FINAL SUBMISSION 2

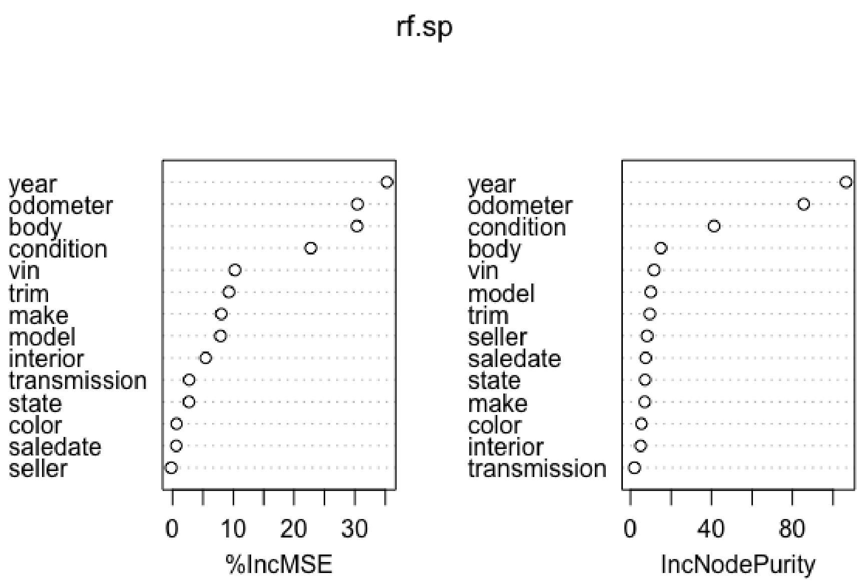
By - Rushat Chopra (20854690)

Accuracy of Algorithm 1 in Test set:
MSE : 0.2898476
R^2 : 0.6500216

Accuracy of Algorithm 1 in Training set:
MSE : 0.2922281
R^2 : 0.6439893

Algorithm 1 type: Random forest and Regression trees

Graph that measures variable importance:

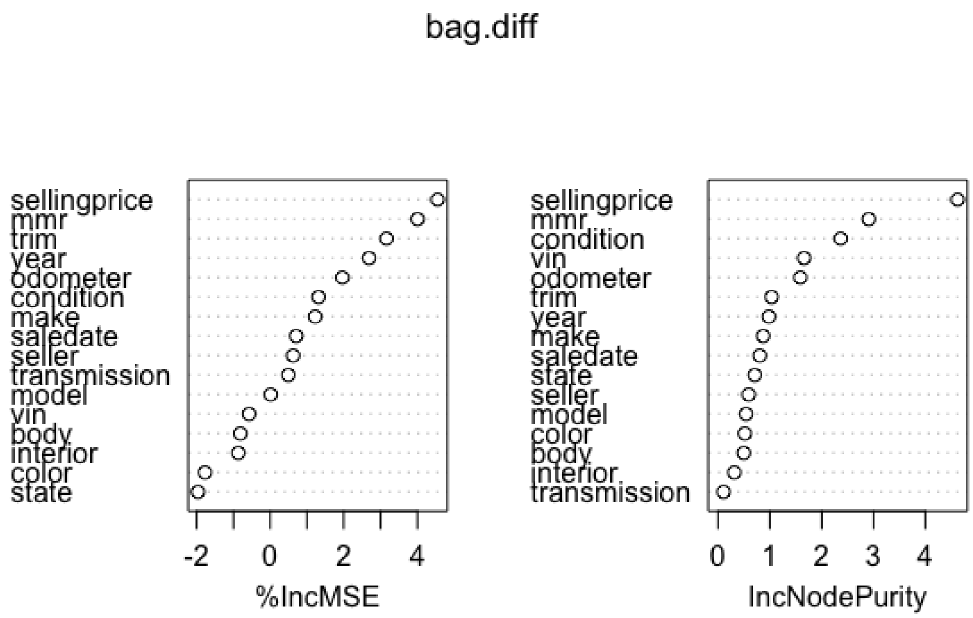


Accuracy of Algorithm 2 in Test set:
MSE : 0.0111818
R^2 : 0.8127233

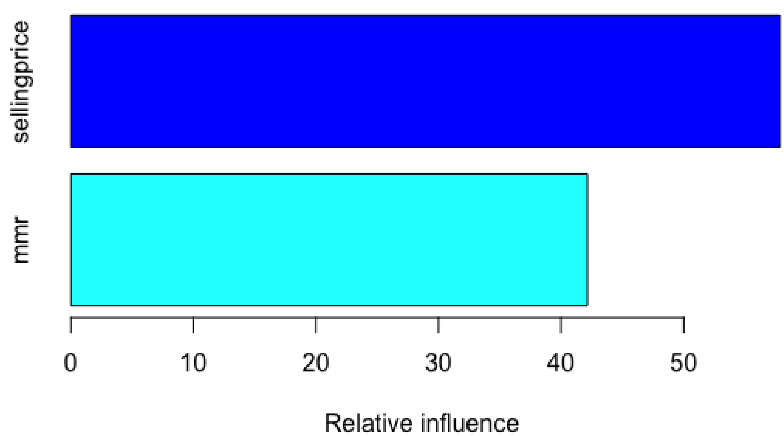
Accuracy of Algorithm 2 in Training set:
MSE : 0.01444225
R^2 : 0.7427869

Algorithm 2 type: Boosting

Graph that measures variable importance:



Upon using just the selling price and mmr to get the final result, the importance graph was:



Code from submission 1:

Code used from submission 1 which uses the training data :

Algorithm 1)
Training MSE - 0.2922281
Training R^2 - 0.6439893
Using random forest and regression trees to get the value.
I have compared the results of both, however, the regression tree gave a better value in the end. The random forest gave a close result and I have included it in my code in order to get the variable importance and use it as a comparison.
Importing the dataset and then making a few changes::

```
### importing dataset :
data <- read.csv("Desktop/training_small_final_pc.csv", header = TRUE)

library(tree)
library(randomForest)
set.seed(100)
train<- sample(1:nrow(data),500)
data$sellingprice<- log(data$sellingprice)

data.test = data[-train, ]
sp.test<- data[-train,"sellingprice"]

```

Implementing a random forest:

```
### Using random forest and bagging in order to see the MSE and R^2
### to compare the final value. RF helps gives a better variable importance.
rf.sp <- randomForest(sellingprice~.-mmr, data = data, subset = train,
                      importance = TRUE,na.action=na.omit,mtry=6)
yhat.rf <- predict(rf.sp,newdata = data.test)
yhat.rf <- na.roughfix(yhat.rf)
mse <- mean((yhat.rf - sp.test)^2)
mse
rss<- mean((yhat.rf - sp.test)^2)
tss <- mean((data$sellingprice-mean(sp.test))^2)
rsq <- 1 - rss/tss
rsq
importance(rf.sp)

varImpPlot(rf.sp)
```

Values from the random forest (as received):

```
> mse <- mean((yhat.rf - sp.test)^2)
> mse
[1] 0.3131778
> rss<- mean((yhat.rf - sp.test)^2)
> tss <- mean((data$sellingprice-mean(sp.test))^2)
> rsq <- 1 - rss/tss
> rsq
[1] 0.618467
```

The above is low accuracy

Thus, implementing a regression tree :

```
## To get a better MSE and R^2, using regression trees which is giving a better value
tree.sp <- tree(sellingprice~.-mmr, data = data , subset = train)
plot(tree.sp)
text(tree.sp, pretty = 0)
yhat.tree <-  predict(tree.sp, newdata = data.test)
prune.sp<- prune.tree(tree.sp)
mse <- mean((yhat.tree-sp.test)^2)
mse
rss<- mean((yhat.tree-sp.test)^2)
tss <- mean((data$sellingprice-mean(sp.test))^2)
rsq<- 1 - rss/tss
rsq
```

Value received from the regression tree:

```
> mse <- mean((yhat.tree-sp.test)^2)
> mse
[1] 0.2922281
> rss<- mean((yhat.tree-sp.test)^2)
> tss <- mean((data$sellingprice-mean(sp.test))^2)
> rsq<- 1 - rss/tss
> rsq
[1] 0.6439893
```

Giving us a higher level of accuracy than the random forest. Thus using the results from a regression tree.

Using the test data, the results achieved are :

```
> prune.sp<- prune.tree(tree.sp)
> mse <- mean((yhat.tree-sp.test)^2)
> mse
[1] 0.2898476
> rss<- mean((yhat.tree-sp.test)^2)
> tss <- mean((data$sellingprice-mean(sp.test))^2)
> rsq<- 1 - rss/tss
> rsq
[1] 0.6500216
```

Algorithm 2)

Training MSE - 0.01444225
Training R^2 - 0.7427869
Used boosting for this algorithm.

Importing the data and making a few changes :

```r
### importing dataset :
data <- read.csv("Desktop/training_small_final_pc.csv", header = TRUE)
library(gbm)
library(randomForest)
set.seed(100)
train<- sample(1:nrow(data),500)
data$sellingprice<- log(data$sellingprice)
data$mmr <- log(data$mmr)

### new column for the difference between the selling price and mmr
data$difference <- data$sellingprice - data$mmr
```

Using a random forest to check the importance of all the variables and how they correlate to the difference :

```r
### most important features when predicting the diff. :::
bag.diff <- randomForest(difference~., data = data, subset = train,
                         importance = TRUE,na.action=na.omit, ntree = 25
importance(bag.diff)
### we get that mmr and sleeing price have the highest influence .

```

```
> importance(bag.diff)
                 %IncMSE IncNodePurity
year          2.68564776     0.9811430
make          1.22490973     0.8714997
model         0.01278847     0.5412341
trim          3.16206564     1.0339857
body         -0.80993032     0.4965083
transmission  0.49400740     0.1072895
vin          -0.57339560     1.6599613
state        -1.96065397     0.7080283
condition     1.31753097     2.3651459
odometer      1.96040117     1.5877221
color        -1.77041153     0.5156934
interior     -0.86621311     0.3140183
seller        0.62614086     0.5916099
mmr           4.00485306     2.9087377
sellingprice  4.54952052     4.6211607
saledate      0.70689472     0.8054446
```

As we can see, and confirm, the most important variables in the difference are obviously the selling price and the mmr.

Using boosting to get a prediction:

```r
### using boosting in order to predict the difference best using the most
## influential features:

boost.diff <- gbm(difference~mmr+sellingprice, data = data[train,],
                  distribution = "gaussian",n.trees = 5000, interaction.depth = 4)
summary(boost.diff) ## sellingprice has a little higher influence than mmr.
yhat.boost<- predict(boost.diff, newdata= data[-train,], n.trees=5000)
diff.test<- data[-train,"difference"]
mse <- mean((yhat.boost-diff.test)^2)
mse
rss<- mean((yhat.boost-diff.test)^2)
tss <- mean(((data$difference-mean(diff.test))^2))
rsq <- 1 - rss/tss
rsq
```

Results ::

```
> summary(boost.diff) ## sellingprice has a little higher influence than mmr.
                      var  rel.inf
sellingprice sellingprice 57.87155
mmr                   mmr 42.12845
> yhat.boost<- predict(boost.diff, newdata= data[-train,], n.trees=5000)
> diff.test<- data[-train,"difference"]
> mse <- mean((yhat.boost-diff.test)^2)
> mse
[1] 0.01444225
> rss<- mean((yhat.boost-diff.test)^2)
> tss <- mean(((data$difference-mean(diff.test))^2))
> rsq <- 1 - rss/tss
> rsq
[1] 0.7427869
```

We can clearly see that the MSE is low for the MSE using boosting.

In the test data, the results achieved from this are :

```
> yhat.boost<- predict(boost.diff, newdata= data[-train,], n.trees=5000)
> diff.test<- data[-train,"difference"]
> mse <- mean((yhat.boost-diff.test)^2)
> mse
[1] 0.0111818
> rss<- mean((yhat.boost-diff.test)^2)
> tss <- mean(((data$difference-mean(diff.test))^2))
> rsq <- 1 - rss/tss
> rsq
[1] 0.8127233
```