LoanDefaultRMarkDownReport

Arun

May 22, 2018

R Markdown

This is an R Markdown document for loan default problem. The objective of this document is to describe the steps followed to analyze and build model for loan prediction dataset.

Required libraries

```r
library(h2o)
```

```
##
## ----------------------------------------------------------------------
##
## Your next step is to start H2O:
##     > h2o.init()
##
## For H2O package documentation, ask for help:
##     > ??h2o
##
## After starting H2O, you can use the Web UI at http://localhost:54321
## For more information visit http://docs.h2o.ai
##
## ----------------------------------------------------------------------
```

```
##
## Attaching package: 'h2o'
```

```
## The following objects are masked from 'package:stats':
##
##     cor, sd, var
```

```
## The following objects are masked from 'package:base':
##
##     %*%, %in%, &&, ||, apply, as.factor, as.numeric, colnames,
##     colnames<-, ifelse, is.character, is.factor, is.numeric, log,
##     log10, log1p, log2, round, signif, trunc
```

```r
library(readr)
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:h2o':
##
##     hour, month, week, year
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(caTools)
library(recommenderlab)
```

```
## Loading required package: Matrix

## Loading required package: arules

##
## Attaching package: 'arules'

## The following object is masked from 'package:dplyr':
##
##     recode

## The following objects are masked from 'package:base':
##
##     abbreviate, write

## Loading required package: proxy

##
## Attaching package: 'proxy'

## The following object is masked from 'package:Matrix':
##
##     as.matrix

## The following objects are masked from 'package:stats':
##
##     as.dist, dist

## The following object is masked from 'package:base':
##
##     as.matrix

## Loading required package: registry
```

```r
library(ggplot2)
```

Set Working Directory and load the data set

The below code segement load the train and test data set.

```r
setwd("C:\\Users\\arun_manu\\Documents\\CognizantLearning\\DSLA\\R\\Loan-Default")
loans.train <- fread("Loan Prediction train.csv")
loans.test <- fread("Loan Prediction test.csv")
```

Create Train and Test Sample

Create a sample size from the train data set (75% Train, 25% Test)

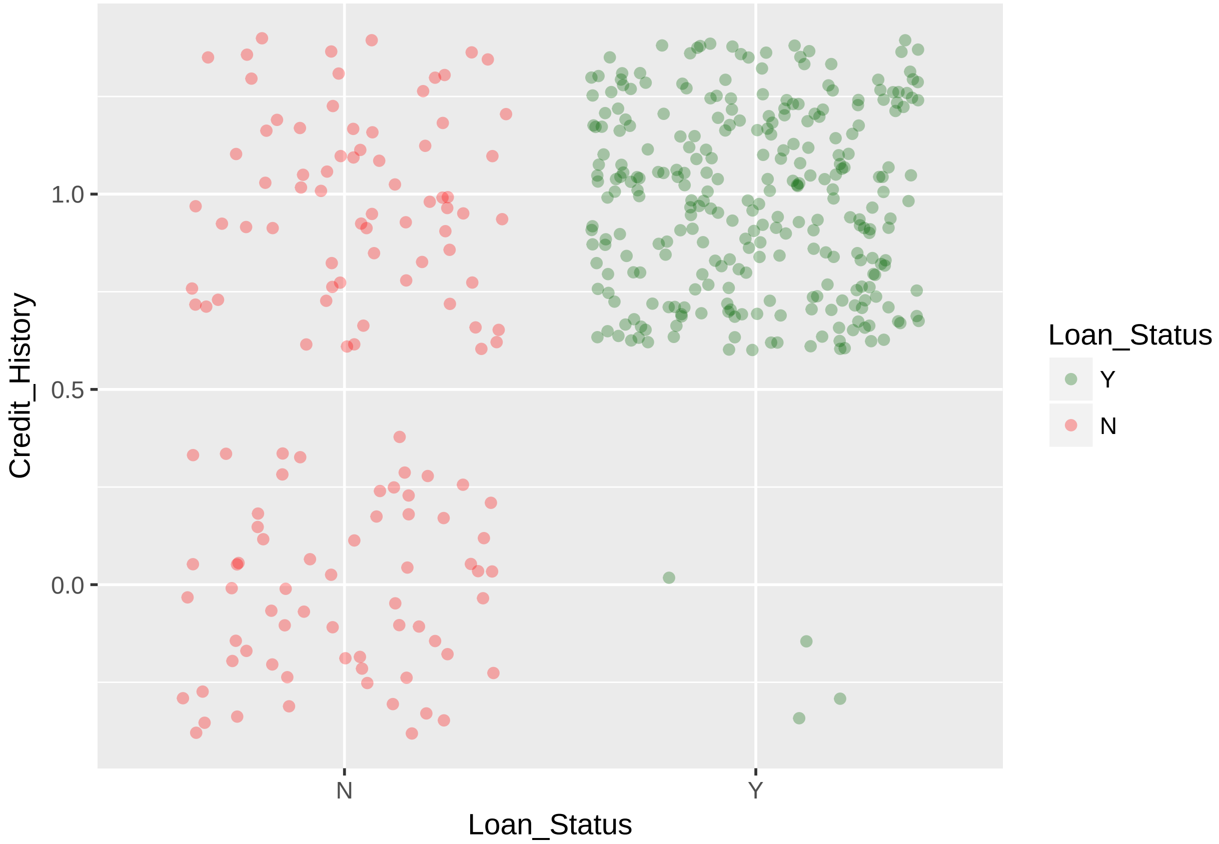
```
set.seed(123)
smp_size <- floor(0.75 * nrow(loans.train))
train_ind <- sample(seq_len(nrow(loans.train)), size = smp_size)
train <- loans.train[train_ind, ]
test <- loans.train[-train_ind, ]
```

Plot the Loan Status Vs Credit History

Application with available credit history have higher chances of getting credit approval

```
plotdata = train
p = ggplot(plotdata,aes(x=Loan_Status,  y=Credit_History, color=Loan_Status))
p + geom_jitter(alpha=0.3) +  scale_color_manual(breaks = c('Y','N'),   values=c('red','darkgreen'))
```

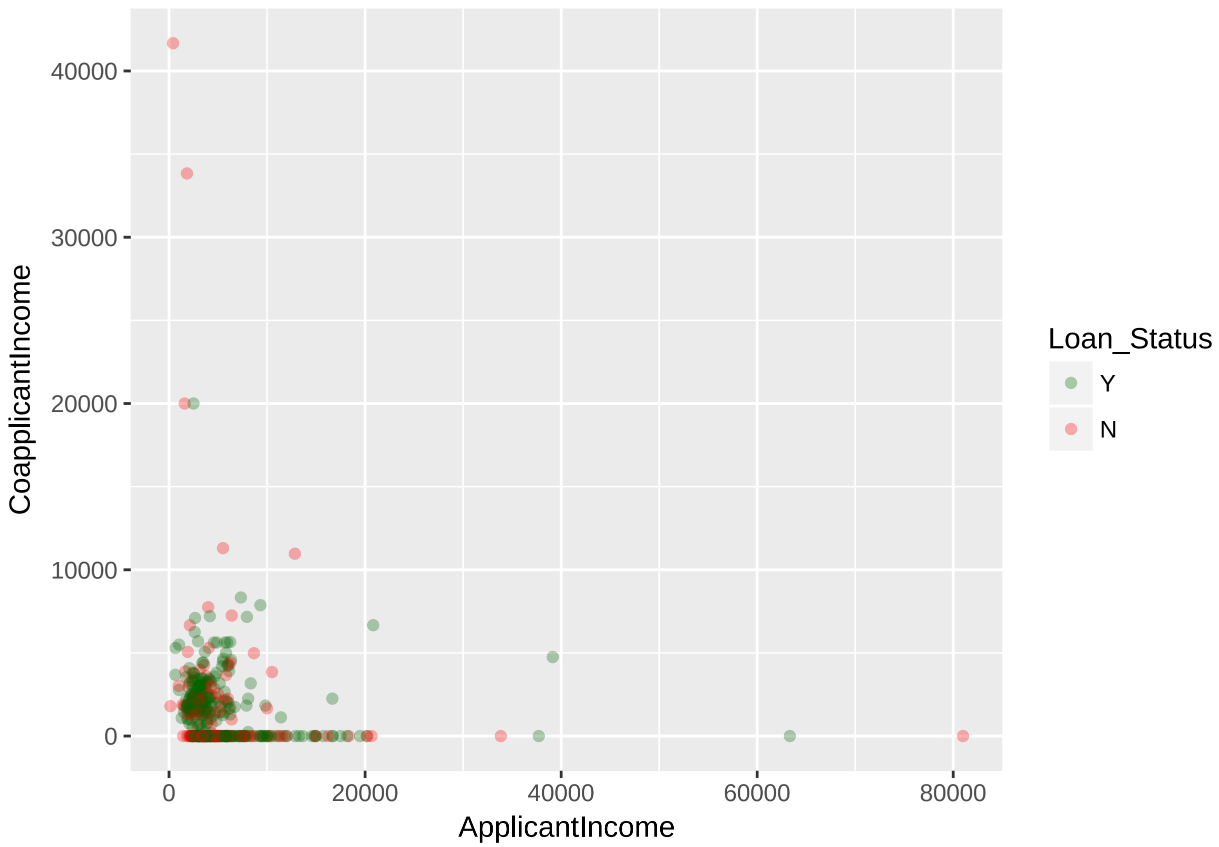
```
## Warning: Removed 42 rows containing missing values (geom_point).
```



Plot the Loan Status Vs ApplicantIncome & Co ApplicantIncome

Application with available credit history have higher chances of getting credit approval

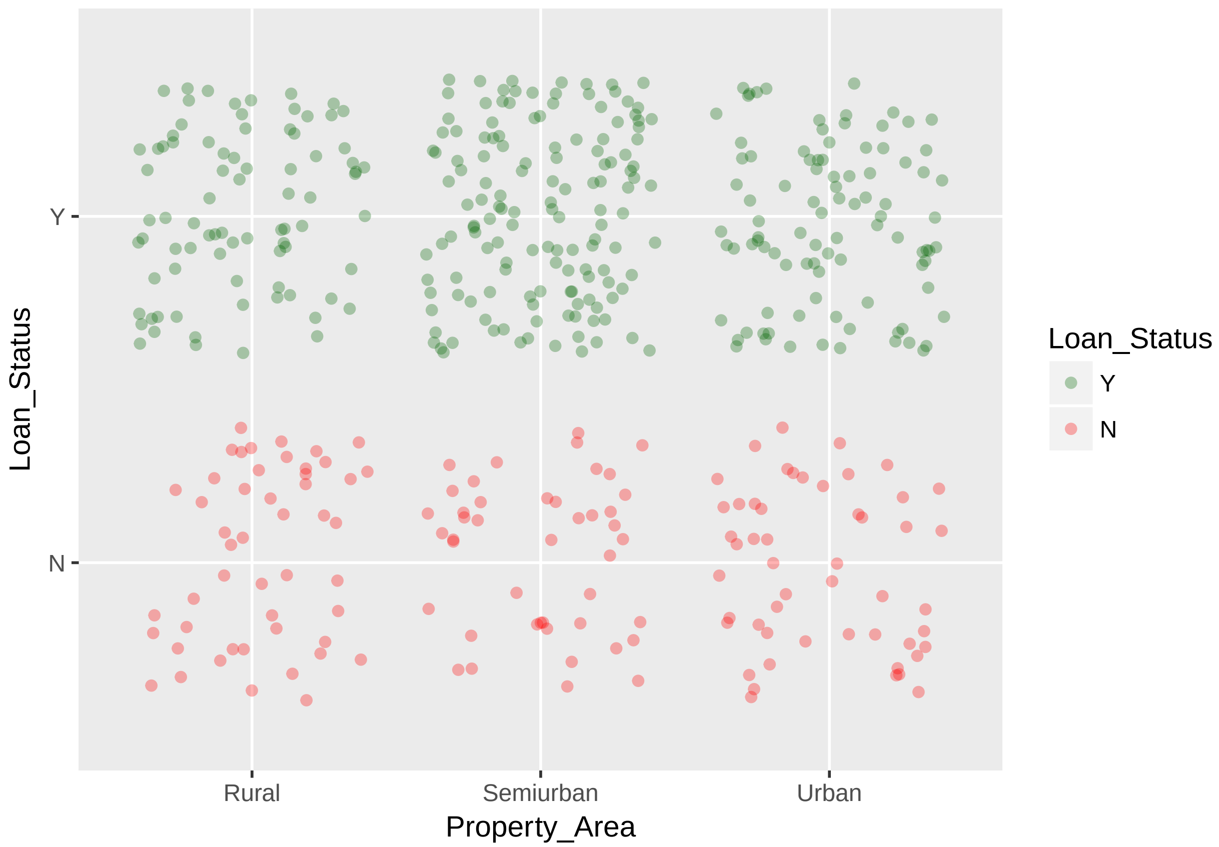
```
p = ggplot(plotdata,aes(x=ApplicantIncome,  y=CoapplicantIncome, color=Loan_Status))
p + geom_jitter(alpha=0.3) +  scale_color_manual(breaks = c('Y','N'),  values=c('red','darkgreen'))
```



Plot the Loan Status Vs Property Area

Application with available credit history have higher chances of getting credit approval

```
p = ggplot(plotdata,aes(x=Property_Area,  y=Loan_Status, color = Loan_Status))
p + geom_jitter(alpha=0.3) +  scale_color_manual(breaks = c('Y','N'), values=c('red','darkgreen'))
```



Remove Loan ID

Loan ID is unique field record and could not contribute in predicting the loan default probability

```
train <- subset( train, select = -c( Loan_ID ))
test <- subset( test, select = -c( Loan_ID ))
```

Exclude NA Data Sets

The below Code segement removes the NA Data sets. Other NA record handling methods na.continue na.fail na.omit

```
train <- na.exclude(train)
test <- na.exclude(test)
list( dimension = dim(train), head = train )
```

```
## $dimension
## [1] 392  12
##
## $head
##      Gender Married Dependents    Education Self_Employed ApplicantIncome
##    1:   Male     Yes          2     Graduate            No            2045
##    2:   Male     Yes          0     Graduate            No           10833
##    3:   Male     Yes          0 Not Graduate            No            1668
##    4:   Male     Yes         3+     Graduate            No            6417
##   5:   Male     Yes          2 Not Graduate            No            6125
##  ---
## 388:   Male      No          0     Graduate            No            5417
## 389:   Male     Yes          0     Graduate            No            2785
## 390:   Male     Yes          2 Not Graduate            No            3083
## 391: Female      No          0 Not Graduate            No            3400
## 392: Female      No          1     Graduate            No            3812
##      CoapplicantIncome LoanAmount Loan_Amount_Term Credit_History
##   1:              1619        101              360              1
##   2:                 0        234              360              1
##   3:              3890        201              360              0
##   4:                 0        157              180              1
##   5:              1625        187              480              1
##  ---
## 388:                 0        168              360              1
## 389:              2016        110              360              1
## 390:              2168        126              360              1
## 391:                 0         95              360              1
## 392:                 0        112              360              1
##      Property_Area Loan_Status
##   1:         Rural           Y
##   2:     Semiurban           Y
##   3:     Semiurban           N
##   4:         Rural           Y
##   5:     Semiurban           N
##  ---
## 388:         Urban           Y
## 389:         Rural           Y
## 390:         Urban           Y
## 391:         Rural           N
## 392:         Rural           Y
```

Create Factor for Categorial variables

As.factor -> Create factor variable.

```r
train$Self_Employed <- as.factor(train$Self_Employed)
train$Property_Area <- as.factor(train$Property_Area)
train$Gender <- as.factor(train$Gender)
train$Dependents <- as.factor(train$Dependents)
train$Married <- as.factor(train$Married)
train$Education <- as.factor(train$Education)
train$Loan_Status <- as.factor(train$Loan_Status)
```

Start the h2o

h2o : Deep Learning library h2o.init-> Initialize the h2o java instance IP -> IP Address of the Host port -> port number Xmx -> Max Heap Memory

```r
h2o.init(nthreads=-1)
```

```
##
## H2O is not running yet, starting it now...
```

```
##
## Note:  In case of errors look at the following log files:
##     C:\Users\ARUN_M~1\AppData\Local\Temp\RtmpqiAIaX/h2o_arun_manu_started_from_r.out
##     C:\Users\ARUN_M~1\AppData\Local\Temp\RtmpqiAIaX/h2o_arun_manu_started_from_r.err
##
##
## Starting H2O JVM and connecting: ... Connection successful!
##
## R is connected to the H2O cluster:
##     H2O cluster uptime:         7 seconds 189 milliseconds
##     H2O cluster timezone:       America/New_York
##     H2O data parsing timezone:  UTC
##     H2O cluster version:        3.18.0.8
##     H2O cluster version age:    1 month and 4 days
##     H2O cluster name:           H2O_started_from_R_arun_manu_wbv567
##     H2O cluster total nodes:    1
##     H2O cluster total memory:   1.76 GB
##     H2O cluster total cores:    4
##     H2O cluster allowed cores:  4
##     H2O cluster healthy:        TRUE
##     H2O Connection ip:          localhost
##     H2O Connection port:        54321
##     H2O Connection proxy:       NA
##     H2O Internal Security:      FALSE
##     H2O API Extensions:         Algos, AutoML, Core V3, Core V4
##     R Version:                  R version 3.5.0 (2018-04-23)
```

Create Data Frame

Create data frame to get converted to h2o data table frame as.data.table -> Convert to table data frame as.h2o -> Convert to h2o data frame

```
train <- as.data.table(train)
dat_h2o <- as.h2o(train)
```

```
##
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```

```
head(train)
```

```
##    Gender Married Dependents    Education Self_Employed ApplicantIncome
## 1:   Male     Yes          2     Graduate            No            2045
## 2:   Male     Yes          0     Graduate            No           10833
## 3:   Male     Yes          0 Not Graduate            No            1668
## 4:   Male     Yes         3+     Graduate            No            6417
## 5:   Male     Yes          2 Not Graduate            No            6125
## 6:   Male     Yes          2 Not Graduate            No            4226
##    CoapplicantIncome LoanAmount Loan_Amount_Term Credit_History
## 1:              1619        101              360              1
## 2:                 0        234              360              1
## 3:              3890        201              360              0
## 4:                 0        157              180              1
## 5:              1625        187              480               1
## 6:              1040        110              360               1
##    Property_Area Loan_Status
## 1:         Rural           Y
## 2:     Semiurban           Y
## 3:     Semiurban           N
## 4:         Rural           Y
## 5:     Semiurban           N
## 6:         Urban           Y
```

Create the model using h2o deep learning library

Brief overview of the parameters used:

X and Y: List of the predictors and target variable respectively

training_frame: H2O training frame data

activation: Indicates which activation function to use

hidden: Number of hidden layers and their size

l1: L1 regularization

train_samples_per_iteration: Number of training samples per iteration

classification_stop: Stopping criterion for classification error

epochs: How many times the dataset should be iterated

overwrite_with_best_model: If TRUE, overrides the final model with the best model

standardize: If TRUE, auto standardize the data

distribution: The distribution function of the response. It can be AUTO

missing_values_handling: Ways to handle missing values

stopping_metric: The stopping metric criterion

nfold: Specifying the number of folds for N Fold cross validation

```
model <- h2o.deeplearning(x = 1:11,
                          y = 12,
                          training_frame = dat_h2o,
                          activation = "RectifierWithDropout",
                          hidden = c(500,1000),
                          input_dropout_ratio = 0.2,
                          l1 = 1.0e-5,
                          train_samples_per_iteration = -1,
                          classification_stop = -1,
                          epochs = 100,
                          overwrite_with_best_model = TRUE,
                          standardize = TRUE,
                          distribution = "AUTO",
                          #c("AUTO", "gaussian", "bernaulli",
                          #  "multinomial", "poisson", "quantile"),
                          missing_values_handling = "MeanImputation",
                          #c("MeanImputation", "Skip"),
                          stopping_metric = "AUTO",
                          # c("AUTO", "logloss", "MSE"),
                          nfolds = 5
                          )
```

```
##
  |
  |                                                                 |   0%
  |
  |==                                                               |   3%
  |
  |======                                                           |   9%
  |
  |=============                                                    |  20%
  |
  |===============                                                  |  23%
  |
  |=====================                                            |  32%
  |
  |===============================                                  |  47%
  |
  |========================================                         |  62%
  |
  |=================================================                |  76%
  |
  |====================================================             |  81%
  |
  |======================================================           |  84%
  |
  |=======================================================          |  84%
  |
  |=======================================================          |  85%
  |
  |========================================================         |  86%
  |
  |=========================================================        |  88%
  |
  |==========================================================       |  89%
  |
  |===========================================================      |  91%
  |
  |=============================================================    |  94%
  |
  |===============================================================  |  97%
  |
  |=================================================================| 100%
```

Create Confusion Matrix

Create Confusion Matrix Based on test data frame

```
test <- as.data.table(test)
dat_h2o_test <- as.h2o(test)
```

```
##
```

```
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```

```
h2o.confusionMatrix(model,dat_h2o_test)
```

```
## Confusion Matrix (vertical: actual; across: predicted)  for max f1 @ threshold = 0.247155486854121:
##          N   Y    Error     Rate
## N       15  25 0.625000   =25/40
## Y        2  95 0.020619    =2/97
## Totals 17 120 0.197080  =27/137
```

h2o.varimp : obtaining the variable importance

```
predset <- as.data.table(loans.test)
dat_h2o_pred <- as.h2o(predset)
```

```
##
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```

```
head( as.data.table( h2o.varimp(model)))
```

```
##                   variable relative_importance scaled_importance
## 1:          Credit_History           1.0000000         1.0000000
## 2:       CoapplicantIncome           0.6738093         0.6738093
## 3: Property_Area.Semiurban           0.6594438         0.6594438
## 4:     Property_Area.Rural           0.6045292         0.6045292
## 5:        Loan_Amount_Term           0.5976136         0.5976136
## 6:            Dependents.1           0.5972681         0.5972681
##    percentage
## 1: 0.07105873
## 2: 0.04788003
## 3: 0.04685924
## 4: 0.04295707
## 5: 0.04246566
## 6: 0.04244111
```

```
pred <- h2o.predict(model, dat_h2o_pred)
```

```
##
  |
  |                                                                 |   0%
  |
  |=================================================================| 100%
```