GEORGETOWN UNIVERSITY

ANLY 512 - Statistical Learning for Analysis

Used Car Price Prediction

Group members: Guiming Xu, Kuiyu Zhu, Yuxuan Yao
Net ID: gx26, kz175, yy560

Introduction

Background

Over the last few decades, the used car market has demonstrated a significant growth in value contributing the larger share of the overall market value. Many of our friends in the US buy used cars, and the used car dealers in the US Market size of 2020 is about $119 bn, and the used cars market is keeping growing. Therefore, pricing and forecasting the price of used cars become particularly important both for dealers and consumers. Then we decide to build a model to predict used cars' prices.

Dataset

Our data is 20,000 used cars transaction records of USA (https://raw.githubusercontent.com/smeetvikani/Used-Car-Price-Predictor-Model/master/data/Step3_output_clean_df.csv)

Data Processing

We totally have 23 features in raw data and we select only 11 of them, which we are familiar with and also the basic information in some used-car websites. Because some variables, like vin code, engine code, are the unique ID of a car and which are not useful in the classification prediction. Our target label is price, and the price is close to the normal distribution but right skewed.

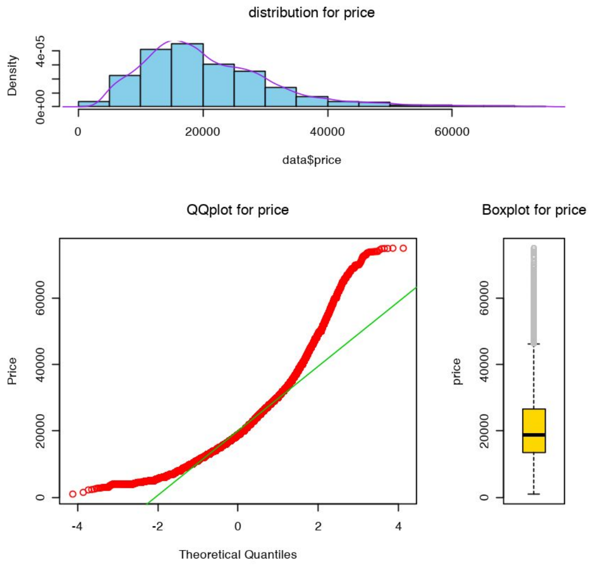


Figure 1: distribution of price

Also, for the categorical features, we transform them into numbers, such like transmission, body style. drive_type. And some variables are very large like milage, we scale it to avoid the situation that the coefficient may be too small in the linear regression part.

Moreover, here are the correlation plots of our selected 11 features:

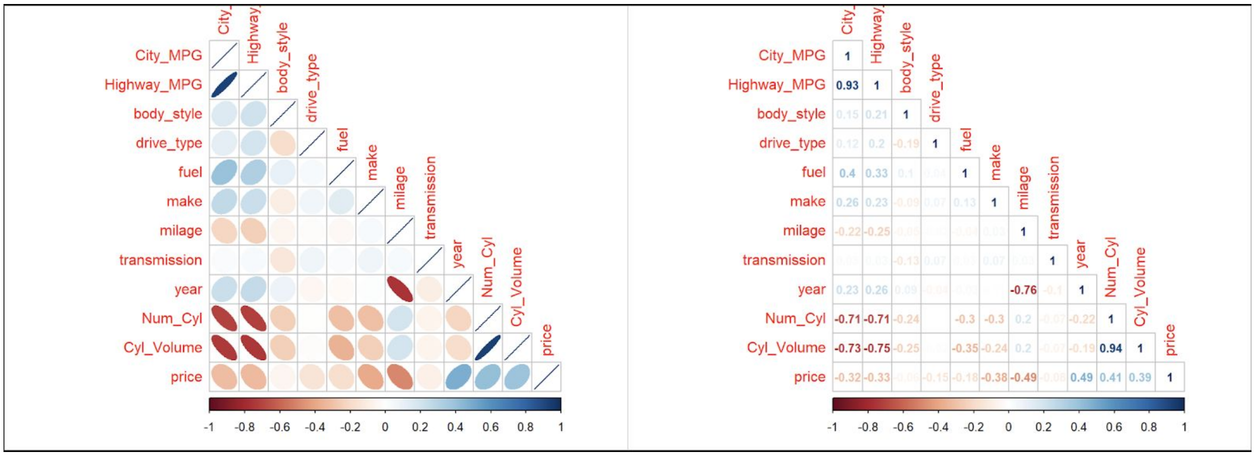
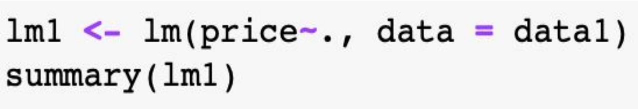


Figure 2: Correlation Plot

As we can see, the most positively related with price is year while the most negatively related is mileage. And there will be some interesting findings in the later features selection analysis.

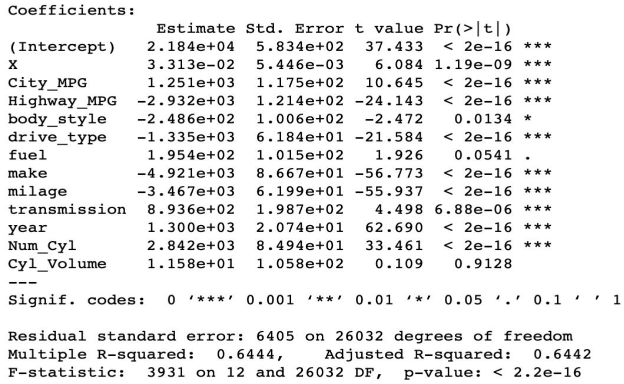
Regression and Model Selection

And then we start building models to make predictions, and our target variable is price. We mainly used the following models, including lasso, random forest, generalized additive model, and so on



```
lm1 <- lm(price~., data = data1)
summary(lm1)
```

Figure 3: Multiple Linear Regression



```
Coefficients:
               Estimate Std. Error t value Pr(>|t|)    
(Intercept)   2.184e+04  5.834e+02  37.433  < 2e-16 ***
X             3.313e-02  5.446e-03   6.084 1.19e-09 ***
City_MPG      1.251e+03  1.175e+02  10.645  < 2e-16 ***
Highway_MPG  -2.932e+03  1.214e+02 -24.143  < 2e-16 ***
body_style   -2.486e+02  1.006e+02  -2.472   0.0134 *  
drive_type   -1.335e+03  6.184e+01 -21.584  < 2e-16 ***
fuel          1.954e+02  1.015e+02   1.926   0.0541 .  
make         -4.921e+03  8.667e+01 -56.773  < 2e-16 ***
milage       -3.467e+03  6.199e+01 -55.937  < 2e-16 ***
transmission  8.936e+02  1.987e+02   4.498 6.88e-06 ***
year          1.300e+03  2.074e+01  62.690  < 2e-16 ***
Num_Cyl       2.842e+03  8.494e+01  33.461  < 2e-16 ***
Cyl_Volume    1.158e+01  1.058e+02   0.109   0.9128    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6405 on 26032 degrees of freedom
Multiple R-squared:  0.6444,	Adjusted R-squared:  0.6442 
F-statistic:  3931 on 12 and 26032 DF,  p-value: < 2.2e-16
```

Figure 4: Summary plot of multiple linear regression

The first model we used is multiple linear regression, and all predictors were selected for prediction. From the summary plot, we can find that all predictors are significant except Cylinder Volume.

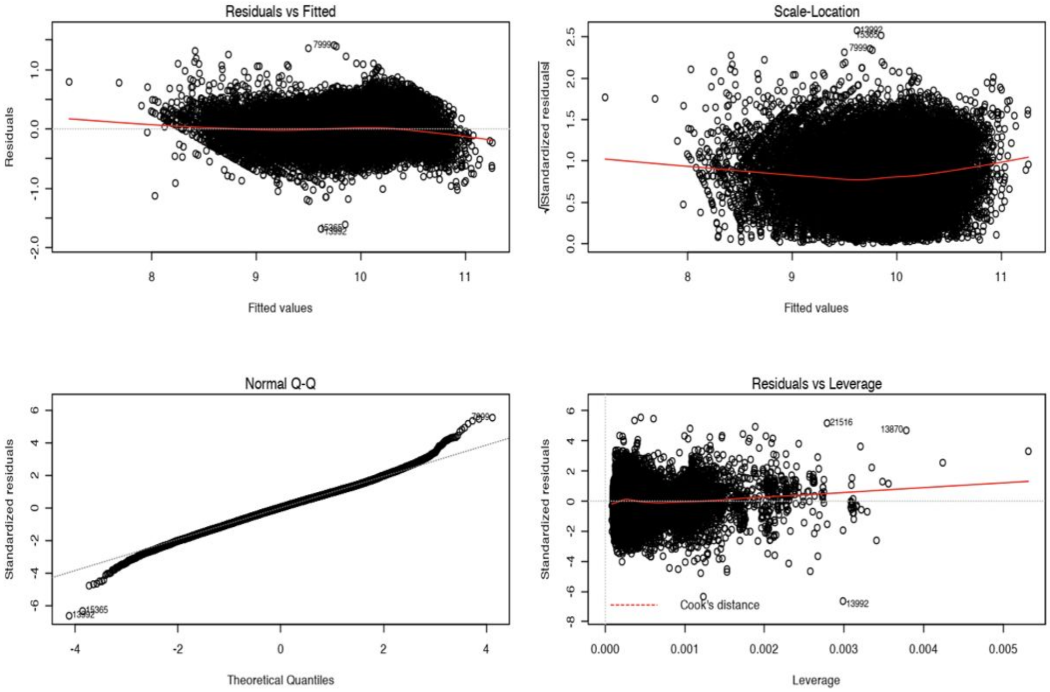
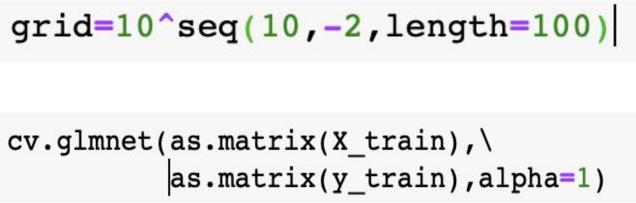


Figure 5: Residuals vs Fitted, etc

And then we plot the residuals versus predicted values, due to the smooth fit red line, we can find there is just a little pattern in the residuals, and the shape is not like a funnel, so there is no heteroscedasticity problem.

For the plot of the leverage point, there are just two or three black observations that are not unusual in terms of one of the predictor's values. Next, in order to predict more accurately, we respectively used leave one out and 10 fold CV to predict by using multiple linear regression, and to obtain RMSE.

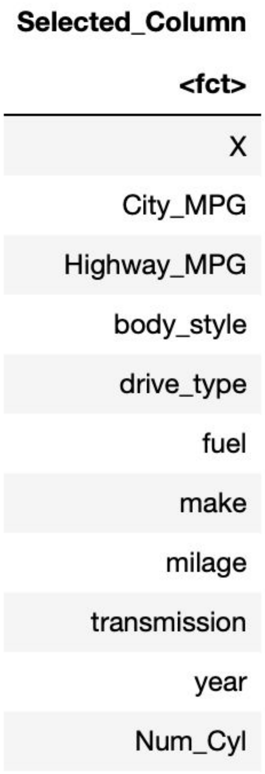


```
grid=10^seq(10,-2,length=100)
```

```
cv.glmnet(as.matrix(X_train),\
          as.matrix(y_train),alpha=1)
```

Figure 6: lambdas chosen

Then we choose to use a lasso to determine whether there are predictors that have low correlation with price. By using 100 different lambda, and using the k-fold CV to reduce error.



Selected_Column
<fct>
X
City_MPG
Highway_MPG
body_style
drive_type
fuel
make
milage
transmission
year
Num_Cyl

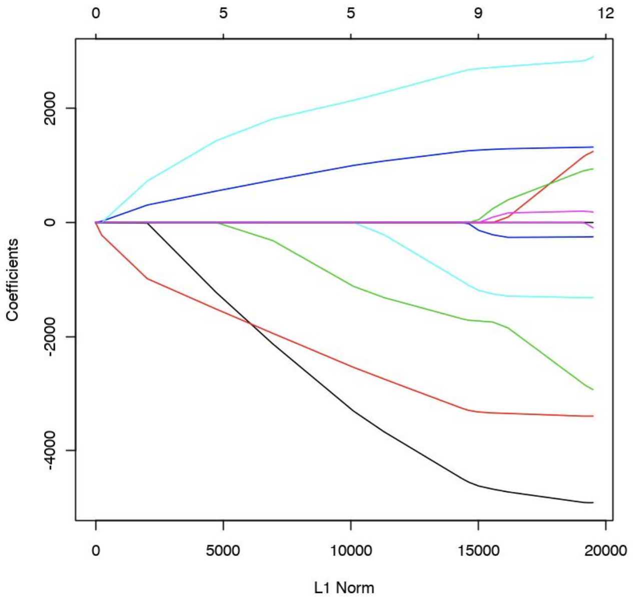
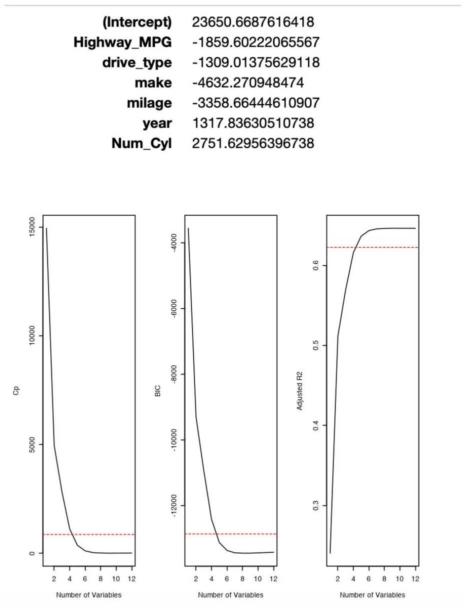


Figure 7: Selected Columns **Figure 8: Plot of coefficients**

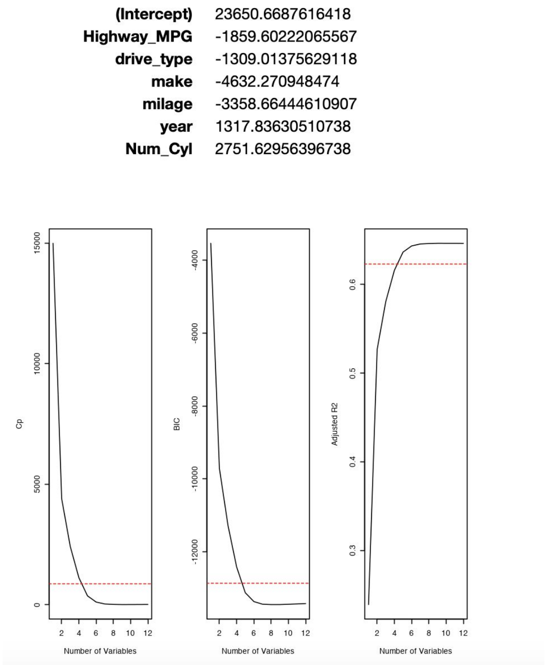
Finally we got the same conclusion as multiple linear regression. Cylinder Volume is not an important predictor, and finally we get the RMSE slightly worse than the linear model, and the RMSE is 6296.

And we also try Ridge method, and the RMSE is 6319.



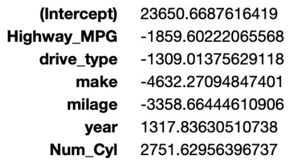
(Intercept)	23650.6687616418
Highway_MPG	-1859.60222065567
drive_type	-1309.01375629118
make	-4632.270948474
milage	-3358.66444610907
year	1317.83630510738
Num_Cyl	2751.62956396738

Figure 9: Forward Stepwise Selection



(Intercept)	23650.6687616418
Highway_MPG	-1859.60222065567
drive_type	-1309.01375629118
make	-4632.270948474
milage	-3358.66444610907
year	1317.83630510738
Num_Cyl	2751.62956396738

Figure 10: Backward Stepwise Selection



(Intercept)	23650.6687616419
Highway_MPG	-1859.60222065568
drive_type	-1309.01375629118
make	-4632.27094847401
milage	-3358.66444610906
year	1317.83630510738
Num_Cyl	2751.62956396737

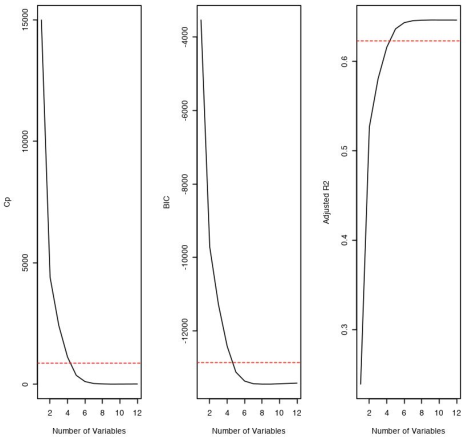


Figure 11: Best Subset Selection

Next, in order to further analyze the number of predictors, we used forward stepwise and backward stepwise. We can find the coefficients of each method are really the same. By observing AIC, BIC and adjusted r-square, we decided to use 6 variables for the GAM analysis.

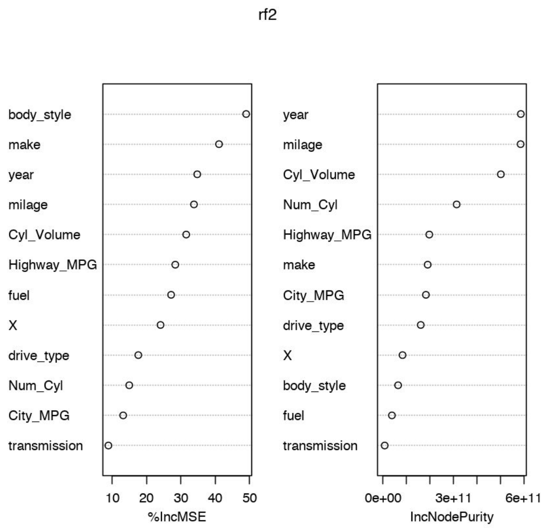


Figure 12: Random Forest Feature Importance Plot)

For random forest, all predictors were selected to predict, and this model has the lowest RMSE, and we also find an interesting feature that transmission is another unimportant feature by plotting a feature importance graph, and the RMSE for random forest is 3345.

We fit five different GAMs, and we just show the three graphs of them.

1. gam1 = gam(price~s(Num_Cyl,2)+s(Highway_MPG,2)+s(year,4)+
   s(milage,4)+make+drive_type+
   s(Cyl_Volume,2),data=subset_train)

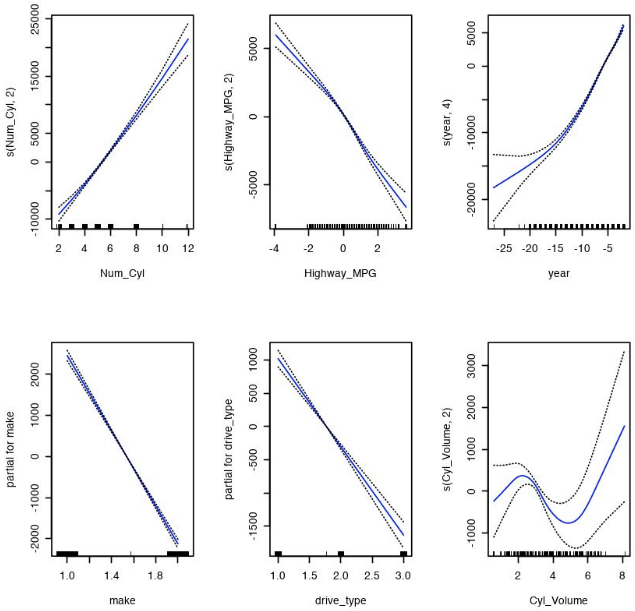


Figure 13: gam1

2. gam2 = gam(price~s(Num_Cyl,3)+s(Highway_MPG,3)+s(year,2)+

   s(milage,2)+make+drive_type+

   lo(Cyl_Volume,span=0.7),data=subset_train)

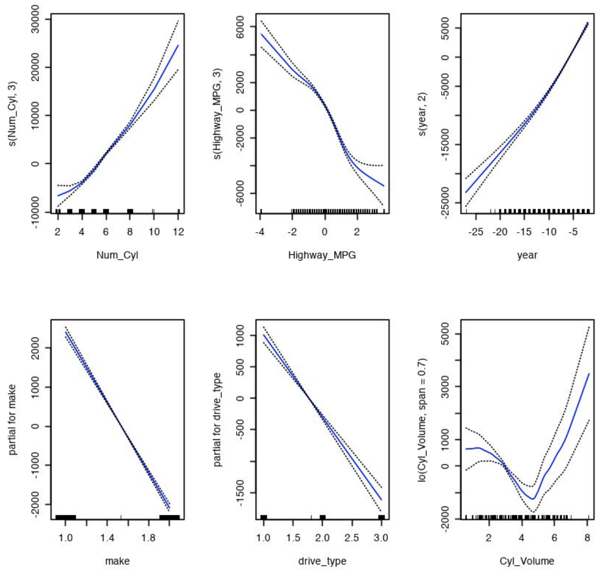


Figure 14: gam2

3. gam3 = gam(price~s(Num_Cyl)+s(Highway_MPG,4)+s(year,4)+ lo(milage,span=0.5)+make+drive_type+ s(Cyl_Volume,3),data=subset_train)

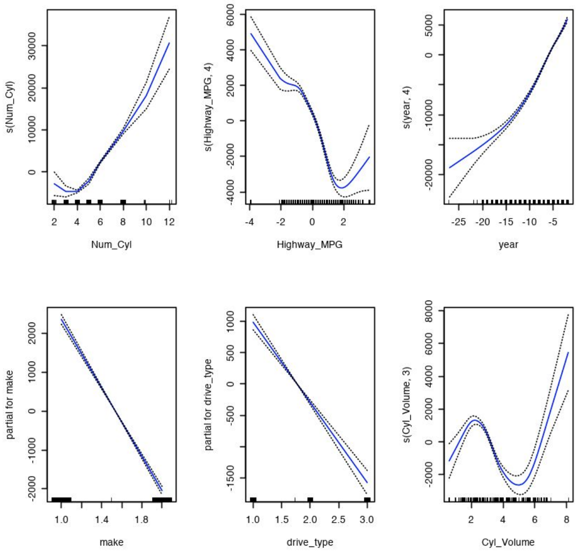
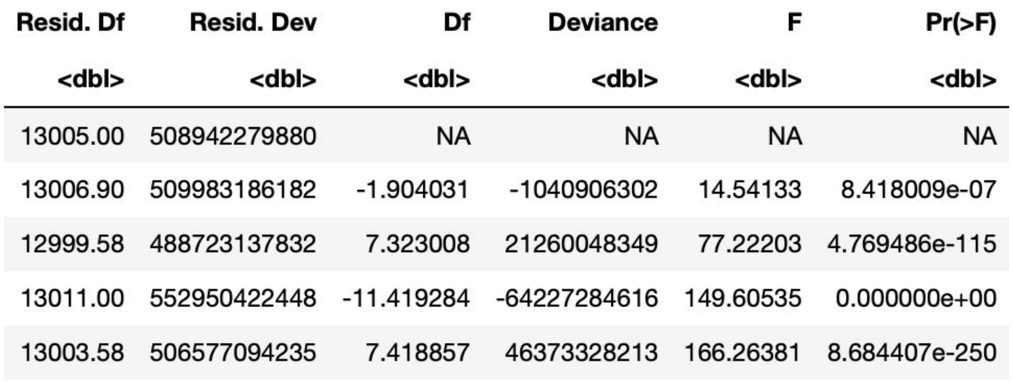


Figure 15: gam3

In the end, by using the six variables selected in the previous step, we used several different combinations of local regression and smooth spline, and used anova to select the best model for prediction. The final RMSE is 6222.



Resid. Df	Resid. Dev	Df	Deviance	F	Pr(>F)
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
13005.00	508942279880	NA	NA	NA	NA
13006.90	509983186182	-1.904031	-1040906302	14.54133	8.418009e-07
12999.58	488723137832	7.323008	21260048349	77.22203	4.769486e-115
13011.00	552950422448	-11.419284	-64227284616	149.60535	0.000000e+00
13003.58	506577094235	7.418857	46373328213	166.26381	8.684407e-250

Figure 16: ANOVA table

Therefore, in the regression part, random forest is the best model with lowest RMSE value.

Clustering and Classification

For the part of clustering, we applied kmeans clustering method with two dimensional Principal Component Analysis. Here are two plots of different R libraries.

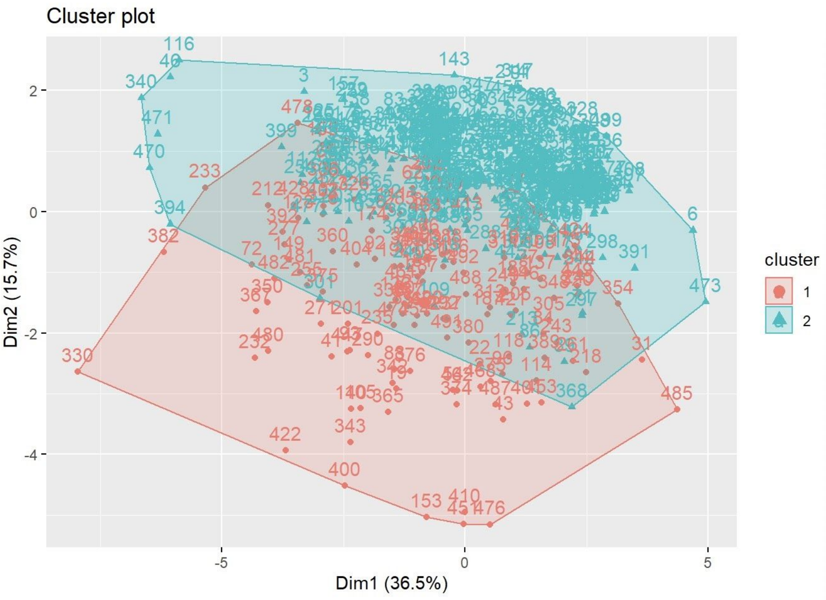


Figure 17: Kmeans & PCA2 via "factoextra"

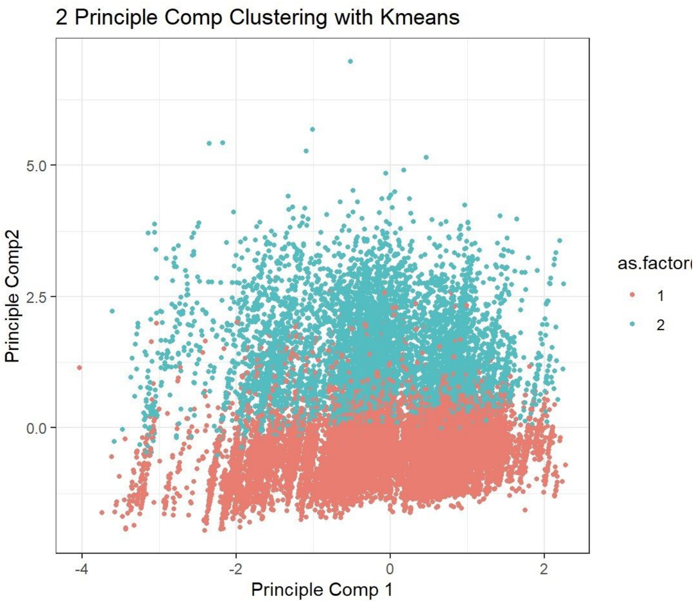
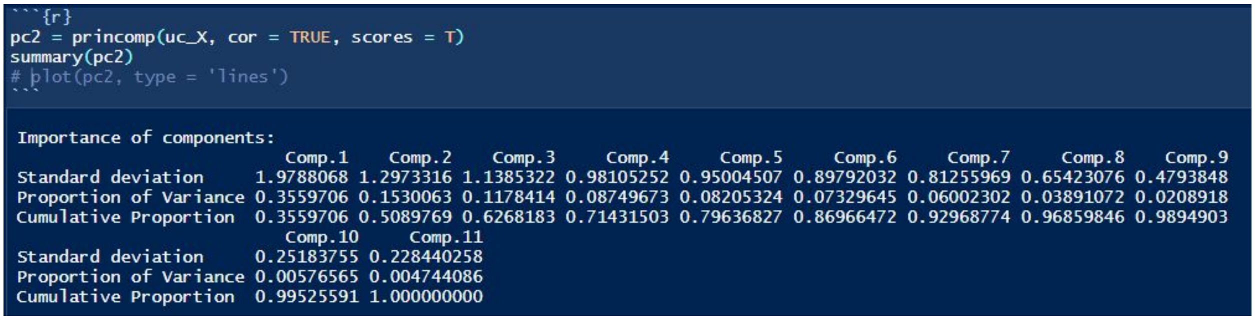


Figure 18: Kmeans & PCA2 via "psych"

The results of these two plots are basically telling the same idea. There are two clusters. One is located at the top of the graph, while another cluster is located at the downside of the graph. This makes me continue exploring the principle components of our design matrix.

So I printed the information of each component out.



~~~
```{r}
pc2 = princomp(uc_X, cor = TRUE, scores = T)
summary(pc2)
# plot(pc2, type = 'lines')
```
~~~

```
Importance of components:
                          Comp.1    Comp.2    Comp.3     Comp.4     Comp.5     Comp.6     Comp.7     Comp.8    Comp.9
Standard deviation     1.9788068 1.2973316 1.1385322 0.98105252 0.95004507 0.89792032 0.81255969 0.65423076 0.4793848
Proportion of Variance 0.3559706 0.1530063 0.1178414 0.08749673 0.08205324 0.07329645 0.06002302 0.03891072 0.0208918
Cumulative Proportion  0.3559706 0.5089769 0.6268183 0.71431503 0.79636827 0.86966472 0.92968774 0.96859846 0.9894903
                          Comp.10     Comp.11
Standard deviation     0.25183755 0.228440258
Proportion of Variance 0.00576565 0.004744086
Cumulative Proportion  0.99525591 1.000000000
```

Figure 19: Components' Information

Here we see that 9 or 10 components are needed to bring the cumulative proportion of variance close to 1. In addition You can get a similar idea from the plot as well.

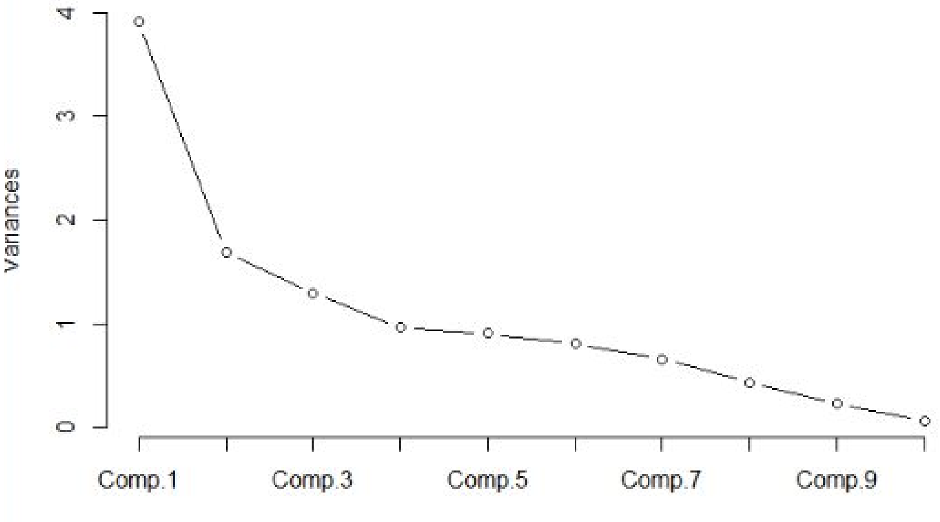
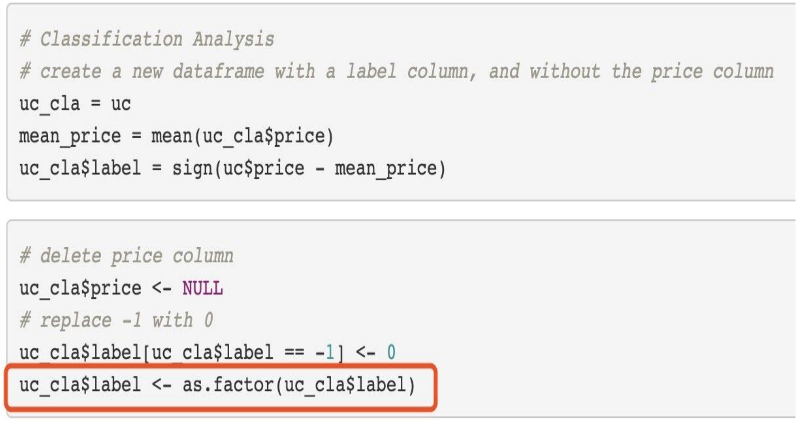


Figure 20: N_components Variance Plot

Next, we built some binary classification models mainly focusing on accuracy. We define and create the label by our own, If the price is higher than the mean price, marked as 1, otherwise 0.

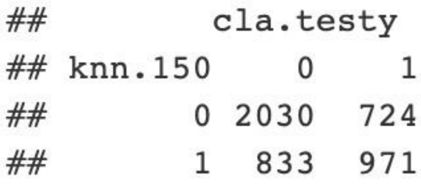


```
# Classification Analysis
# create a new dataframe with a label column, and without the price column
uc_cla = uc
mean_price = mean(uc_cla$price)
uc_cla$label = sign(uc$price - mean_price)
```

```
# delete price column
uc_cla$price <- NULL
# replace -1 with 0
uc_cla$label[uc_cla$label == -1] <- 0
uc_cla$label <- as.factor(uc_cla$label)
```

Figure 21: Binary Labeling

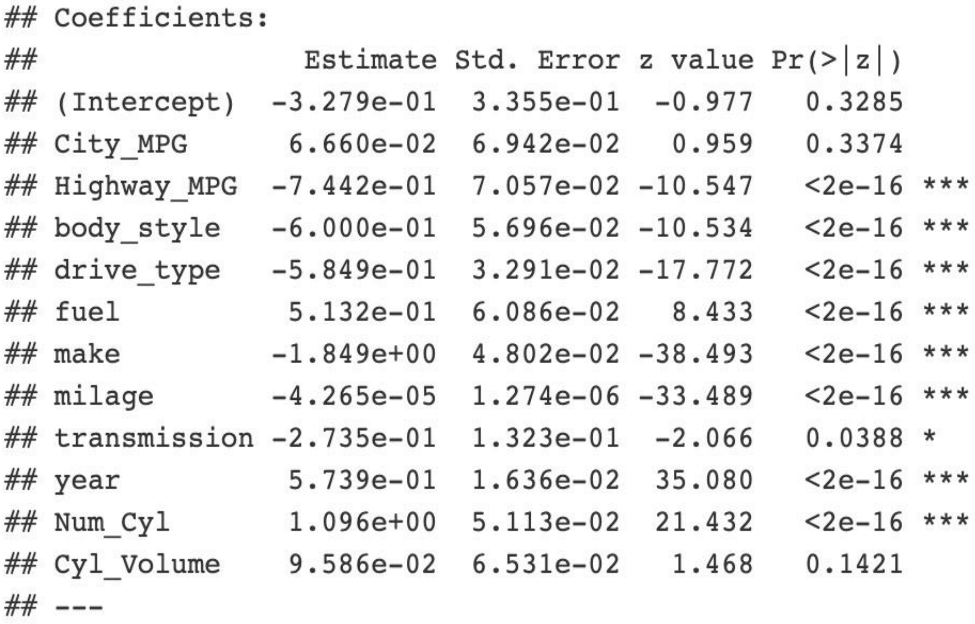
For the KNN method, at the very first we picked k = 150, which is close to sqrt of the number of observations, the accuracy is pretty low, so I tried some other small k values. It proved that when k is large it might be under-fitting, and overfitting when k is small. Note that when k=150, the accuracy is about 0.658, and the following is the confusion matrix.



```
##          cla.testy
## knn.150     0    1
##         0 2030  724
##         1  833  971
```

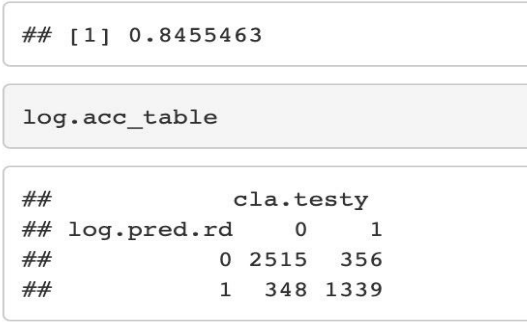
Figure 22: Confusion Matrix of KNN method (k=150)

Since, the knn method needs a lot of memory, it takes a long time to try many different ks. So we moved on, and tried logistic regression, the accuracy is about 0.846 and two features showing in the model summary are not significant respectively.



```
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -3.279e-01  3.355e-01  -0.977   0.3285
## City_MPG      6.660e-02  6.942e-02   0.959   0.3374
## Highway_MPG  -7.442e-01  7.057e-02 -10.547   <2e-16 ***
## body_style   -6.000e-01  5.696e-02 -10.534   <2e-16 ***
## drive_type   -5.849e-01  3.291e-02 -17.772   <2e-16 ***
## fuel          5.132e-01  6.086e-02   8.433   <2e-16 ***
## make         -1.849e+00  4.802e-02 -38.493   <2e-16 ***
## milage       -4.265e-05  1.274e-06 -33.489   <2e-16 ***
## transmission -2.735e-01  1.323e-01  -2.066   0.0388 *
## year          5.739e-01  1.636e-02  35.080   <2e-16 ***
## Num_Cyl       1.096e+00  5.113e-02  21.432   <2e-16 ***
## Cyl_Volume    9.586e-02  6.531e-02   1.468   0.1421
## ---
```

Figure 23: Logistic Regression Model Summary



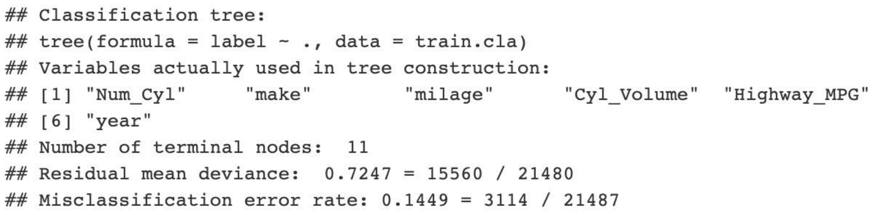
```
## [1] 0.8455463
```

```
log.acc_table
```

```
##             cla.testy
## log.pred.rd    0    1
##           0 2515  356
##           1  348 1339
```

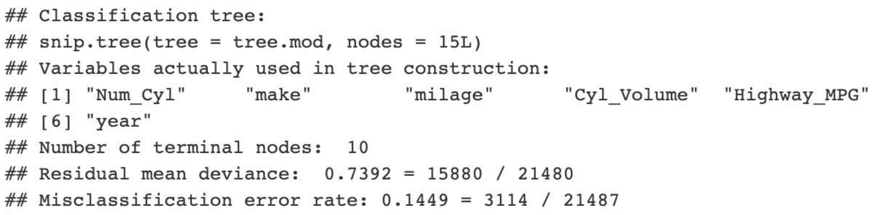
Figure 24: Logistic Regression Accuracy and Confusion Matrix

Then , we did two decision trees, pruned and unpruned, the results are basically the same since both of them only use 6 features.



```
## Classification tree:
## tree(formula = label ~ ., data = train.cla)
## Variables actually used in tree construction:
## [1] "Num_Cyl"     "make"        "milage"      "Cyl_Volume"  "Highway_MPG"
## [6] "year"
## Number of terminal nodes:  11
## Residual mean deviance:  0.7247 = 15560 / 21480
## Misclassification error rate: 0.1449 = 3114 / 21487
```

Figure 25: Unpruned Decision Tree Model Summary



```
## Classification tree:
## snip.tree(tree = tree.mod, nodes = 15L)
## Variables actually used in tree construction:
## [1] "Num_Cyl"     "make"        "milage"      "Cyl_Volume"  "Highway_MPG"
## [6] "year"
## Number of terminal nodes:  10
## Residual mean deviance:  0.7392 = 15880 / 21480
## Misclassification error rate: 0.1449 = 3114 / 21487
```

Figure 26: Pruned Decision Tree Model Summary

From the plot below, it tells us all features should be included to get a smallest error, but we won't do that for overfitting concern. So for our decision tree models, 6 may be the best number of features (tree size) included in the model.

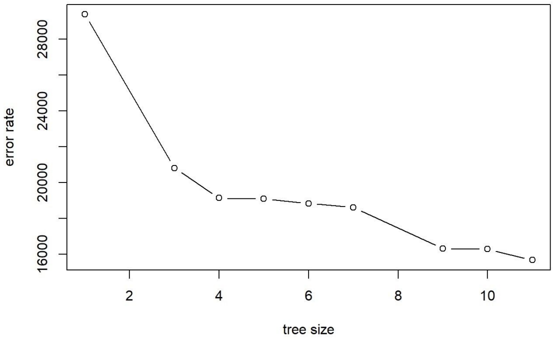


Figure 27: Tree Size vs Error Rate Plot

Last but not least, we also implemented classification random forest, and bagging models. Random forest with 100 trees has the highest accuracy among all these models, while knn may not be the right method as you can see in the following classification accuracy table.

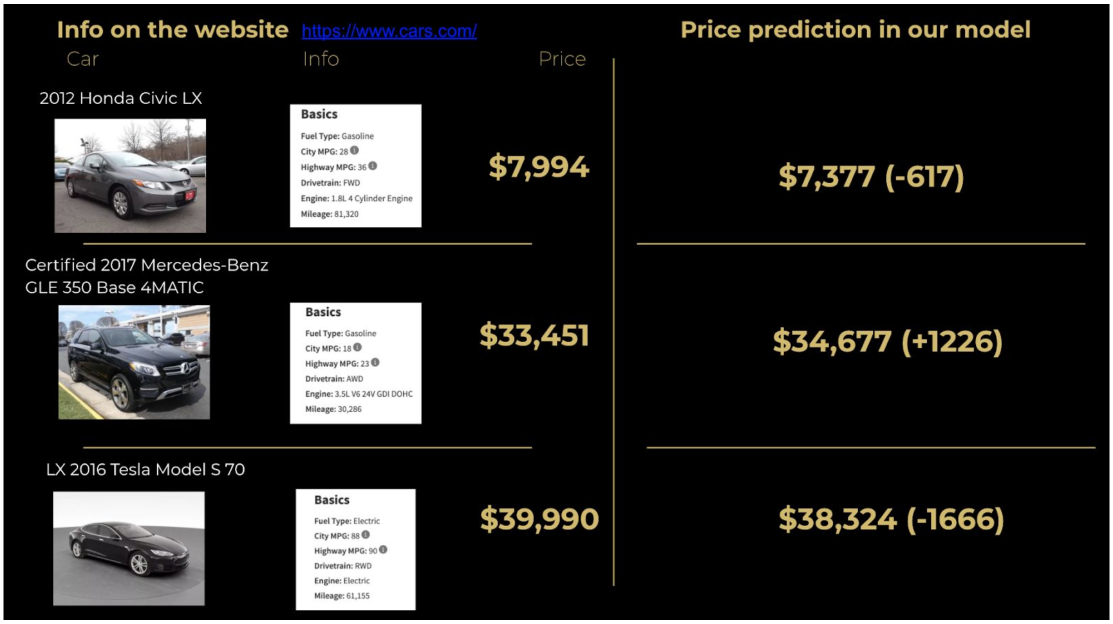
Models	Classification Accuracy
Best Knn	0.768
Logistic Regression	0.846
Decision Tree	0.849
Random Forest (100)	0.926
Bagging (100)	0.913

Table 1: Classification Models and Accuracy

Conclusion

There are 2 interesting findings in doing this prediction, cylinder volume is an important factor when we buy a car in real life, but in the regression part, the modeling results and the LASSO show that it is actually not significant in price prediction. And the transmission type is also a feature we consider about buying a car, but in the random forest variable importance plot, it is the last one.

As for the conclusion, our best model is random forest, so we test our model by entering the real data on the website, cars.com. Here are the results:



Info on the website https://www.cars.com/

Price prediction in our model

Car	Info	Price	Price prediction in our model
2012 Honda Civic LX	**Basics** **Fuel Type:** Gasoline **City MPG:** 28 **Highway MPG:** 36 **Drivetrain:** FWD **Engine:** 1.8L 4 Cylinder Engine **Mileage:** 81,320	$7,994	$7,377 (-617)
Certified 2017 Mercedes-Benz GLE 350 Base 4MATIC	**Basics** **Fuel Type:** Gasoline **City MPG:** 18 **Highway MPG:** 23 **Drivetrain:** AWD **Engine:** 3.5L V6 24V GDI DOHC **Mileage:** 30,286	$33,451	$34,677 (+1226)
LX 2016 Tesla Model S 70	**Basics** **Fuel Type:** Electric **City MPG:** 88 **Highway MPG:** 90 **Drivetrain:** RWD **Engine:** Electric **Mileage:** 61,155	$39,990	$38,324 (-1666)

We enter the basic information from the web and the left prices are the real price, the right are the price our model predicts and the difference between them. In general, we achieve our goal, which is predicting the used cars’ price by machine learning modeling. We are satisfied with our model performance.