# Random Forest

```
## Loading required package: lattice
## Loading required package: ggplot2
```

## Stratified sampling

```
set.seed(123456)
inTrain = createDataPartition(y=data$target, p=0.75, list=FALSE)
training = data[inTrain,]
testing = data[-inTrain,]
dim(training);dim(testing)
```

```
## [1] 358  19
```

```
## [1] 119  19
```

## Random Forest

- can handles the overfitting problem you faced with decision trees. This approach having the majority's vote count as the actual classification decision.
- can deal with "small n large p"-problem, high-order interactions, correlated predictor variables.

```
formular1 =
as.factor(target)~new_outstanding_principal_balance+initial_loan_amount+fico+
sales_channel__c+type+current_collection_method+term+average_bank_balance__c+
lender1

my_forest <- train(formular1, method="rf", data=training,
trControl=trainControl(method="cv"),number=3)
```

```
## Loading required package: randomForest
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
print(my_forest)
```

```
## Random Forest
##
## 358 samples
##  18 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
##
## Summary of sample sizes: 322, 322, 322, 322, 322, 323, ...
##
```

```
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.7151587  0.2768319  0.04241798   0.1280355
##    7    0.6677778  0.2236602  0.08039734   0.1616732
##   12    0.6594444  0.2058128  0.07842160   0.1513176
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
my_forest$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry, number = 3)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 29.33%
## Confusion matrix:
##    0   1 class.error
## 0 36  93  0.72093023
## 1 12 217  0.05240175
```

## Apply Random Forest to the testing data

```
forest_pred <- predict(my_forest, testing)
forest_pred_df <- data.frame(forest_pred, target=testing$target)

confusionForest=table(forest_pred_df$forest_pred, forest_pred_df$target)
print(confusionForest)
```

```
##
##      0  1
##   0 13  6
##   1 21 79
```

```
Observed_Accuracy =
(confusionForest[1,1]+confusionForest[2,2])/sum(confusionForest)
print(Observed_Accuracy)
```

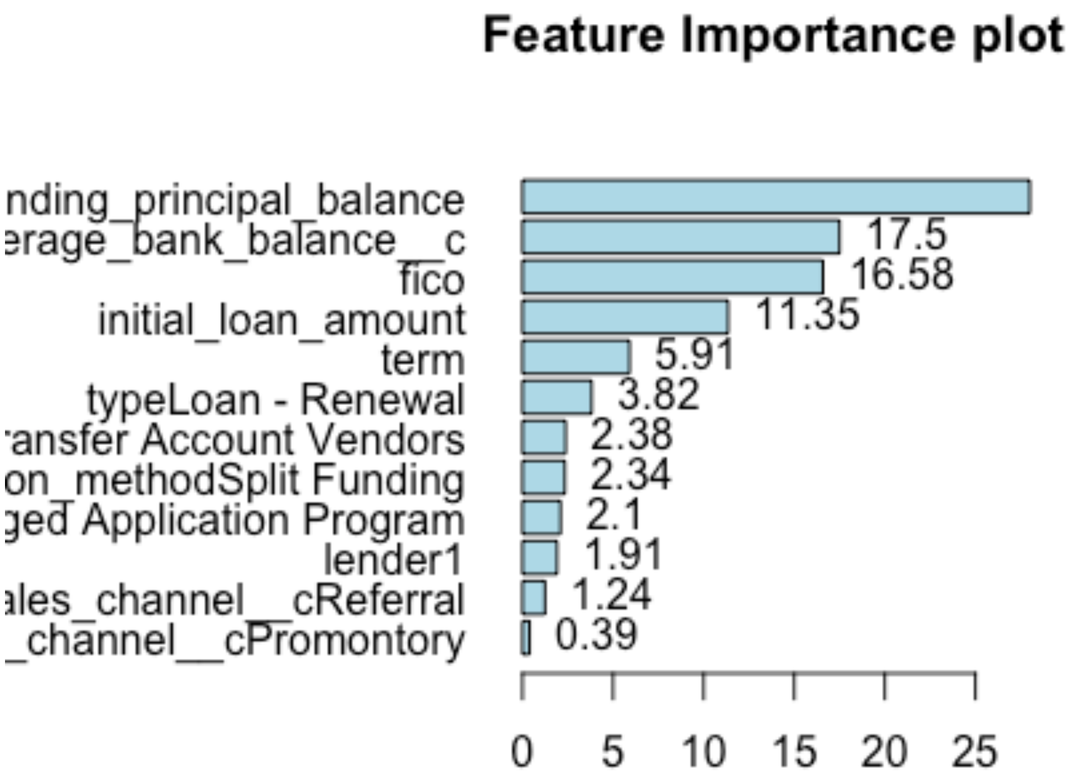
```
## [1] 0.7731092
```

```
Expected_Accuracy =
(sum(confusionForest[,1])*sum(confusionForest[1,])/sum(confusionForest)+sum(c
onfusionForest[,2])*sum(confusionForest[2,])/sum(confusionForest))/sum(confus
ionForest)
```

```r
kappa = (Observed_Accuracy-Expected_Accuracy)/(1-Expected_Accuracy)
print(kappa)

## [1] 0.359322
```

## Random forest variable importance measures and plots

```r
forestImp <- round(varImp(my_forest$finalModel, scale = FALSE),2)
forestImp1=forestImp[order(forestImp$Overall),,drop=FALSE]
forestImp[order(-forestImp$Overall),,drop=FALSE]

##                                                    Overall
## new_outstanding_principal_balance                    28.00
## average_bank_balance__c                              17.50
## fico                                                 16.58
## initial_loan_amount                                  11.35
## term                                                  5.91
## typeLoan - Renewal                                    3.82
## current_collection_methodTransfer Account Vendors     2.38
## current_collection_methodSplit Funding                2.34
## sales_channel__cFAP: Managed Application Program      2.10
## lender1                                               1.91
## sales_channel__cReferral                              1.24
## sales_channel__cPromontory                            0.39

counts <- forestImp1[,1]
par(mai=c(1,2,1,1))
bp=barplot(counts, main="Feature Importance plot", horiz=TRUE,
names.arg=rownames(forestImp1),col="light blue",las=1)
text(x= counts, bp, labels=as.character(counts), pos=4)
```



Feature Importance plot
nding_principal_balance
erage_bank_balance__c
fico
initial_loan_amount
term
typeLoan - Renewal
ansfer Account Vendors
on_methodSplit Funding
ged Application Program
lender1
ales_channel__cReferral
_channel__cPromontory
17.5
16.58
11.35
5.91
3.82
2.38
2.34
2.1
1.91
1.24
0.39
0
5
10
15
20
25