Submitted by Akshay Malik

***

### 1) Conceptual Question

#### This first question is conceptual and written responses are expected. For each item below, indicate whether the appropriate method would be classification or regression, and whether we are most interested in inference or prediction. Please include a written sentence or two explaining why you made this choice. Also, indicate what n and p are for each section.

> **(a)** A dataset contains data for **350** manufacturing companies in Europe. The following variables are included in the data for each company: **industry, number of employees, salary of the CEO**, and **total profit.**
> **We are interested in learning which variables impact the CEO's salary.**

Ans.
This is a question of inference because we will try and interpret the role/contribution of observations towards a dependent variable and not predict the value of the dependent variable itself (the salary). The model appropriate for this scenario is a regression model which would show which of independent variables are statistically significant and how much impact they have on CEO's salary. The n in this case is the sample set of 350 observations and p is 4 - number of independent variables (or regressors).

> **(b)** A market research company is hired to help a startup analyze their new product. **We want to know whether the product will be a success or failure.** Similar products exist on the market so the market research company gathers data on 31 similar products. The company records the following data points about each previously launched product: **price of the product, competition price, marketing budget, ten other variables,** and **whether or not it succeeded or failed.**

Ans.
This is a question of prediction because the model will try and predict whether the outcome will be a success or a failure. The appropriate model for this scenario would be a classification model because the outcome or the dependent variable we wish to predict is categorical (binary – two choices) and not a continuous variable. Here, n- number of samples is 31 and p is the number of classifiers, in this case being 14. (Although logistic regression would also work, but since we had to choose one that gives a discrete outcome of either success or failure, I would recommend classification. Logistic regression would have an output of probability of success which is a continuous variable and not the outcome desired in this case)

> **(c)** Every week data is collected for the world stock market in 2012. The data points collected include the % change in the dollar, the % change in the market in the United States, the % change in the market in China, and

the % change in the market in France. **We are interested in predicting the % change in the dollar in relation to the changes every week in the world stock markets.**

Ans.
This is a question of prediction and not inference because the outcome must be the value of the dependent variable. Since the outcome desired is a continuous numeric value, we would use a regression model (OLS or random forest or any of the other predictive modeling techniques) to predict the % change in dollar in relation to changes in stock markets worldwide. Here the n – number of observations is 52 (number of weeks in a year) and p – number of predictors would be 3 (US, China and France markets).

### 2) Applied Question

#### For this second applied question you will develop several predictive models. These should be written in R or Python and the code should be submitted. The models will predict whether a car will get high or low gas mileage. The question will be based on the Cars_mileage data set that is a part of this repo.

> **(a)** Create a binary variable that represents whether the car's mpg is above or below its median. Above the median should be represented as 1. Name this variable **mpg_binary**.

> **(b)** Which of the other variables seem most likely to be useful in predicting whether a car's mpg is above or below its median? **Describe your findings and submit visual representations of the relationship between mpg_binary and other variables.**

> **(c)** Split the data into a training set and a test set.

> **(d)** Perform two of the following in order to predict mpg_binary:

>> **LDA** (Linear discriminant analysis)
>> **QDA** (Quadratic discriminant analysis)
>> **Logistic regression**
>> **KNN** (K-nearest neighbors)
>> **Decision Tree**
>> **Random Forests**
>> **Gradient Boosting**
>> **LASSO** (Least Absolute Shrinkage and Selection Operator)
>> **Elastic Net Method**
>> **Ridge regression analysis**

> For each of the two you select: **What is the test error(s) of the model obtained? Do you have any other observations?**

Ans.

To answer the second questions, I have used two methods – Logistic regression and Random Forest Classification.

We begin by pulling in the data and looking at it's structure and value types

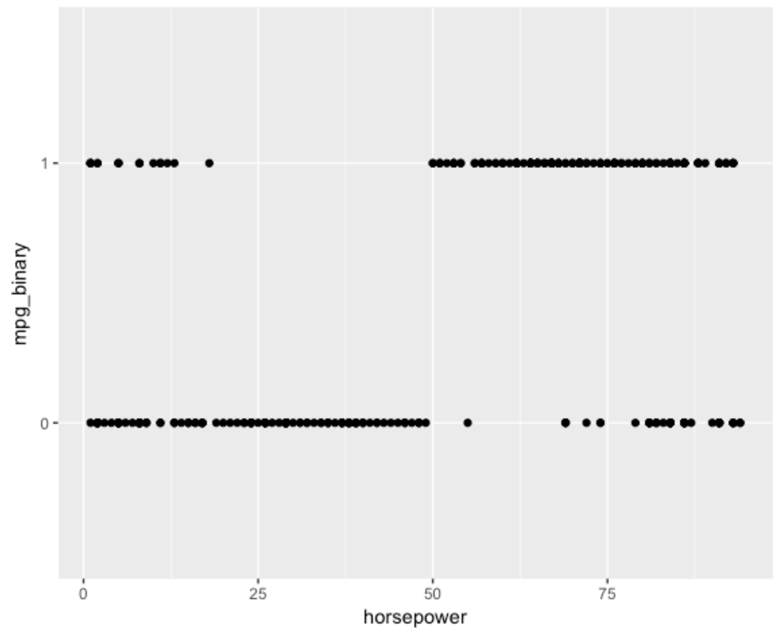
```
#pull in data
setwd("~/Documents/Github Code/data-analyst-data-test")
carData <- read.csv("Cars_mileage.csv")

#Reading data
head(carData)
str(carData)
summary(carData)
```

As suggested in the question, a new variable called mpg_binary is added to the data set in both models that sets to 1 is the mpg value is above the median value of mpg (23) and to 0 is not. Before we split the data, we must cast some nominal variables as factors.

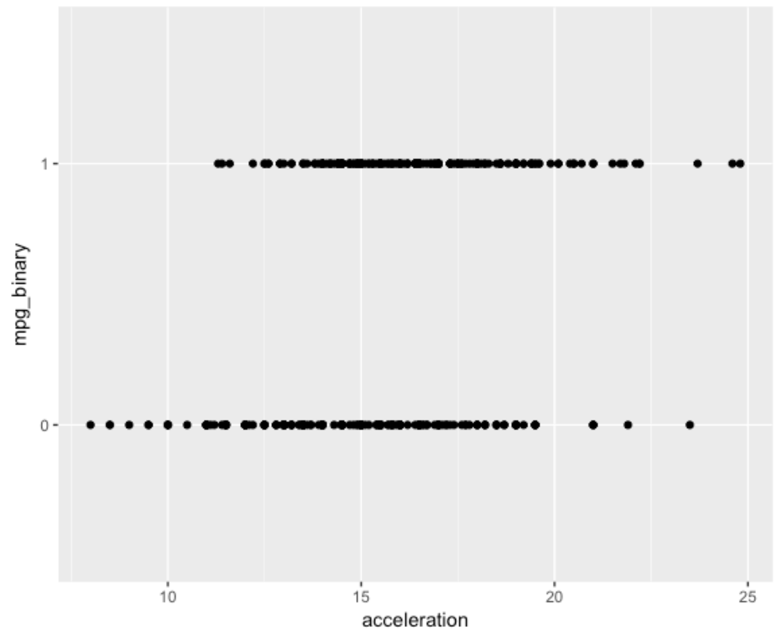
```
#create a new column called mpg_binary
medMPG <- median(carData$mpg)
carData$mpg_binary <- with(carData, ifelse(mpg >median(mpg), 1, 0 ))
carData$mpg_binary <- as.factor(carData$mpg_binary)
carData$year <- as.factor(carData$year)
carData$origin <- as.factor(carData$origin)
carData$cylinders <- as.factor(carData$cylinders)
carData$horsepower <- as.integer(carData$horsepower)
```

Here the dependent variable is mpg_binary and all others are considered independent variable. The name is not taken into consideration as a variable as it is nominal and related to no format of value structure. In order to gauge their relative impact on the dependent variable, we will try and see some plots and charts

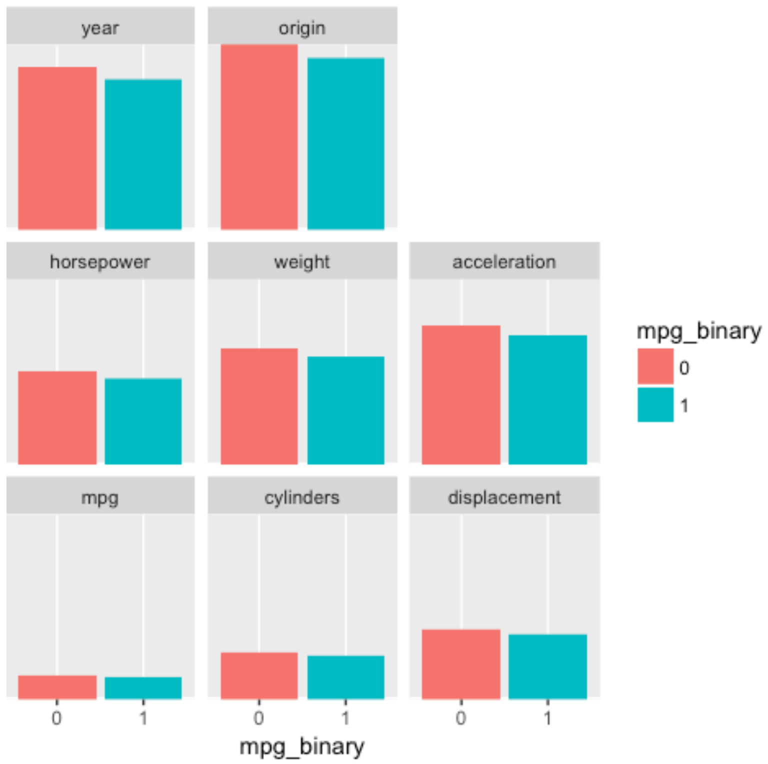
```
#plotting some varibales
ggplot(data =carData) +
  geom_point(mapping = aes(x= horsepower , y= mpg_binary ))
```



```
ggplot(data =carData) +
  geom_point(mapping = aes(x= acceleration, y= mpg_binary))
```



```
#plotting by melting along the dependent variable
carMelt <- carData[,c(1:8, 10)]
head(carMelt)
meltMpg <- melt(carMelt, id ='mpg_binary')
str(meltMpg)

ggplot(meltMpg, aes(mpg_binary, variable, fill= mpg_binary)) +
 geom_bar(stat = "identity" ) +
  facet_wrap(~variable, as.table = FALSE )
```



This shows that horsepower, weight and displacement may have a greater predicting power as compared to others.

Both the models follow a simple rule of split – train – test. The data is manually split into two halves but setting a seed to 198 and creating two sets

```
#split the data manually
set.seed(198)
trainIndex <- createDataPartition(carData$mpg_binary, p = .5,
                                  list = FALSE,
                                  times = 1)
mpgTrain <- carData[ trainIndex,]
mpgTest  <- carData[-trainIndex,]
head(mpgTrain)
```

1. Logistic regression:

In order to use logistic regression, I modelled multiple formulas to test them using various metrics for good ness of fit. A few and final are as follows:

```
# First one with all variables except name
glm(formula = mpg_binary ~ cylinders + displacement + horsepower +
    weight + acceleration + year + origin, family = binomial(link = "logit"),
    data = carData)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.47927  -0.02711  -0.00015   0.04085   2.55963

Coefficients:
              Estimate Std. Error z value  Pr(>|z|)
(Intercept)  15.515553   6.418248   2.417   0.01563 *
cylinders4    6.180378   5.049975   1.224   0.22101
cylinders5    7.313010   5.431018   1.347   0.17813
cylinders6    3.097630   5.123940   0.605   0.54548
cylinders8    8.792336   5.826388   1.509   0.13128
displacement -0.005792   0.014439  -0.401   0.68832
horsepower    0.007545   0.011019   0.685   0.49349
weight       -0.007833   0.001941  -4.035 0.0000546 ***
acceleration -0.012469   0.128359  -0.097   0.92261
year71       -3.144683   1.882002  -1.671   0.09474 .
year72       -3.992411   1.743747  -2.290   0.02205 *
year73       -4.009563   1.765749  -2.271   0.02316 *
year74        1.952220   3.057577   0.638   0.52316
year75       -1.253785   1.696132  -0.739   0.45978
year76        2.864146   2.163997   1.324   0.18565
year77        0.464536   1.931578   0.240   0.80995
year78        0.037286   1.776620   0.021   0.98326
year79        4.359410   1.879929   2.319   0.02040 *
year80        7.012713   2.711280   2.586   0.00970 **
year81        7.137487   2.128766   3.353   0.00080 ***
year82        6.374181   2.141525   2.976   0.00292 **
origin2       0.726344   0.889353   0.817   0.41409
origin3       0.759288   0.884168   0.859   0.39047
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 549.792  on 396  degrees of freedom
Residual deviance:  90.598  on 374  degrees of freedom
AIC: 136.6

Number of Fisher Scoring iterations: 9

> pR2(mpg1)
         llh      llhNull           G2     McFadden         r2ML         r2CU 
 -45.2989156 -274.8959879  459.1941446    0.8352143    0.6854651    0.9143890 
```

As we analyze this result, it is important to note that McFadden is above 0.4 and hence shows over fitting. This could be caused because of one or more variables. To test I ran a few models and found that displacement and weight cause over-fitting. This was also intuitive from our readings of the bar graphs comparing all variables earlier.

Eventually the model that I believe is the best model to predict using logistic regression without over-fitting is as follows:

```
# Best Model - McFadden is between 0.2 - 0.4  and AIC value highest among models tested
mpg4 <- glm(mpg_binary ~ horsepower + acceleration + origin,
        family="binomial"(link="logit"), data = carData)
summary(mpg4)
pR2(mpg4)

#95% confidence intervals
confint.default(mpg4)
```

```
glm(formula = mpg_binary ~ horsepower + acceleration + origin, 
    family = binomial(link = "logit"), data = carData)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.4662  -0.5694  -0.4285   0.6308   2.3051  
```

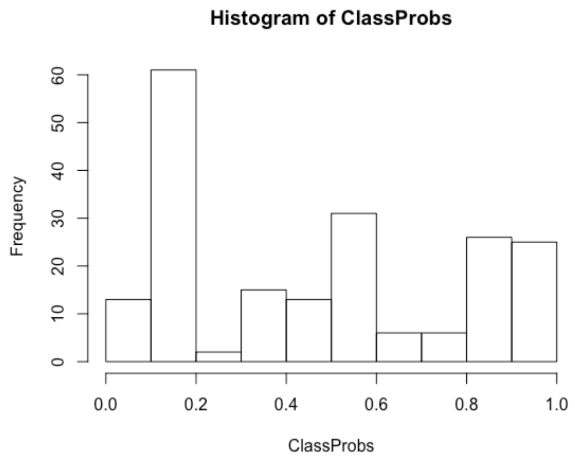
```
Coefficients:
              Estimate Std. Error z value        Pr(>|z|)    
(Intercept)  -4.455643   0.799012  -5.576 0.0000000245484 ***
horsepower    0.026914   0.004577   5.880 0.0000000040979 ***
acceleration  0.139444   0.052394   2.661         0.00778 ** 
origin2       1.850439   0.344640   5.369 0.0000000790906 ***
origin3       2.537576   0.391524   6.481 0.0000000000909 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 549.79  on 396  degrees of freedom
Residual deviance: 369.18  on 392  degrees of freedom
AIC: 379.18

Number of Fisher Scoring iterations: 4

> pR2(mpg4)
        llh      llhNull           G2     McFadden         r2ML         r2CU 
-184.5882571 -274.8959879  180.6154616    0.3285160    0.3655208    0.4875933 

> 
> #95% confidence intervals
> confint.default(mpg4)
                  2.5 %      97.5 %
(Intercept)  -6.0216772 -2.88960954
horsepower    0.0179434  0.03588543
acceleration  0.0367532  0.24213447
origin2       1.1749568  2.52592172
origin3       1.7702030  3.30494858
```

This model has optimal McFadden value (0.2 – 0.4), has all variables that are statistically significant and has an AIC value highest among those that do not over fit. AIC value shows loss of data in comparison to gain in complexity of the model and here we see that our final model has the least amount of loss while gaining maximum complexity.

Now, we use this model to test it on our test set that we created earlier:

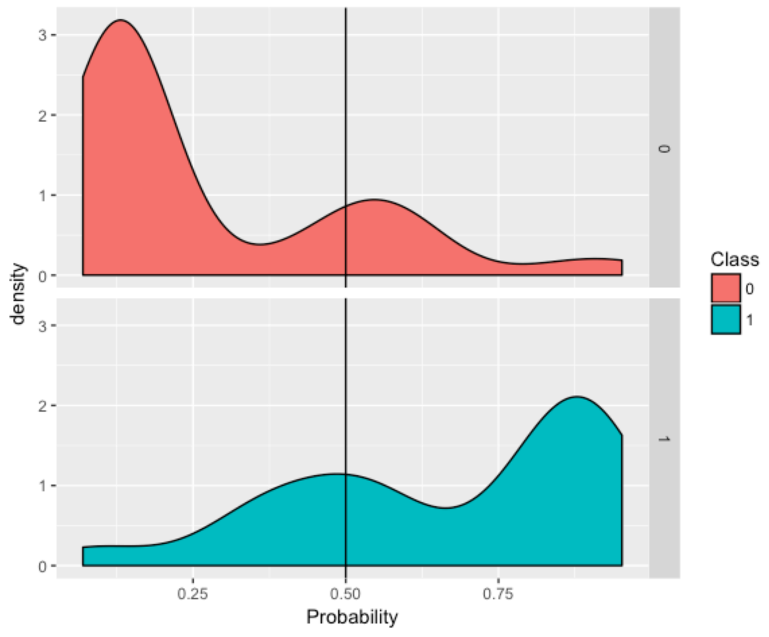
```
#using mpg4 to predict on test set
ClassProbs <- predict(mpg4, mpgTest, type="response")
```

```
#lets look at the distribution of the probabilities
hist(ClassProbs)
```



```
#put these into a data frame so we can compare probabilities with actual
testProbs <- data.frame(Class = mpgTest$mpg_binary , Probs = ClassProbs)
head(testProbs)

> head(testProbs)
   Class      Probs
1      0 0.08909201
3      0 0.10515237
7      0 0.12612419
11     0 0.11250836
12     0 0.08128589
14     0 0.14562779
```

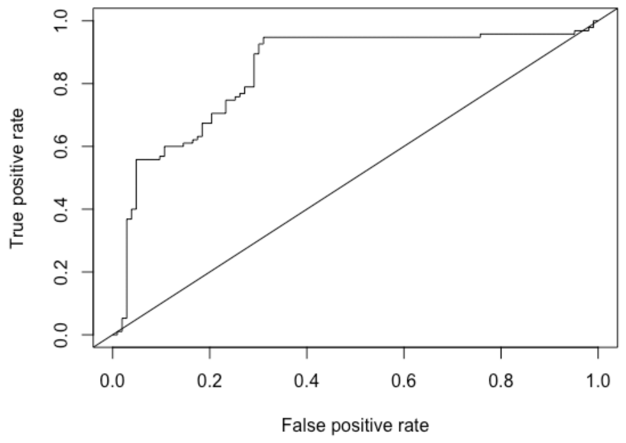
```
#plot the distrubtion of predictied probabilities for each binary output.
testProbsPlot <- ggplot(testProbs, aes(x = Probs, fill= Class)) +
geom_density() +
  facet_grid(Class ~ .) + xlab("Probability") + geom_vline(xintercept = .5)
testProbsPlot
```



```
#This shows that our model did well in predicting 1 and 0 as they are in
terms of their probability of being 1 and 0 respectively.

#create a cutoff threshhold
testProbs$predClass  = ifelse(testProbs$Probs > .5 ,1,0)
head(testProbs)

> head(testProbs)
   Class      Probs predClass
1      0 0.08909201         0
3      0 0.10515237         0
7      0 0.12612419         0
11     0 0.11250836         0
12     0 0.08128589         0
14     0 0.14562779         0


#lets do a confusion matrix
confusionMatrix(testProbs$Class ,testProbs$predClass)

Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 79 24
         1 25 70
```

```
               Accuracy : 0.7525
                 95% CI : (0.6864, 0.811)
    No Information Rate : 0.5253
    P-Value [Acc > NIR] : 0.00000000004026

                  Kappa : 0.504
 Mcnemar's Test P-Value : 1

            Sensitivity : 0.7596
            Specificity : 0.7447
         Pos Pred Value : 0.7670
         Neg Pred Value : 0.7368
             Prevalence : 0.5253
         Detection Rate : 0.3990
   Detection Prevalence : 0.5202
      Balanced Accuracy : 0.7521

       'Positive' Class : 0
```

```
#an ROC Curve
pred <- prediction( testProbs$Probs, testProbs$Class)
perf <- performance(pred,"tpr","fpr")
plot(perf)
abline(a=0, b= 1)
```



```
#lets calculate Area Under the Curve or 'AUC'
> auc(testProbs$Class, testProbs$Probs)
Area under the curve: 0.8405
```

The test error for this logistic regression model can be judged from the output of the confusion matrix. Evidently, the model has 0.75 accuracy and receiver operating characteristic curve, or ROC curve has 0.8 as its area under the curve. ROC curve illustrates the performance of this model.

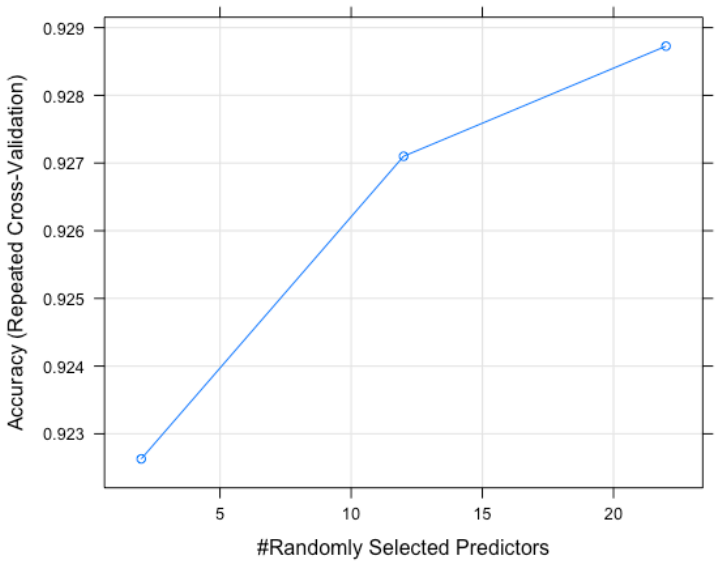
2. Random Forest:

To carry the same model forward in order to use random forest classification algorithm to determine the predicted values of mpg_binary we use the following code:

```
#Using caret library
#fit in random forest
fitControl <- trainControl(method = "repeatedcv", number = 10, repeats = 10)

#Setting seed allows the model to be reproducible as it sets the number of
trees to a fixed number

set.seed(100)

#fit in random forest
rfMPG <- train(mpg_binary ~ ., data = mpgTrain, method = "rf", trControl =
fitControl,metric= 'Accuracy', importance=TRUE)
rfMPG

> rfMPG
Random Forest

199 samples
  7 predictor
  2 classes: '0', '1'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 10 times)
Summary of sample sizes: 179, 180, 179, 178, 178, 179, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   2    0.9226291  0.8453177
  12    0.9271003  0.8539168
  22    0.9287268  0.8570751
```

```
Accuracy was used to select the optimal model using  the largest value.
The final value used for the model was mtry = 22
```

```
plot(rfMPG)
```



```
#look at the variable importance - showing how much each variable impacts
varImp(rfMPG)

rf variable importance

  only 20 most important variables shown (out of 22)

             Importance
weight          100.000
cylinders4       95.157
year72           53.350
acceleration     44.985
year81           35.845
year80           33.882
displacement     31.955
year82           30.050
year73           24.861
year75           13.944
```

```
year78           12.617
cylinders6       12.447
origin3          10.972
year77            9.601
origin2           7.651
horsepower        6.894
year79            5.268
year71            3.644
cylinders5        3.644
year74            3.644

#predicted values on the test set
predProb <- predict(rfMPG,mpgTest)

#lets do a confusion matrix comparing predicted vs actual
confusionMatrix(mpgTest$mpg_binary , predProb)

          Reference
Prediction  0  1
         0 89 14
         1  8 87

               Accuracy : 0.8889
                 95% CI : (0.8366, 0.929)
    No Information Rate : 0.5101
    P-Value [Acc > NIR] : <0.0000000000000002

                  Kappa : 0.778
 Mcnemar's Test P-Value : 0.2864

            Sensitivity : 0.9175
            Specificity : 0.8614
         Pos Pred Value : 0.8641
         Neg Pred Value : 0.9158
             Prevalence : 0.4899
         Detection Rate : 0.4495
   Detection Prevalence : 0.5202
      Balanced Accuracy : 0.8895

       'Positive' Class : 0
```

The test error for this random forest model can be judged from the output of the confusion matrix. The model has 0.88 accuracy (much higher than logistic regression).