Bank Marketing

RohitDixit

Problem Statement

Company wants to automate the loan eligibility process (real time) based on customer detail provided while filling online application form. These details are Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History and others. To automate this process, they have given a problem to identify the customers segments, those are eligible for loan amount so that they can specifically target these customers. Here they have provided a partial data set.

R Code

Importing Library

```r
library(caret)
library(e1071)
library(caTools)
library(rpart)
library(rpart.plot)
library(randomForest)
library(ranger)
library(ggplot2)
library(ggthemes)
```

Data Importing

Data available at https://archive.ics.uci.edu/ml/datasets/bank+marketing

```r
Data <- read.csv("bank-full.csv", header = TRUE, sep = ";")
```

Data Exploration

```r
#Exploring Data
summary(Data)
```

```
##       age                  job           marital          education
##  Min.   :18.00   blue-collar:9732   divorced: 5207   primary  : 6851
##  1st Qu.:33.00   management :9458   married :27214   secondary:23202
##  Median :39.00   technician :7597   single  :12790   tertiary :13301
##  Mean   :40.94   admin.     :5171                    unknown  : 1857
##  3rd Qu.:48.00   services   :4154
##  Max.   :95.00   retired    :2264
##                  (Other)    :6835
##  default        balance        housing      loan           contact
##  no :44396   Min.   : -8019   no :20081   no :37967   cellular :29285
##  yes:  815   1st Qu.:    72   yes:25130   yes: 7244   telephone: 2906
```

```
##              Median :   448                           unknown  :13020
##              Mean   :  1362
##              3rd Qu.:  1428
##              Max.   :102127
##
##       day              month          duration          campaign
##  Min.   : 1.00   may    :13766   Min.   :   0.0   Min.   : 1.000
##  1st Qu.: 8.00   jul    : 6895   1st Qu.: 103.0   1st Qu.: 1.000
##  Median :16.00   aug    : 6247   Median : 180.0   Median : 2.000
##  Mean   :15.81   jun    : 5341   Mean   : 258.2   Mean   : 2.764
##  3rd Qu.:21.00   nov    : 3970   3rd Qu.: 319.0   3rd Qu.: 3.000
##  Max.   :31.00   apr    : 2932   Max.   :4918.0   Max.   :63.000
##                  (Other): 6060
##      pdays          previous           poutcome       y
##  Min.   : -1.0   Min.   :  0.0000   failure: 4901   no :39922
##  1st Qu.: -1.0   1st Qu.:  0.0000   other  : 1840   yes: 5289
##  Median : -1.0   Median :  0.0000   success: 1511
##  Mean   : 40.2   Mean   :  0.5803   unknown:36959
##  3rd Qu.: -1.0   3rd Qu.:  0.0000
##  Max.   :871.0   Max.   :275.0000
##
```

```r
#Check for NA is any
which(is.na(Data), arr.ind = TRUE)
```

```
##      row col
```

```r
# No NA present

#check for types of columns
str(Data)
```

```
## 'data.frame':    45211 obs. of  17 variables:
##  $ age      : int  58 44 33 47 33 35 28 42 58 43 ...
##  $ job      : Factor w/ 12 levels "admin.","blue-collar",..: 5 10 3 2 12 5 5 3 6 10 ...
##  $ marital  : Factor w/ 3 levels "divorced","married",..: 2 3 2 2 3 2 3 1 2 3 ...
##  $ education: Factor w/ 4 levels "primary","secondary",..: 3 2 2 4 4 3 3 3 1 2 ...
##  $ default  : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 2 1 1 ...
##  $ balance  : int  2143 29 2 1506 1 231 447 2 121 593 ...
##  $ housing  : Factor w/ 2 levels "no","yes": 2 2 2 2 1 2 2 2 2 2 ...
##  $ loan     : Factor w/ 2 levels "no","yes": 1 1 2 1 1 1 2 1 1 1 ...
##  $ contact  : Factor w/ 3 levels "cellular","telephone",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ day      : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ month    : Factor w/ 12 levels "apr","aug","dec",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ duration : int  261 151 76 92 198 139 217 380 50 55 ...
##  $ campaign : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ pdays    : int  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ previous : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ poutcome : Factor w/ 4 levels "failure","other",..: 4 4 4 4 4 4 4 4 4 4 ...
```

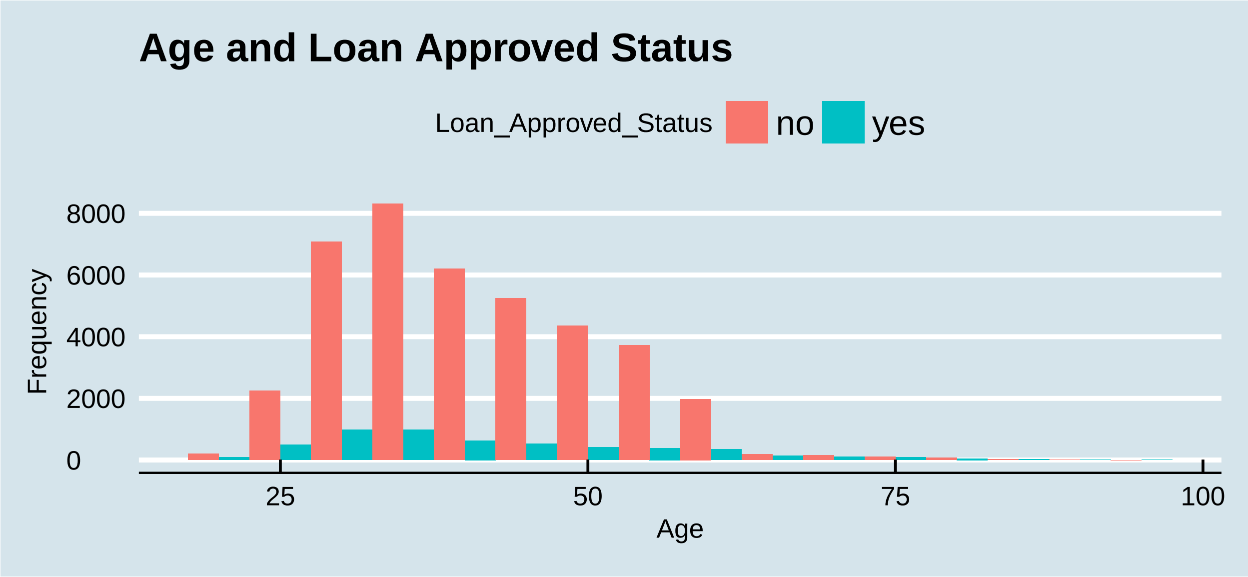
```
##  $ y         : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

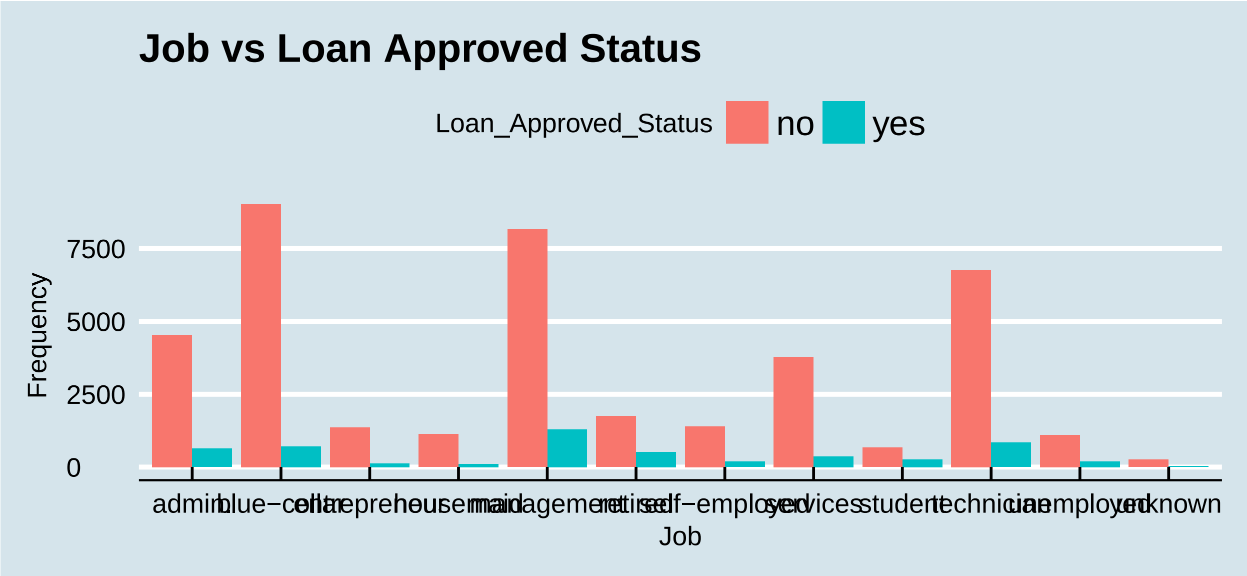
Checking Correlation in numeric variables

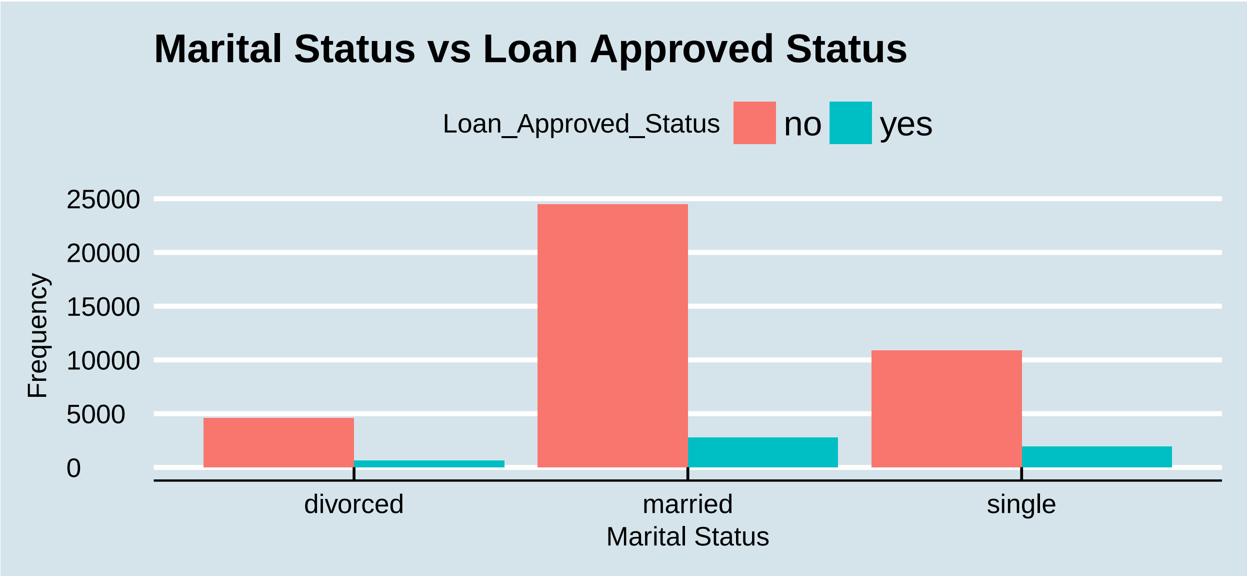
```r
cor(Data[,c(1,6,10,12,13,14,15)])
```

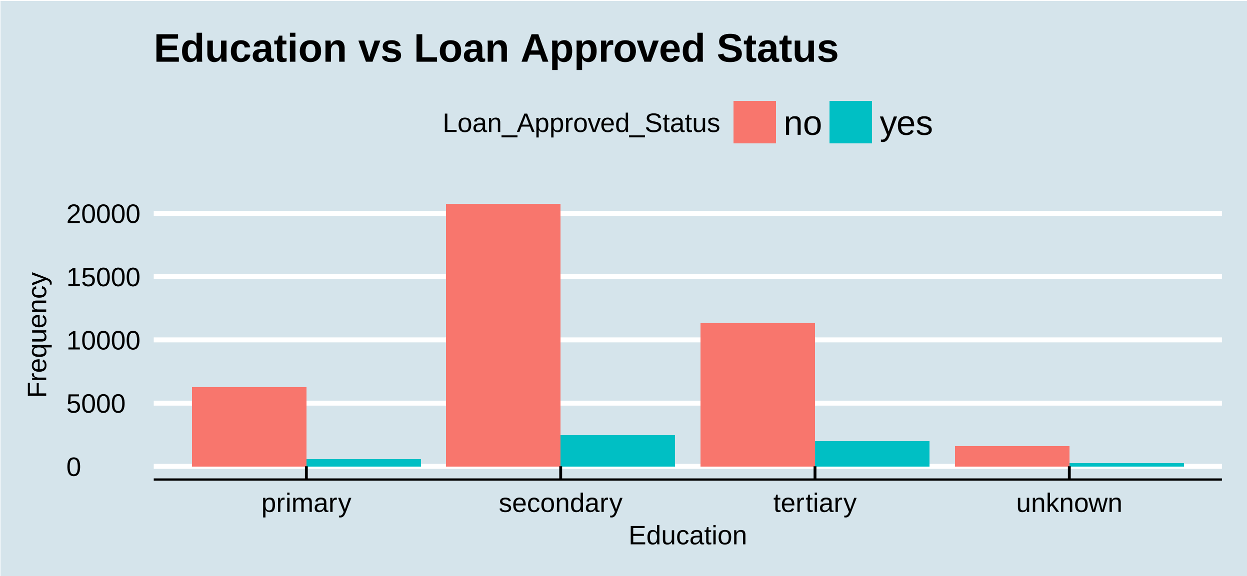
```
##                   age      balance          day     duration     campaign
## age       1.000000000  0.097782739 -0.009120046 -0.004648428  0.004760312
## balance   0.097782739  1.000000000  0.004502585  0.021560380 -0.014578279
## day      -0.009120046  0.004502585  1.000000000 -0.030206341  0.162490216
## duration -0.004648428  0.021560380 -0.030206341  1.000000000 -0.084569503
## campaign  0.004760312 -0.014578279  0.162490216 -0.084569503  1.000000000
## pdays    -0.023758014  0.003435322 -0.093044074 -0.001564770 -0.088627668
## previous  0.001288319  0.016673637 -0.051710497  0.001203057 -0.032855290
##                 pdays     previous
## age      -0.023758014  0.001288319
## balance   0.003435322  0.016673637
## day      -0.093044074 -0.051710497
## duration -0.001564770  0.001203057
## campaign -0.088627668 -0.032855290
## pdays     1.000000000  0.454819635
## previous  0.454819635  1.000000000
```

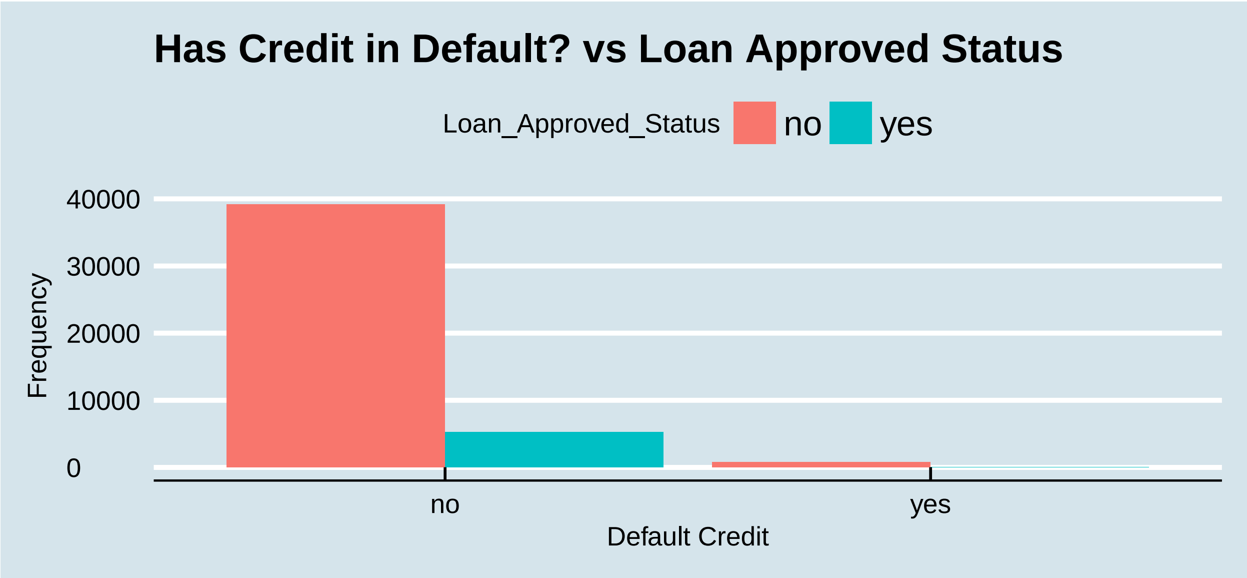
Lets visualize the variables

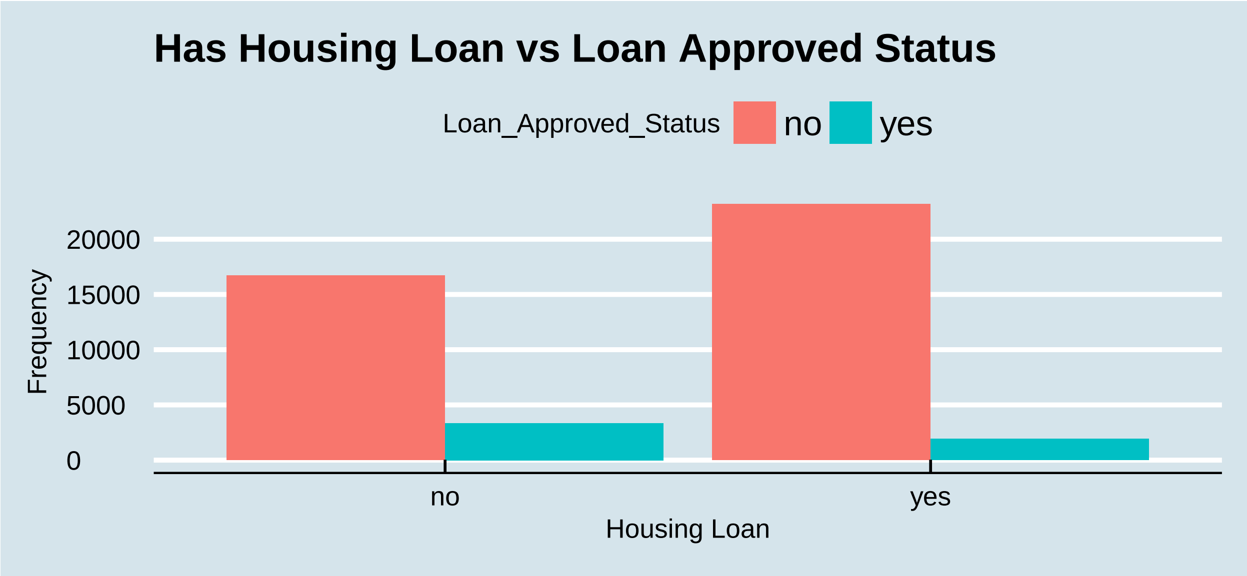


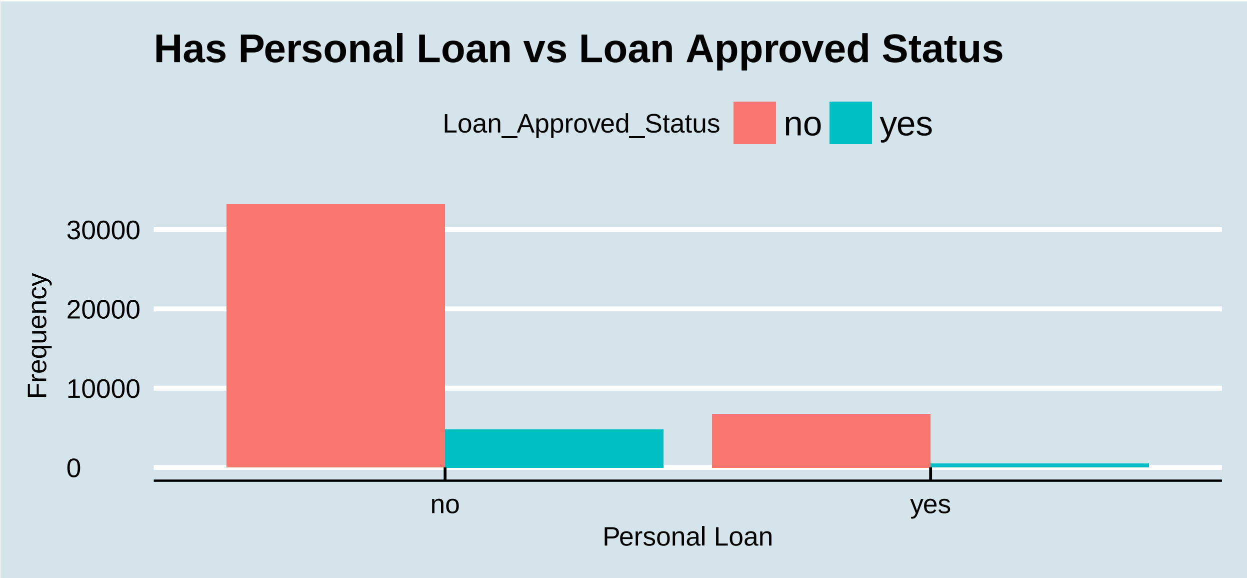

Job vs Loan Approved Status
Loan_Approved_Status
no
yes
Frequency
7500
5000
2500
0
admin.
blue−collar
entrepreneur
housemaid
management
retired
self−employed
services
student
technician
unemployed
unknown
Job

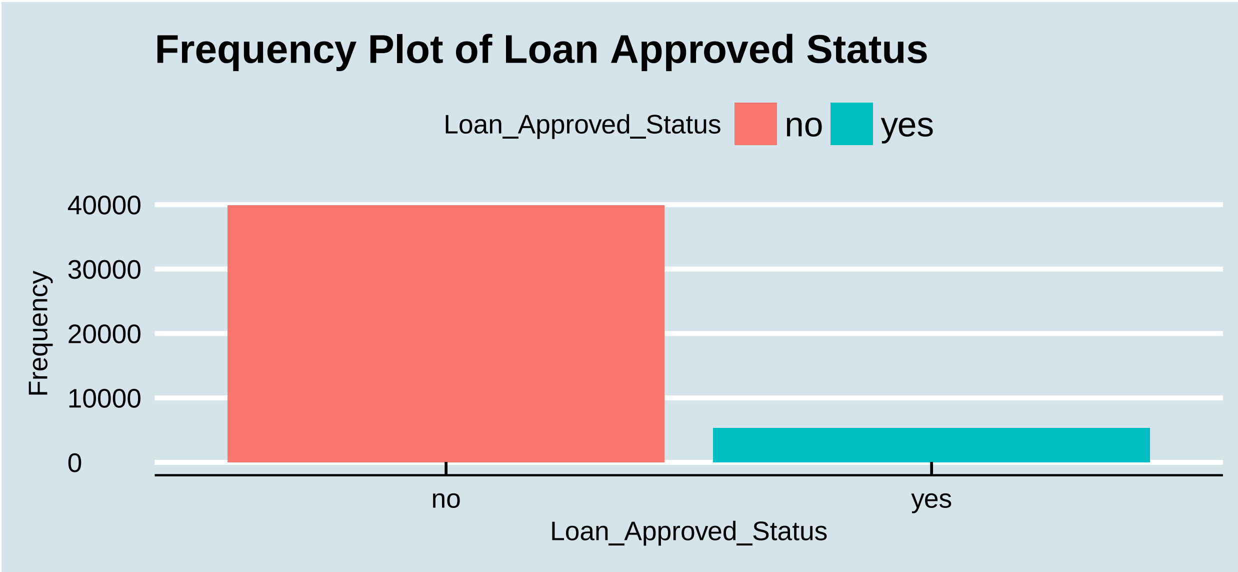

Marital Status vs Loan Approved Status
Loan_Approved_Status
no
yes
Frequency
25000
20000
15000
10000
5000
0
divorced
married
single
Marital Status


Education vs Loan Approved Status
Loan_Approved_Status
no
yes
Frequency
20000
15000
10000
5000
0
primary
secondary
tertiary
unknown
Education


Has Credit in Default? vs Loan Approved Status
Loan_Approved_Status
no
yes
Frequency
40000
30000
20000
10000
0
no
yes
Default Credit


Has Housing Loan vs Loan Approved Status
Loan_Approved_Status
no
yes
20000
15000
10000
5000
0
Frequency
no
yes
Housing Loan


Has Personal Loan vs Loan Approved Status
Loan_Approved_Status
no
yes
30000
20000
10000
0
Frequency
no
yes
Personal Loan

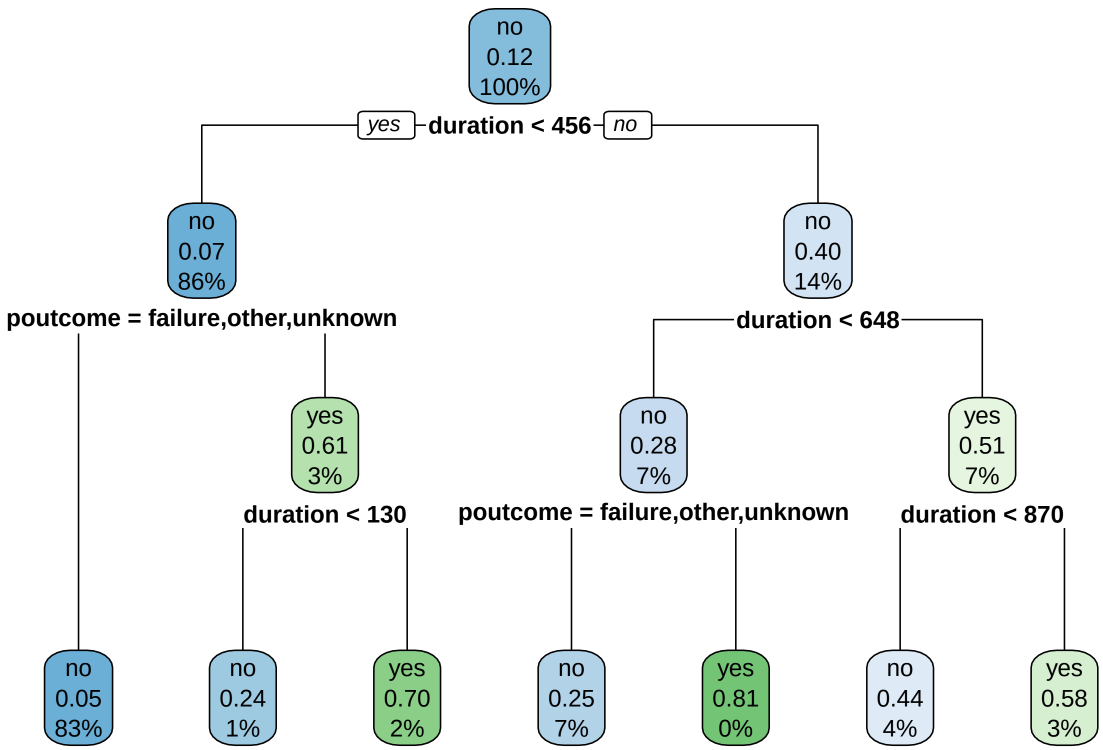


** Filling NA values**
##Splitting Data into Train & Test

```
set.seed(1)
sample = sample.split(Data$age, SplitRatio = .70)
train_data = subset(Data, sample == TRUE)
test_data  = subset(Data, sample == FALSE)
```

Making Decision Tree as first model #89.95% Accuracy

```
model_dtree <- rpart(y~., data=train_data)
rpart.plot(model_dtree)
```




```r
predictions_dtree <- predict(model_dtree, test_data[,-17], type = "class")
confusionMatrix(test_data$y,predictions_dtree)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    no   yes
##        no  11673   288
##        yes  1076   529
##
##                Accuracy : 0.8995
##                  95% CI : (0.8943, 0.9045)
##     No Information Rate : 0.9398
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.388
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9156
##             Specificity : 0.6475
##          Pos Pred Value : 0.9759
##          Neg Pred Value : 0.3296
##              Prevalence : 0.9398
##          Detection Rate : 0.8605
##    Detection Prevalence : 0.8817
##       Balanced Accuracy : 0.7815
##
```

```
##        'Positive' Class : no
##
```

Training a KNN $89.96% Accuracy

```
model_knn<-train(y~.,data=train_data,method='knn')
predictions_knn <- predict(model_knn, test_data[,-17])
confusionMatrix(test_data$y,predictions_knn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    no   yes
##        no  11587   374
##        yes  1222   383
##
##                Accuracy : 0.8824
##                  95% CI : (0.8768, 0.8877)
##     No Information Rate : 0.9442
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2689
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9046
##             Specificity : 0.5059
##          Pos Pred Value : 0.9687
##          Neg Pred Value : 0.2386
##              Prevalence : 0.9442
##          Detection Rate : 0.8541
##    Detection Prevalence : 0.8817
##       Balanced Accuracy : 0.7053
##
##        'Positive' Class : no
##
```

Making RandomForest # 90.64%

```
model_rf<-train(y~.,data=train_data,method='ranger')
predictions_rf <- predict(model_rf, test_data[,-17])
confusionMatrix(test_data$y,predictions_rf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    no   yes
```

```
##         no  11550   411
##         yes   855   750
##
##                Accuracy : 0.9067
##                  95% CI : (0.9017, 0.9115)
##     No Information Rate : 0.9144
##     P-Value [Acc > NIR] : 0.9993
##
##                   Kappa : 0.4918
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9311
##             Specificity : 0.6460
##          Pos Pred Value : 0.9656
##          Neg Pred Value : 0.4673
##              Prevalence : 0.9144
##          Detection Rate : 0.8514
##    Detection Prevalence : 0.8817
##       Balanced Accuracy : 0.7885
##
##        'Positive' Class : no
##
```

Training a SVM with radial kernel 90.19% Accuracy

```
model_svm<-train(y~.,data=train_data,method='svmRadial')
predictions_svm <- predict(model_svm, test_data[,-17])
confusionMatrix(test_data$y,predictions_svm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    no   yes
##         no  11717   244
##         yes  1088   517
##
##                Accuracy : 0.9018
##                  95% CI : (0.8967, 0.9068)
##     No Information Rate : 0.9439
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3906
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9150
##             Specificity : 0.6794
##          Pos Pred Value : 0.9796
```

```
##          Neg Pred Value : 0.3221
##              Prevalence : 0.9439
##          Detection Rate : 0.8637
##    Detection Prevalence : 0.8817
##       Balanced Accuracy : 0.7972
##
##        'Positive' Class : no
##
```

Conclusion

Will use Decision Tree for this dataset as the accuracy is very close to that of SVM or Random Forest, since the time required to train a Decision Tree model is far less than svm or Random Forest model.