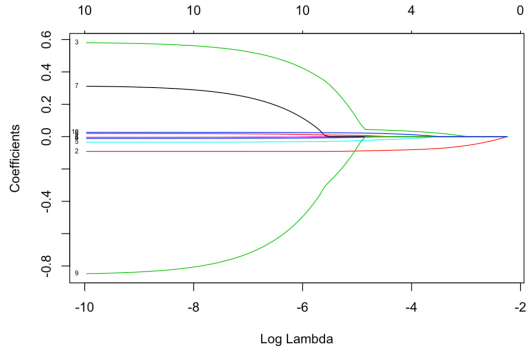
```
print(paste('Accuracy',(1-MSE)*100,"%"))
```

```
## [1] "Accuracy 61.5651549599138 %"
```



LASSO

```
las.fit <- glmnet(x.tr, y.tr, alpha = 1)
```




```
las.pred <- predict(las.fit, s = las.bestlam, newx = x.

las.pred <- ifelse(las.pred > 0.5,1,0)
table(las.pred,testing_data[,1 ])
```

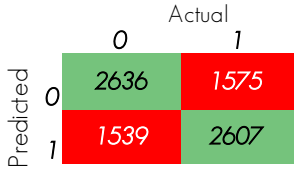
```
##
## las.pred    0    1
##        0 2636 1575
##        1 1539 2607
```

```
MSE <-mean(las.pred != testing_SeriousDlqin2yrs)
MSE
```

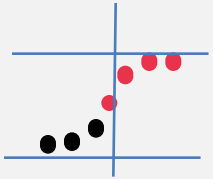
```
## [1] 0.3726218
```

```
print(paste('Accuracy',(1-MSE)*100,"%"))
```

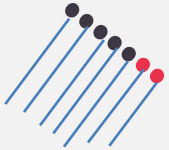
```
## [1] "Accuracy 62.7378245781979 %"
```



Algorithms

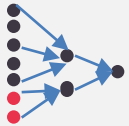


GLM



LDA

SVM



ANN

RF

Logistic Regression

TRAIN

```
glmdl <- glm(SeriousDlqin2yrs ~.,family=binomial(link='logit'),data=training_data)
```

TEST

```
glmdlpredict <- predict(glmdl,testing_data,type='response')
glmdlpredict <- ifelse(glmdlpredict > 0.5,1,0)
```

MODEL

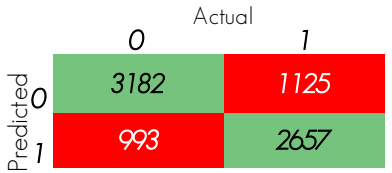
```
summary(glmdl)
```

```
##
## Call:
## glm(formula = SeriousDlqin2yrs ~ ., family = binomial(link = "logit"),
##     data = training_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -8.4904  -1.0161  -0.2558   1.0741   3.1066
##
## Coefficients:
##                                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)                             0.49780    0.03893  12.789  < 2e-16
## RevolvingUtilizationOfUnsecuredLines -0.01224    0.02210  -0.554 0.579748
## age                                  -0.37683    0.02539 -14.841  < 2e-16
## NumberOfTime30.59DaysPastDueNotWorse  3.62786    0.20618  17.596  < 2e-16
## DebtRatio                            -0.06161    0.02726  -2.260 0.023799
## MonthlyIncome                        -0.20013    0.03543  -5.648 1.62e-08
## NumberOfOpenCreditLinesAndLoans       0.01395    0.02793   0.500 0.617396
## NumberOfTimes90DaysLate               3.18999    0.26769  11.917  < 2e-16
## NumberRealEstateLoansOrLines          0.11153    0.02835   3.934 8.35e-05
## NumberOfTime60.89DaysPastDueNotWorse -0.53621    0.32775  -1.636 0.101827
## NumberOfDependents                    0.09467    0.02452   3.861 0.000113
##
```

RESULT

```
confusionMatrix(glmdlpredict, testing_SeriousDlqin2yrs)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction    0    1
##          0 3182 1525
##          1  993 2657
##
##               Accuracy : 0.6987
##                 95% CI : (0.6887, 0.7085)
##    No Information Rate : 0.5004
##    P-Value [Acc > NIR] : < 2.2e-16
##
##                  Kappa : 0.3975
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.7622
##             Specificity : 0.6353
##          Pos Pred Value : 0.6760
##          Neg Pred Value : 0.7279
##              Prevalence : 0.4996
##          Detection Rate : 0.3808
##    Detection Prevalence : 0.5632
##       Balanced Accuracy : 0.6987
##
##        'Positive' Class : 0
##
```



Linear Discriminant Analysis

TRAIN

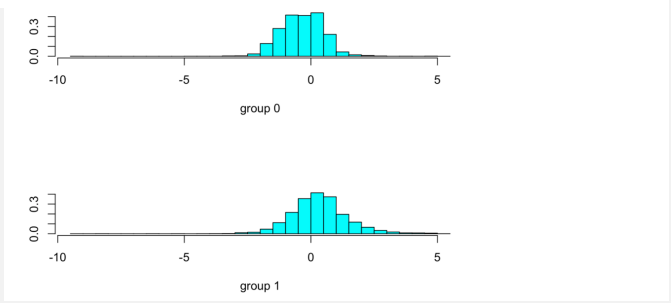
```
ldamdl<-lda(SeriousDlqin2yrs ~., data = training_data)
ldamdl
```

TEST

```
ldamdlpredict<-predict(ldamdl, newdata = testing_data[,c(2,3,4,5,6,7,8,9,10,11)])$class
```

MODEL

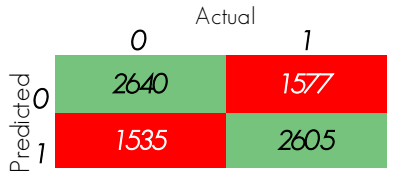
```
##
## Coefficients of linear discriminants:
##                                        LD1
## RevolvingUtilizationOfUnsecuredLines -0.02546800
## age                                  -0.60302787
## NumberOfTime30.59DaysPastDueNotWorse  3.88982613
## DebtRatio                            -0.07301906
## MonthlyIncome                        -0.22669692
## NumberOfOpenCreditLinesAndLoans      -0.03388304
## NumberOfTimes90DaysLate               2.10898015
## NumberRealEstateLoansOrLines          0.12768022
## NumberOfTime60.89DaysPastDueNotWorse -5.70389196
## NumberOfDependents                    0.17499667
```



RESULT

```
confusionMatrix(ldamdlpredict, testing_SeriousDlqin2yrs)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2640 1577
##          1 1535 2605
##
##                Accuracy : 0.6276
##                  95% CI : (0.6171, 0.638)
##     No Information Rate : 0.5004
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.2552
##  Mcnemar's Test P-Value : 0.4624
##
##             Sensitivity : 0.6323
##             Specificity : 0.6229
##          Pos Pred Value : 0.6260
##          Neg Pred Value : 0.6292
##              Prevalence : 0.4996
##          Detection Rate : 0.3159
##    Detection Prevalence : 0.5046
##       Balanced Accuracy : 0.6276
##
##       'Positive' Class : 0
##
```



Support Vector Machine

TRAIN

```
svmmdl <- ksvm(SeriousDlqin2yrs ~ ., data=training_data, type = "C-svc", kernel = "rbfdot",kpar = list(sigma = 0.1), C = 10, prob.model = TRUE)
```

TEST

```
svmmdlpredict <- predict(svmmdl,testing_data[,c(2,3,4,5,6,7,8,9,10,11)])
table(svmmdlpredict,testing_data[,1 ])
```

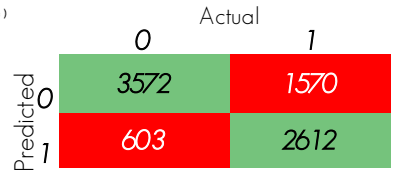
MODEL

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 10
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.1
##
## Number of Support Vectors : 5150
##
## Objective Function Value :  -40507.17
## Training error : 0.249252
## Probability model included.
```

RESULT

```
confusionMatrix(svmmdlpredict, testing_SeriousDlqin2yrs)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3572 1570
##          1  603 2612
##
##                Accuracy : 0.74
##                  95% CI : (0.7304, 0.7494)
##     No Information Rate : 0.5004
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4801
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8556
##             Specificity : 0.6246
##          Pos Pred Value : 0.6947
##          Neg Pred Value : 0.8124
##              Prevalence : 0.4996
##          Detection Rate : 0.4274
##    Detection Prevalence : 0.6153
##       Balanced Accuracy : 0.7401
##
##        'Positive' Class : 0
##
```



Artificial Neural Network R

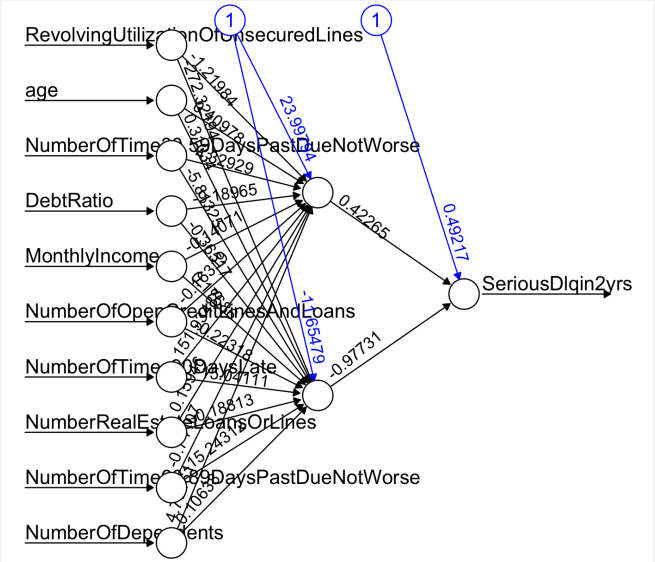
TRAIN

```
nnmdl <- neuralnet(SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines  + age + NumberOfTime30.59DaysPastDueN
otWorse
                   + DebtRatio  + MonthlyIncome +NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
                     NumberRealEstateLoansOrLines +  NumberOfTime60.89DaysPastDueNotWorse
                   + NumberOfDependents, data = training_data, hidden=2, threshold=0.01)
```

TEST

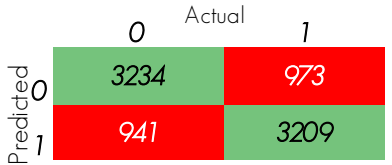
```
nnmdlpredict <- compute(nnmdl, testing_data[,c(2,3,4,5,6,7,8,9,10,11)])

results <- data.frame(actual = testing_SeriousDlqin2yrs, prediction = nnmdlpredict$net.result)
nnmdlpredict <- round(results$prediction)
```

MODEL



RESULT

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3234  973
##          1  941 3209
##
##                Accuracy : 0.7709704
##                  95% CI : (0.7618078, 0.7799438)
##     No Information Rate : 0.5004188
##     P-Value [Acc > NIR] : < 0.0000000000000022
##
##                   Kappa : 0.5419435
##  Mcnemar's Test P-Value : 0.4785833
##
##             Sensitivity : 0.7746108
##             Specificity : 0.7673362
##          Pos Pred Value : 0.7687188
##          Neg Pred Value : 0.7732530
##              Prevalence : 0.4995812
##          Detection Rate : 0.3869810
##    Detection Prevalence : 0.5034103
##       Balanced Accuracy : 0.7709735
##
##        'Positive' Class : 0
##
```



Artificial Neural Network in Matlab

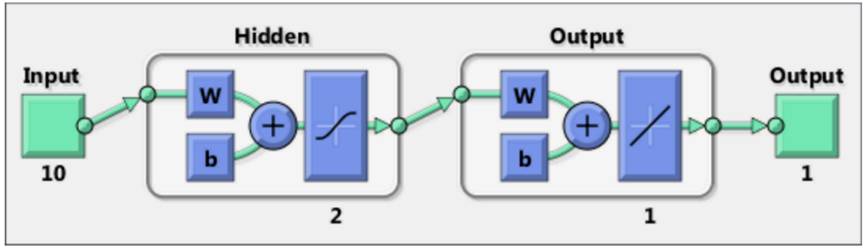
TRAIN

```
% Create a Fitting Network
hiddenLayerSize = 2;
net = fitnet(hiddenLayerSize,trainFcn);

% Setup Division of Data for Training, Validation, Testing
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;

% Train the Network
[net,tr] = train(net,x,t);
```

TEST

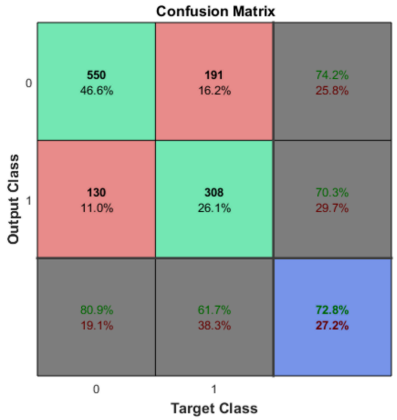
```
testX = x(:,tr.testInd);
testT = t(:,tr.testInd);
```

MODEL



RESULT

```
plotconfusion(testT,testY)
```



Random Forest

TRAIN

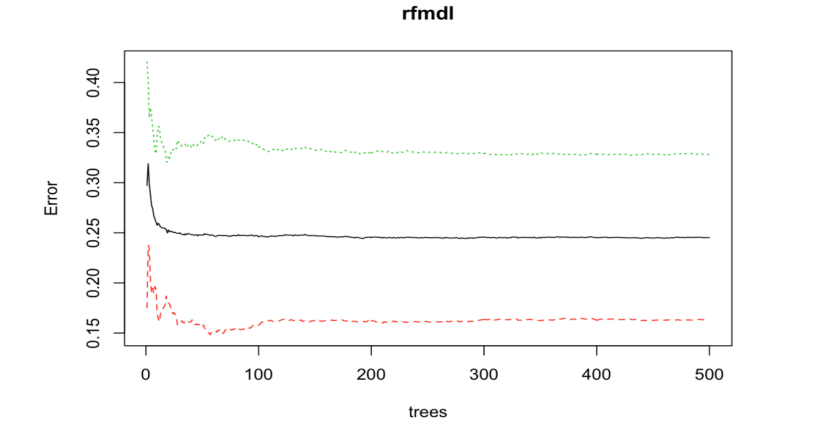
```
set.seed(100)
rfmdl <- randomForest(training_data[,-c(1,2,7,12)], factor(training_data$SeriousDlqin2yrs),
                      sampsize=1000, do.trace=TRUE, importance=TRUE, ntree=500, forest=TRUE)
```

TEST

```
rfmdlpredict <- data.frame(SeriousDlqin2yrs=predict(rfmdl,testing_data[,-c(1,2,7,12)],type="prob")[,2])
rfmdlpredict <- ifelse(rfmdlpredict > 0.5,1,0)
```

MODEL

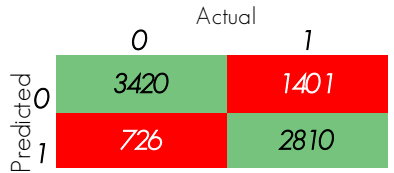
```
plot(rfmdl)
```



RESULT

```
confusionMatrix(rfmdlpredict, testing_SeriousDlqin2yrs)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3420 1401
##          1  726 2810
##
##                Accuracy : 0.7455
##                  95% CI : (0.736, 0.7548)
##     No Information Rate : 0.5039
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4916
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8249
##             Specificity : 0.6673
##          Pos Pred Value : 0.7094
##          Neg Pred Value : 0.7947
##              Prevalence : 0.4961
##          Detection Rate : 0.4092
##    Detection Prevalence : 0.5769
##       Balanced Accuracy : 0.7461
##
##        'Positive' Class : 0
##
```



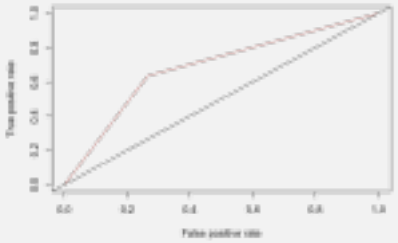
Result Analysis

Confusion Matrix , Roc Curve & Accuracy

Results

Predicted \ Actual	0	1
0	3182	1125
1	993	2657

GLM

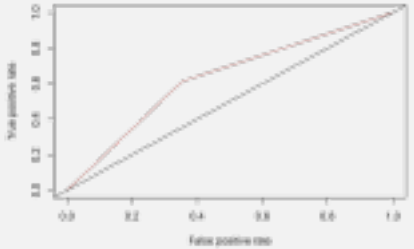


AUC: 0.69

VS

Predicted \ Actual	0	1
0	2640	1577
1	1535	2605

LDA

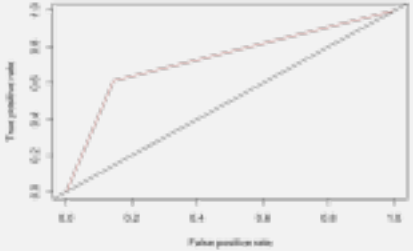


AUC: 0.63

VS

Predicted \ Actual	0	1
0	3572	1570
1	603	2612

SVM

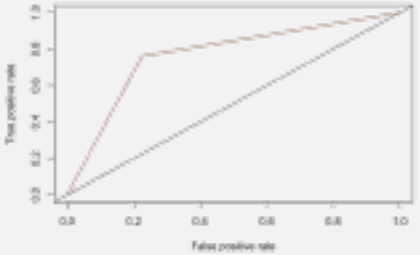


AUC: 0.73

VS

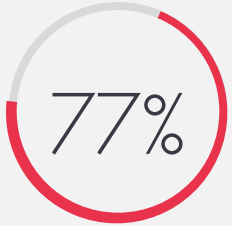
Predicted \ Actual	0	1
0	3234	973
1	941	3209

ANN



AUC: 0.77

Accuracy



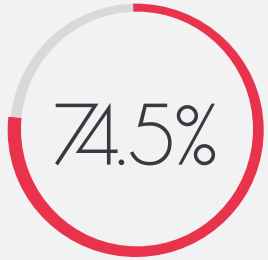
Model Accuracy
to Beat

Logistic
Regression

Linear Discriminant
Analysis

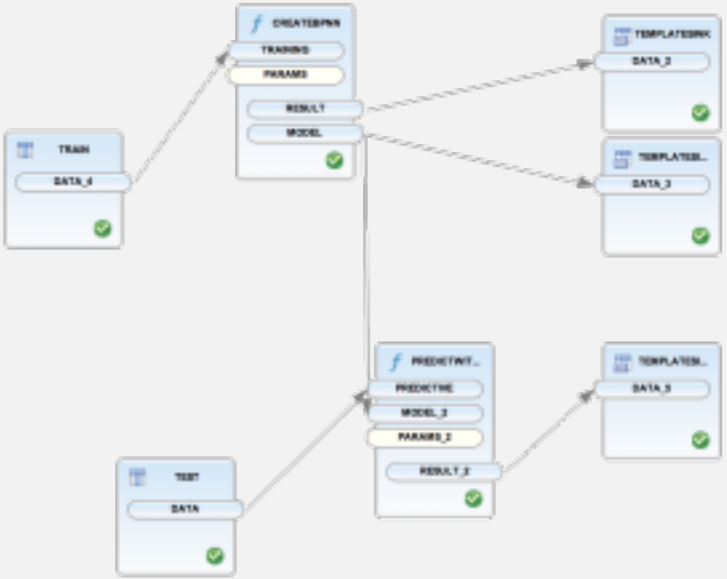
Support Vector
Machine

Artificial Neural
Network



Random
Forest

Performance (SAP HANA)



- R (Time: 44.48 Seconds)

```
> ptm <- proc.time()
>
>
> nnmdl <- neuralnet(SeriousDlqin2yrs ~ RevolvingUtilizationOfUnsecuredLines  + age + NumberOfTime30.59DaysPastDueNotWorse
+                      + DebtRatio  + MonthlyIncome +NumberOfOpenCreditLinesAndLoans + NumberOfTimes90DaysLate +
+                        NumberRealEstateLoansOrLines +  NumberOfTime60.89DaysPastDueNotWorse
+                      + NumberOfDependents, data = training_data, hidden=2, threshold=0.01)
>
> proc.time() - ptm
   user  system elapsed
 37.603   9.114  44.480
```

- SAP HANA (Processing Time: 6.05 Seconds)

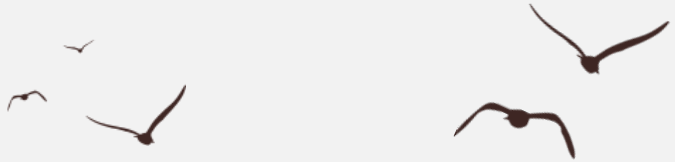
```
Statement 'CALL "PAL_DEV"."GMSC::bpnn"()'
successfully executed in 6.052 seconds  (server processing time: 5.798 seconds) - Rows Affected: 8359
```

Challenges

- Pre-processing and wrangling the data.
- Handling the Skew of the data
- Choosing the right parameters for the algorithms

Conclusion

- Artificial Neural Network performed the best for our particular use case.
- We were able to improve fractionally(.9%) the predictive accuracy of the credit scoring sample model.



Thanks and Questions