Report Two

Muiz Murad

28 October 2019

```
library(ROracle)
```

```
## Loading required package: DBI
```

```
drv <- dbDriver("Oracle")

## Refer to Oracle Database Net Services Administator's Guide for
## details on connect string specification.
host <- "oracle.vittl.it.bond.edu.au"
port <- 1521
sid <- "inft320"
connect.string <- paste(
  "(DESCRIPTION=",
  "(ADDRESS=(PROTOCOL=tcp)(HOST=", host, ")(PORT=", port, "))",
  "(CONNECT_DATA=(SID=", sid, ")))", sep = "")

## Use username/password authentication.
con <- dbConnect(drv, username = "A13599863", password = "A13599863",
                 dbname = connect.string)

## Run a SQL statement by creating first a resultSet object.
rs <- dbSendQuery(con, "select * from brucedba.BankMarketing")
```

Executive Summary

The purpose of this report is to show how data points collected by the Protguese Telemarketing Company can be used at the bank to predict customer loan defaults and vital information that can be used to immediately cancel the loan at the time of application, and, hence save the bank from unexpected losses. Providing the telco with such a model to reject potentially bad loans and only approve good loans, can help to increase profit margins.

The method used in this analysis is binary prediction (only having two possible outcomes) using Logistic Regression.In this case, the outcome being a loan to be good. When we consider all loans with a probability greater than 0.5 as good loans and the rest as bad, we get 88.73% of good loans predicted as no and 11.27% of loans predicted as yes. Giving a total accuracy of 77.46%.

There are future improvements that can be made to the model. More complex procedures can be adopted. Quality of data collection can be improved. There are multiple processes that can be taken by the telco to create an environment to retain data and have data motivated decision making.The bank investing in predictive modeling is equivalent to investing in the future. Frequent retraining of the model is a must to keep up to date with changing trends and data.

Introduction

This report will focus on making a model, using a logistic regression to predict whether or not the telemarketing company's customers will default or not on a loan. Regression models were used to analyse the data and develop models that will be used on both train and test data.

```
#We Now Extract DataPoints From The DataSet
#Extract All Rows
loan <- fetch(rs)
```

```
#Clean DataSet And Remove Unecessary Variable(s)
library(dplyr)
clean_loan <- select(loan,-c(DURATION))
```

```
#Convert Factors to Numeric
factornames <- c("AGE","PDAYS","PREVIOUS","EMP_VAR_RATE","CONS_PRICE_IDX","CONS_CONF_IDX","EURIBOR3M","

clean_loan[,factornames] <- lapply(factornames, function (x) as.numeric(as.character(clean_loan[,x])))
```

```
#Convert Factors to Categorical
factornames1 <- c("JOB","MARITAL","EDUCATION","DEFAULTCREDIT","HOUSING","LOAN","CONTACT","MONTH","DAY_O

clean_loan[,factornames1] <- lapply(factornames1, function (x) as.factor(as.character(clean_loan[,x])))

order("Y")
```

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

Split the data into training and test sets to ensure that we are able to have data to test the model with as the training data will also produce high accuracies as it has been used to create the model already. The test data acts as forgein data on the model to produce accuracies.

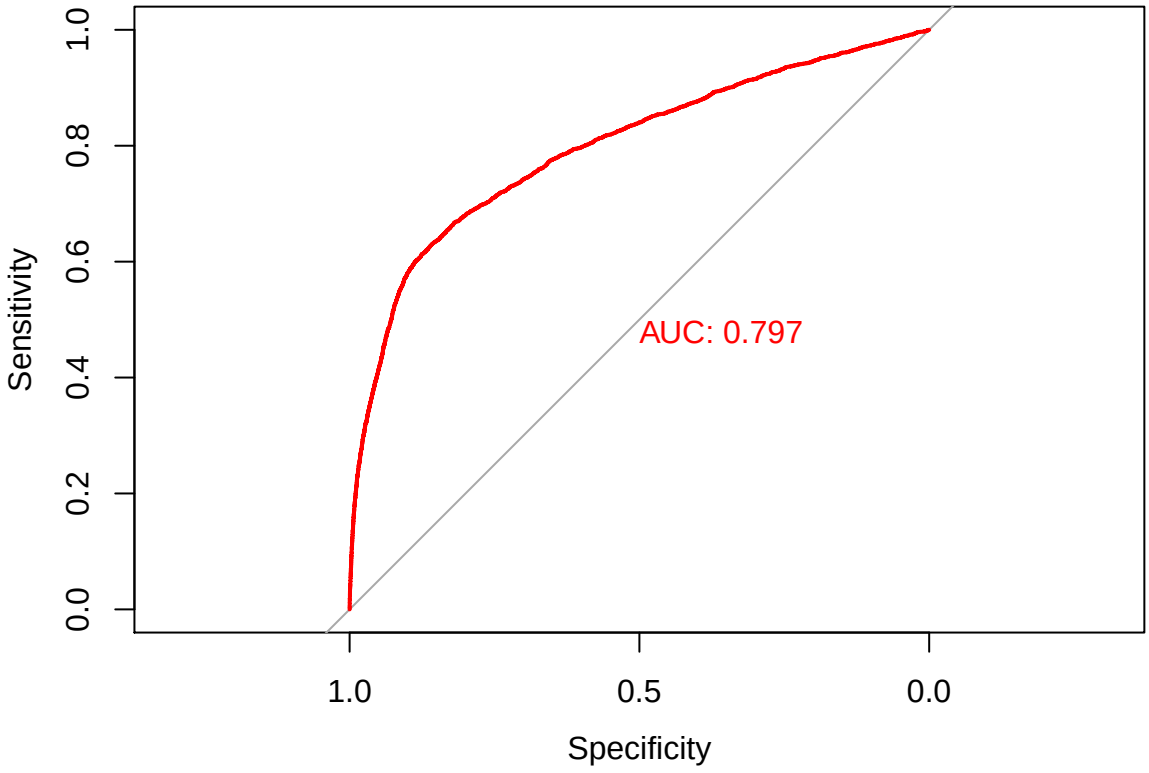
```
#Split Data Into Train And Test Data
sample_data = sample.split(clean_loan, SplitRatio = 0.70)
datatrainset <- subset(clean_loan, sample_data == TRUE)
datatestset <- subset(clean_loan, sample_data == FALSE)
```

```
#Fit Logisitic Regression Model To All variables
glmall <- glm(formula = Y ~ .,
                  family  = binomial(link = "logit"),
                  data    = datatrainset)

glmall2 <- glm(formula = Y ~ .,
                  family  = binomial(link = "logit"),
                  data    = datatestset)
```

```
#Select Variables With Less Than 0.1 P Value (Signifcant Variables) And Create Optimal Model
```

```
#Create GLM (Generalised Linear Model) With Signifcant Variables
glmoptimal <- glm(formula=Y~JOB+EDUCATION+DEFAULTCREDIT+CONTACT+MONTH+DAY_OF_WEEK+CAMPAIGN+PDAYS+POUTCO

glmoptimal2 <- glm(formula=Y~JOB+EDUCATION+DEFAULTCREDIT+CONTACT+MONTH+DAY_OF_WEEK+CAMPAIGN+PDAYS+POUTC
```

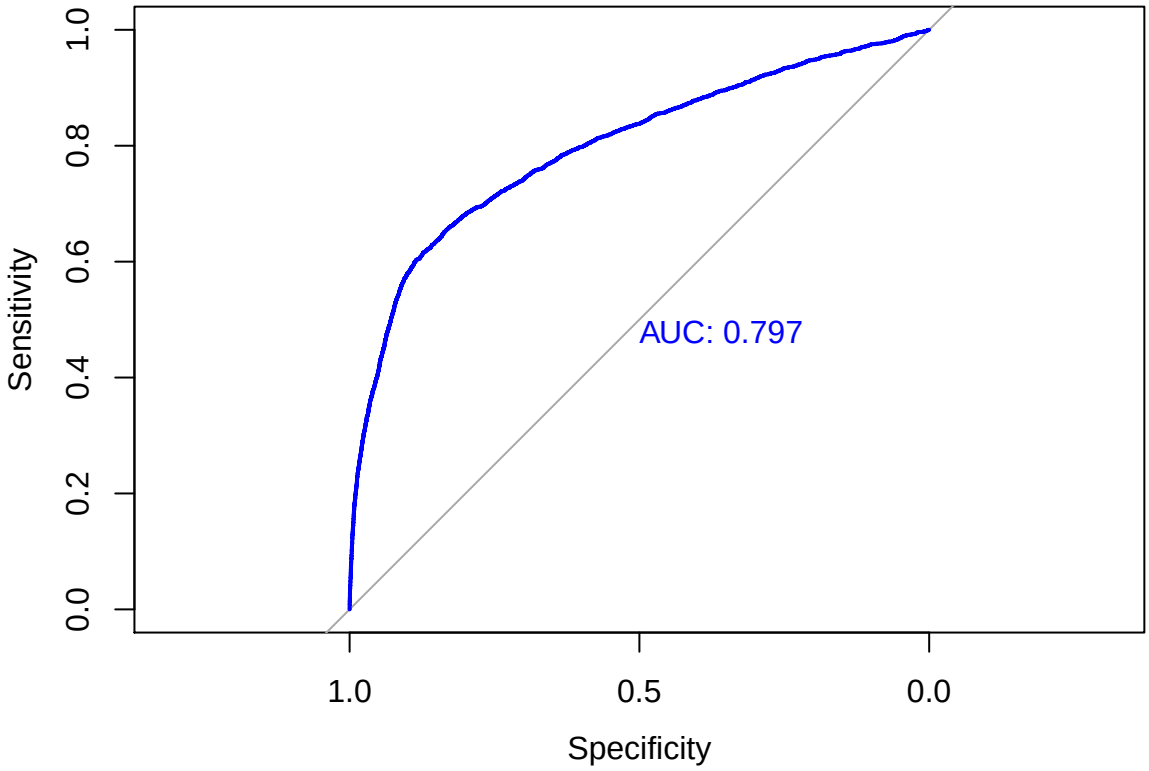
```
#Plot ROC Curve With Significant Variables
library(pROC)
prob <- predict(object = glmoptimal, newdata = datatrainset, type = "response")
ROC_SIG <- roc(datatrainset$Y,prob,col="red", plot = TRUE,print.auc =TRUE)
```



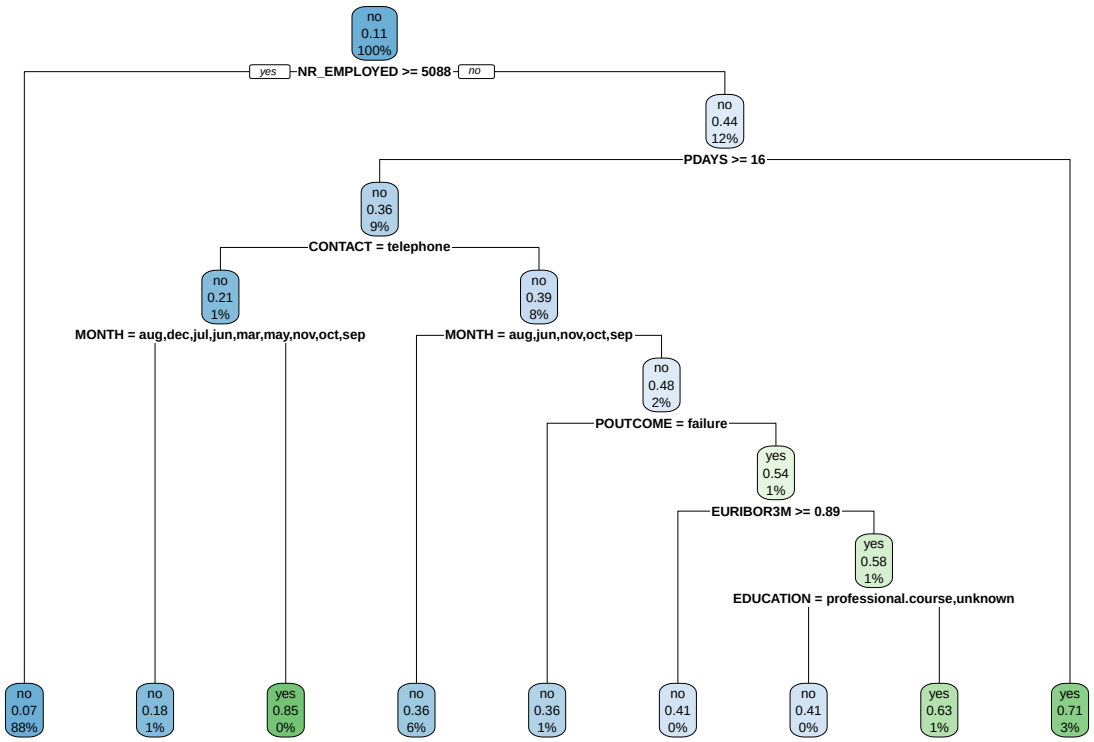

```
#Plot ROC With All Variables
prob2 <- predict(object = glmall, newdata=datatrainset, type = "response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

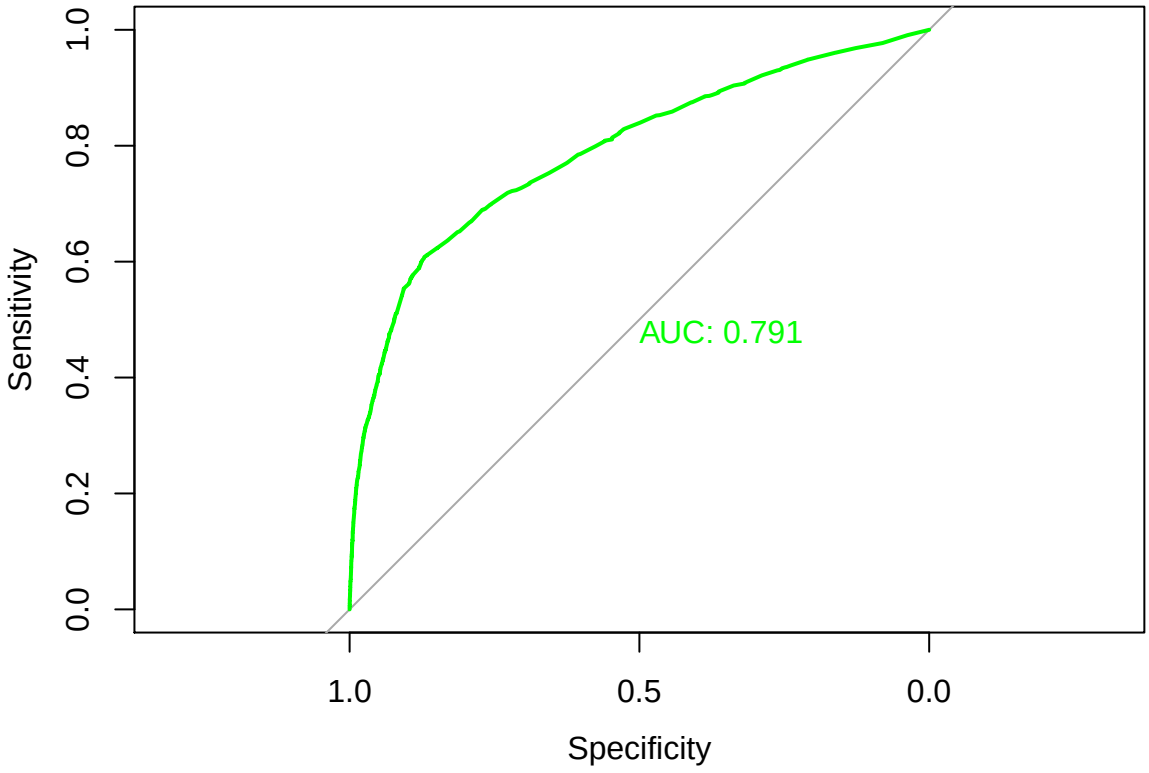
```
ROC_ALL <- smooth(roc(datatrainset$Y, prob2,col="blue",plot = TRUE,print.auc =TRUE))
```



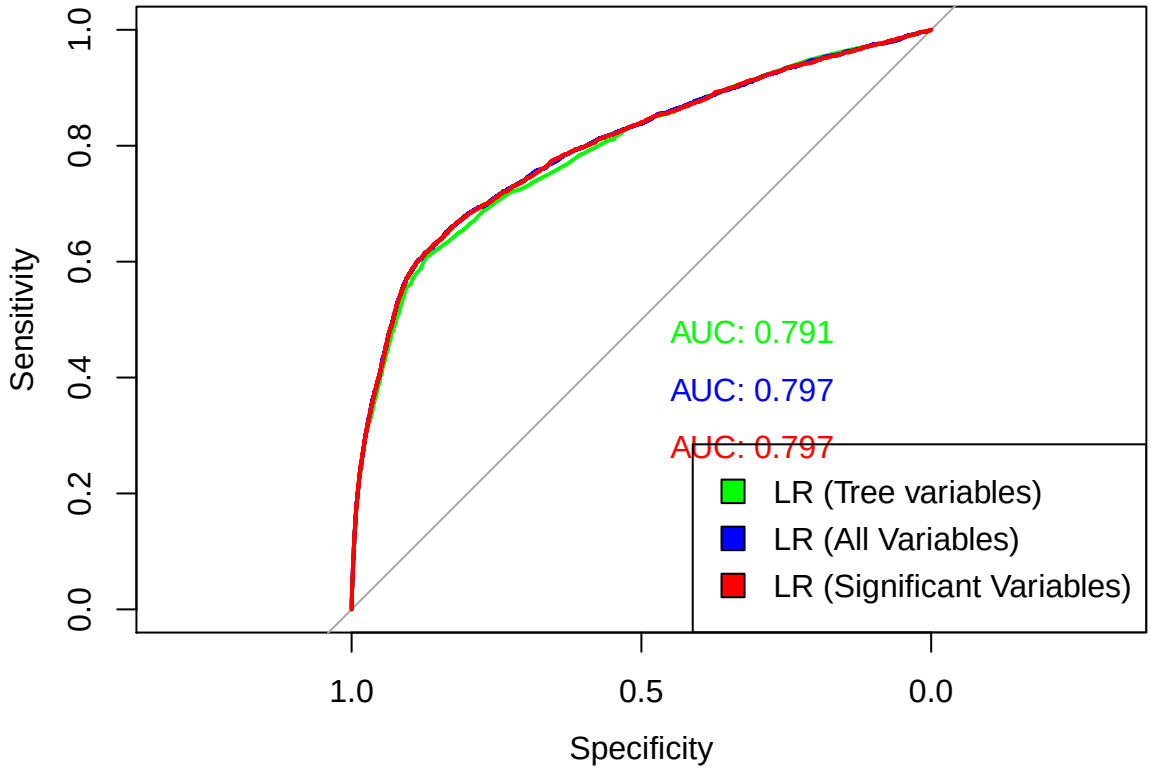

```
#Using Decision Tree Variables Plot ROC Curve For Comparison
binary.model <- rpart(Y~., data = datatrainset,cp=0.0025)
rpart.plot(binary.model)
```




```
glmdt <- glm(formula = Y ~ NR_EMPLOYED+POUTCOME+CONTACT+MONTH+EMP_VAR_RATE+PREVIOUS+DAY_OF_WEEK,
                  family  = binomial(link = "logit"),
                  data    = datatrainset)
glmdt2 <- glm(formula = Y ~ NR_EMPLOYED+POUTCOME+CONTACT+MONTH+EMP_VAR_RATE+PREVIOUS+DAY_OF_WEEK,
                  family  = binomial(link = "logit"),
                  data    = datatestset)

prob3 <- predict(object = glmdt, newdata = datatrainset, type = "response")
ROC_DT <- smooth(roc(datatrainset$Y, prob3,col="green",plot = TRUE,print.auc =TRUE))
```

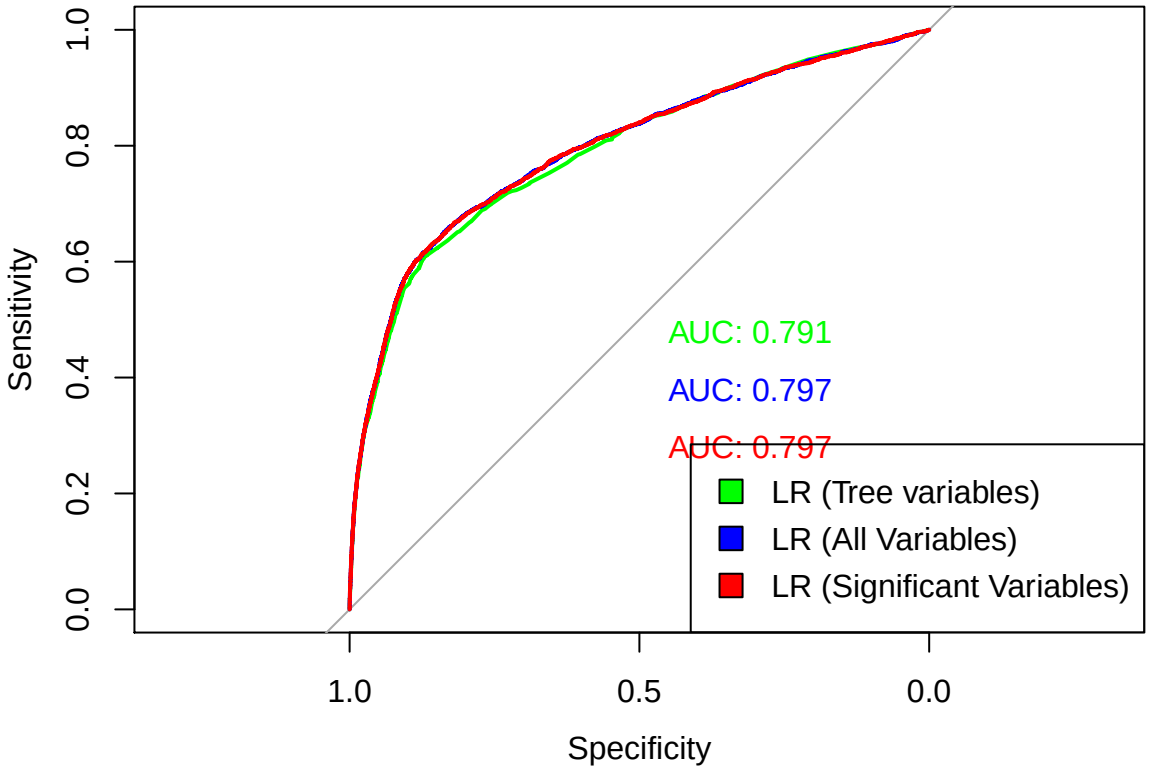



```
ROC_DT <- roc(datatrainset$Y, prob3,col="green",plot = TRUE,print.auc =TRUE,print.auc.y = 0.5,print.auc
ROC_ALL <- roc(datatrainset$Y, prob2,col="blue",plot = TRUE,print.auc =TRUE,add = TRUE,print.auc.y = 0.4
ROC_SIG <- roc(datatrainset$Y,prob,col="red", plot = TRUE,print.auc =TRUE,add = TRUE,print.auc.y = 0.3,p
legend("bottomright", legend = c("LR (Tree variables)", "LR (All Variables)", "LR (Significant Variables
 fill = c("green", "blue", "red"))
```

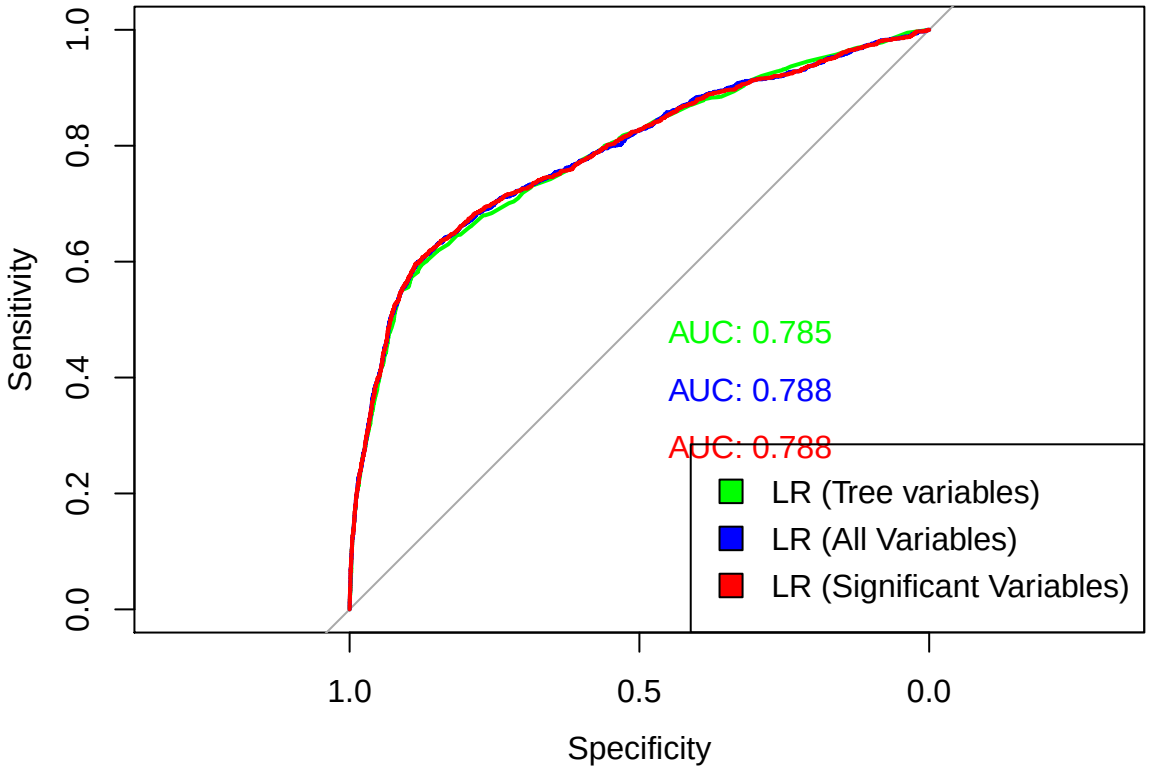



```
 ROC_PLOT <- function(loandata){
   if (loandata==TRUE){
     prob <- predict(object = glmoptimal, newdata = datatrainset, type = "response")
prob2 <- predict(object = glmall, newdata=datatrainset, type = "response")
prob3 <- predict(object = glmdt, newdata = datatrainset, type = "response")
ROC_DT <- roc(datatrainset$Y, prob3,col="green",plot = TRUE,print.auc =TRUE,print.auc.y = 0.5,print.auc
ROC_ALL <- roc(datatrainset$Y, prob2,col="blue",plot = TRUE,print.auc =TRUE,add = TRUE,print.auc.y = 0.4
ROC_SIG <- roc(datatrainset$Y,prob,col="red", plot = TRUE,print.auc =TRUE,add = TRUE,print.auc.y = 0.3,p

} else {
   prob7 <- predict(object = glmoptimal, newdata = datatestset, type = "response")
 prob6 <- predict(object = glmall, newdata= datatestset, type = "response")
 prob5 <- predict(object = glmdt, newdata = datatestset, type = "response")
ROC_DT2 <- roc(datatestset$Y, prob5,col="green",plot = TRUE,print.auc =TRUE,print.auc.y = 0.5,print.auc
ROC_ALL3 <- roc(datatestset$Y, prob6,col="blue",plot = TRUE,print.auc =TRUE,add = TRUE,print.auc.y = 0.4
ROC_SIG4 <- roc(datatestset$Y,prob7,col="red", plot = TRUE,print.auc =TRUE,add = TRUE,print.auc.y = 0.3
}
}
```

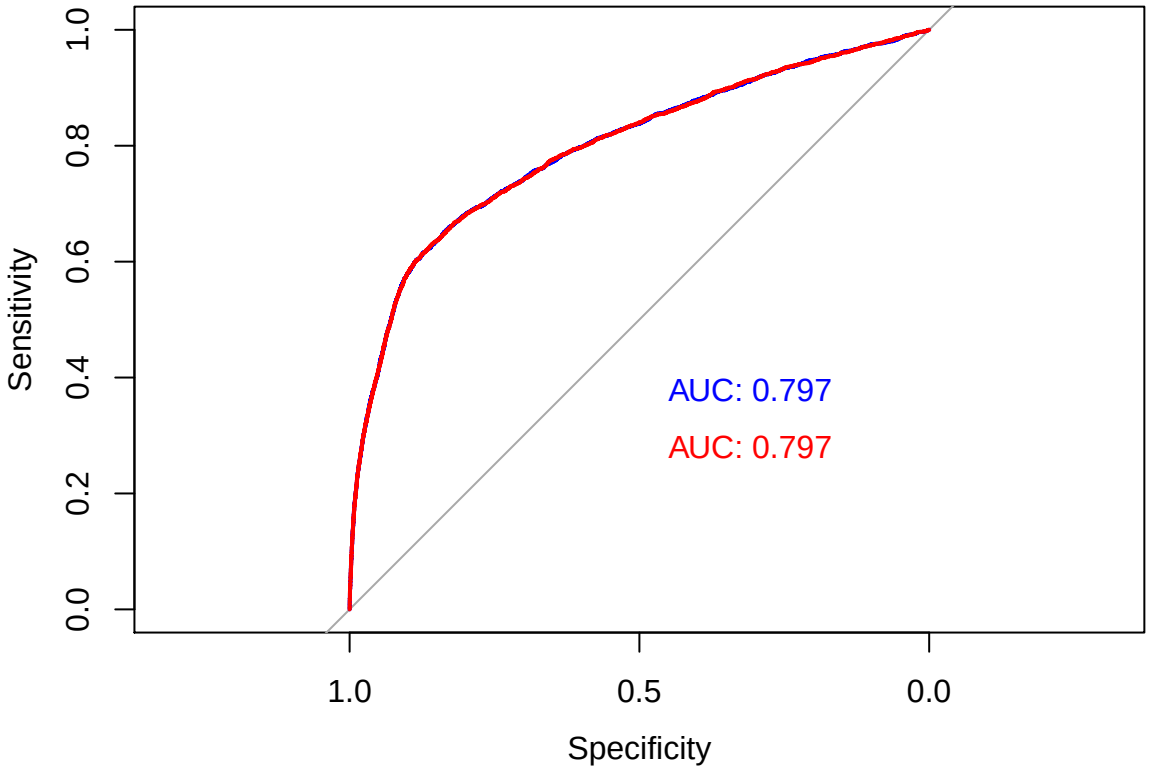
```
#Plot ROC Curves With Training Data
ROC_PLOT(loandata = TRUE)
legend("bottomright", legend = c("LR (Tree variables)", "LR (All Variables)", "LR (Significant Variables
 fill = c("green", "blue", "red"))
```




```
#Plot ROC Curves With Test Data
ROC_PLOT(loandata = FALSE)
legend("bottomright", legend = c("LR (Tree variables)", "LR (All Variables)", "LR (Significant Variables
 fill = c("green", "blue", "red"))
```




```
#Train Data ROC Curves
ROC_ALL <- roc(datatrainset$Y, prob2,col="blue",plot = TRUE,print.auc =TRUE,print.auc.y = 0.4,print.auc
ROC_SIG <- roc(datatrainset$Y,prob,col="red", plot = TRUE,print.auc =TRUE,add = TRUE,print.auc.y = 0.3,
```



ROC Curve

The ROC Graph measures how well the model can perform in terms of specificty and sensivity. Sensitivity measures how well the model can correctly predict that a person will default whislt specifcity is a measure on how well the model can correctly predict that someone won't default on their loan. In order for this to function they need to have an equal balance as if one of the metrics is out of proportion, it will weigh out the other and reduce it. The best models are ones which are the furthest away from the linear line in curvature as shown above, the model could be better aswell. The AUC (Area Under The Curve), is a measure of accuracy and as shown above it is very similar for both the models.

Training Data Accuracy

```
library(MLmetrics)
```

```
##
## Attaching package: 'MLmetrics'
```

```
## The following objects are masked from 'package:caret':
##
##     MAE, RMSE
```

```
## The following object is masked from 'package:base':
##
##     Recall
```

```
pred6 <- ifelse(glmoptimal$fitted.values < 0.5, 0, 1)
CM3 <- ConfusionMatrix(y_pred = pred6, y_true = datatrainset$Y)
```

```r
sum(diag(CM3))/sum(CM3)
```

```
## [1] 0.9013249
```

```r
print(CM3)
```

```
##       y_pred
## y_true     0     1
##    no  25206   380
##    yes  2465   781
```

Testing Data Accuracy

```r
pred6 <- ifelse(glmoptimal$fitted.values < 0.5, 0, 1)
CM6 <- ConfusionMatrix(y_pred = pred6, y_true = datatrainset$Y)
sum(diag(CM6))/sum(CM6)
```

```
## [1] 0.9013249
```

```r
print(CM6)
```

```
##       y_pred
## y_true     0     1
##    no  25206   380
##    yes  2465   781
```

```r
#Accuracy For Test Models
pred2 <- ifelse(glmall2$fitted.values < 0.5, 0, 1)
CM4 <- ConfusionMatrix(y_pred = pred2, y_true = datatestset$Y)
sum(diag(CM4))/sum(CM4)
```

```
## [1] 0.8993202
```

```r
pred3 <- ifelse(glmdt2$fitted.values < 0.5, 0, 1)
CM5 <- ConfusionMatrix(y_pred = pred3, y_true = datatestset$Y)
sum(diag(CM5))/sum(CM5)
```

```
## [1] 0.8987536
```

```r
pred4 <- ifelse(glmoptimal2$fitted.values < 0.5, 0, 1)
CM6 <- ConfusionMatrix(y_pred = pred4, y_true = datatestset$Y)
sum(diag(CM6))/sum(CM6)
```

```
## [1] 0.8998058
```

```r
#Sensitivty Analysis For Train Models
SP1 <- CM1[1,1]/(CM1[1,1] + CM1[1,2])
print(SP1)
```

```
## [1] 0.98507
```

```r
SP2 <- CM2[1,1]/(CM2[1,1] + CM2[1,2])
print(SP2)
```

```
## [1] 0.9876886
```

```r
SP3 <- CM3[1,1]/(CM3[1,1] + CM3[1,2])
print(SP3)
```

```
## [1] 0.9851481
```

```r
#Sensitivty Analysis For Test Models
SP4 <- CM4[1,1]/(CM1[1,1] + CM1[1,2])
print(SP4)
```

```
## [1] 0.4220668
```

```r
SP5 <- CM5[1,1]/(CM2[1,1] + CM2[1,2])
print(SP5)
```

```
## [1] 0.4229657
```

```r
SP6 <- CM6[1,1]/(CM3[1,1] + CM3[1,2])
print(SP6)
```

```
## [1] 0.4225748
```

```r
#Specificity Analysis For Train Models
SN1 <- CM1[2,2]/(CM1[2,2] + CM1[2,1])
print(SN1)
```

```
## [1] 0.2406038
```

```r
SN2 <- CM2[2,2]/(CM2[2,2] + CM2[2,1])
print(SN2)
```

```
## [1] 0.2159581
```

```r
SN3 <- CM3[2,2]/(CM3[2,2] + CM3[2,1])
print(SN3)
```

```
## [1] 0.2406038
```

```r
#Sensitivity Analysis For Test Models
SN4 <- CM4[2,2]/(CM4[2,2] + CM4[2,1])
print(SN4)
```

```
## [1] 0.2245337
```

```r
SN5 <- CM5[2,2]/(CM5[2,2] + CM5[2,1])
print(SN5)
```

```
## [1] 0.2030129
```

```r
SN6 <- CM6[2,2]/(CM6[2,2] + CM6[2,1])
print(SN6)
```

```
## [1] 0.2195122
```

```r
#Precision Analysis For Train Models
SN1 <- CM1[1,1]/(CM1[1,1] + CM1[2,1])
print(SN1)
```

```
## [1] 0.9109111
```

```r
SN2 <- CM2[1,1]/(CM2[1,1] + CM2[2,1])
print(SN1)
```

```
## [1] 0.9109111
```

```r
SN3 <- CM2[1,1]/(CM2[1,1] + CM2[2,1])
print(SN1)
```

```
## [1] 0.9109111
```

```
#Precision Analysis For Test Models
SN4 <- CM4[1,1]/(CM4[1,1] + CM4[2,1])
print(SN4)
```

```
## [1] 0.9090067
```

```
SN5 <- CM5[1,1]/(CM5[1,1] + CM5[2,1])
print(SN5)
```

```
## [1] 0.9068968
```

```
SN6 <- CM6[1,1]/(CM6[1,1] + CM6[2,1])
print(SN6)
```

```
## [1] 0.9085714
```

Data Metrics

It could be seen that the sensitivty and specifity of the Train and Decision Tree variables wsa high and this was due to overffitting of the model and is not a suitbale way to determine of the model was good. When the test set of the data was applied, the values were lower which is correct as this was the case due to the model seeing foregin data and testing it on the model. Overall the model is suitable to be used but needs further refinement.

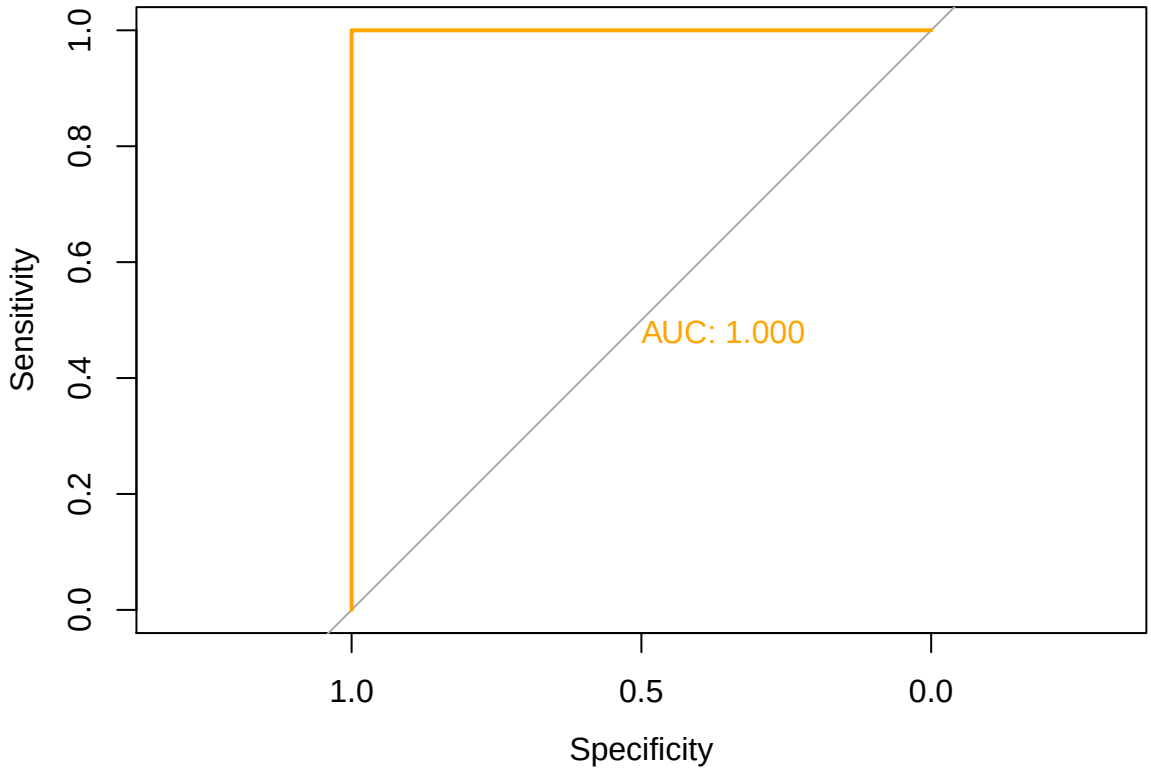
```
#Make New Customer Data
dummydata <- data.frame("JOB" = c("services","unemployed","blue-collar"),
                        "DEFAULTCREDIT" = c("no","no","yes"),
                        "CONTACT" = c("cellular" , "telephone" , "telephone"),
                        "MONTH" = c ("may","oct","aug"), "DAY_OF_WEEK" = c("tue","wed","mon"),
                        "CAMPAIGN" = (5), "PDAYS" = c(10, 999, 7),
                        "POUTCOME" = c("failure","nonexistent","success"),
                        "EMP_VAR_RATE" = c(-3.6,-2.0,2.3),
                        "CONS_PRICE_IDX" = c(98.9444, 95.423,92.102),
                        "CONS_CONF_IDX" = c(-44.0, -47.5, -33.90),"NR_EMPLOYED" = c(5200, 5000, 6100),"
)

dummydata
```

```
##           JOB DEFAULTCREDIT   CONTACT MONTH DAY_OF_WEEK CAMPAIGN PDAYS
## 1    services            no  cellular   may         tue        5    10
## 2  unemployed            no telephone   oct         wed        5   999
## 3 blue-collar           yes telephone   aug         mon        5     7
##      POUTCOME EMP_VAR_RATE CONS_PRICE_IDX CONS_CONF_IDX NR_EMPLOYED
## 1     failure         -3.6        98.9444         -44.0        5200
## 2 nonexistent         -2.0        95.4230         -47.5        5000
## 3     success          2.3        92.1020         -33.9        6100
##           EDUCATION   Y
## 1          basic.4y  no
## 2          basic.9y yes
## 3 university.degree yes
```

New Customer Predictions

As it can be seen from above new customer data was provided to the model which generate the following probability of defaults for Customer(s) 1,2 and 3 respectively:

```r
#Determine New Customer Data Predictions
glmdummy <- glm(formula=Y~JOB+EDUCATION+DEFAULTCREDIT+CONTACT+MONTH+DAY_OF_WEEK+CAMPAIGN+PDAYS+POUTCOME+

yprobnew <- predict(object = glmdummy, newdata = dummydata, type = "response")
rocdummy <- roc(predictor = yprobnew, response = dummydata$Y,plot = TRUE, col="orange",print.auc = TRUE)
```




```r
print(yprobnew)
```

```
##           1           2           3
## 0.999999749 0.929574276 0.001245695
```

Model Conclusion

The model made which will be presented to the management team, using the GLM package using the most significant variables from the dataset to predict the likelihood of a customer defaulting on their loan. The variables picked are the most significant in that they are able to predict the chance of default being either yes or no the best on useen data. This will allow the team to make predictions best suited for the purposes of the firm.

The variables used to make this decision were:

-Job -Education -Default Credit -Contact -Month -Day Of The Week -Campaign -PDAYS -POutcome -Emp Var Rate -Cons Price IDX -Cons Conf IDX -NR Employed

Model Accuracy

The best model resulted in 90% overall accuracy. This is a high accuracy and it performed the same on the train and test data.

The model however is not the ideal model to be used and will be refined in the future and tweaked for accurate predictions but it is suited for the needs and purposes specified until further updates are developed.