Used Car Price Prediction Case Report

Student Names: Jiachen Liu

Executive Summary:

Pricing model has been widely used in many areas including website and insurance company. For example, the used car retailer CarMax uses pricing model to determine price of used cars. In this project, we use dataset from Kaggle that contains used car selling information and try to build pricing model. In order to do so, experiment with different kinds of machine learning algorithms including linear regression, lasso regression, ridge regression, elastic net, KNN, and XGboost, which showcase linear model, KNN and gradient boosted tree model. Then we use these models on test set and evaluate the performance by RMSE, R^2 and MAE. The goal of this project is to predict price of a used car based on the giving information.

I. Background and Introduction

1. Background and Problem

Nowadays, people choose to buy and sell their used cars. Everyone wants their car to be sold at an ideal price. However, how to determine the ideal price is not easy. Therefore, building a pricing model based on historical dataset is the best way to solve this problem. Pricing model has been widely used in many fields. For example, many used car websites such as CarMax use pricing model to determine the price of the used car. Also, the pricing model can be applied in other industries such as insurance company to determine the price of insurance. In this project, we will implement a pricing model to forecast the price of the car in used car market based on giving information.

2. Goal of project

The pricing model has been widely used in many fields such as used car retailer and insurance company. For the goal of our project, we will implement several regression models to predict the price of a used car.

3. The possible solution

First of all, we perform some exploratory data analysis to check the balance of the data and find possible correlations by visualization methods. After the EDA process, we perform feature engineering on the whole dataset. For example, we remove unnecessary columns and check missing values in the dataset. For the categorical data in the dataset, we choose the One-Hot Encoding method to encode these categorical data. Also, we will reduce the dimension of the data by PCA and normalize data. Then, we will select the appropriate model for this regression problem. For the selection of the model, we will not only do some linear models such as Linear Regression, Lasso Regression, Ridge Regression, Elastic Net, but also we try the KNN model and tree model such as XGBoost. We will evaluate the result by RMSE, R^2, MAE and plotting the scatter plot of prediction value and true value.

II. Workflow of the Project

In this case study, we have five main steps as the following chart shows, which are Input Data, Exploratory Data Analysis, Feature Engineering, Model Implementation, and Model Evaluation.

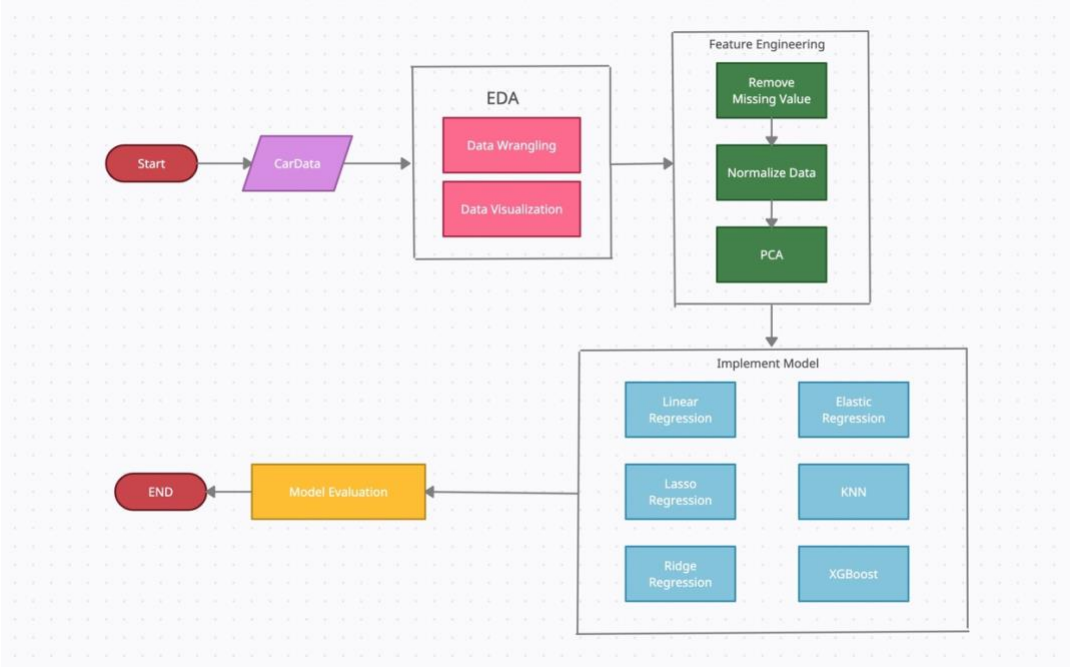


Figure_1: Workflow of the Project

III. EDA (Data Exploration and Visualization)

1. Data Exploration

a. Data Description

The dataset is an open dataset derived from Kaggle (1). The whole dataset contains thirteen CSV files, which provides details for used cars in different brands. The whole dataset contains almost 100,000 observations and 10 features. For the features of the dataset, there are 4 categorical features and 6 numerical features. Due to the computation cost, we only subset the data, whose brand is BMW. In BMW dataset, we have 10,781 observations and 10 features. We can see detailed information in the following plot.

```
    model               year         price          transmission         mileage
 Length:10781       Min.   :1996   Min.   :  1200   Length:10781       Min.   :     1
 Class :character   1st Qu.:2016   1st Qu.: 14950   Class :character   1st Qu.:  5529
 Mode  :character   Median :2017   Median : 20462   Mode  :character   Median : 18347
                    Mean   :2017   Mean   : 22733                      Mean   : 25497
                    3rd Qu.:2019   3rd Qu.: 27940                      3rd Qu.: 38206
                    Max.   :2020   Max.   :123456                      Max.   :214000
   fuelType             tax             mpg          engineSize       brand
 Length:10781       Min.   :  0.0   Min.   :  5.5   Min.   :0.000   Length:10781
 Class :character   1st Qu.:135.0   1st Qu.: 45.6   1st Qu.:2.000   Class :character
 Mode  :character   Median :145.0   Median : 53.3   Median :2.000   Mode  :character
                    Mean   :131.7   Mean   : 56.4   Mean   :2.168
                    3rd Qu.:145.0   3rd Qu.: 62.8   3rd Qu.:2.000
                    Max.   :580.0   Max.   :470.8   Max.   :6.600
```

Figure_2: Summary of BMW Dataset

b. Feature Description

As the table shows, we have 10 variables in our dataset, which are model, year, price, transmission, mileage, fuelType, tax, mpg, engineSize and brand. In our case, our input variable is model, year, transmission, mileage, fuelType, tax, mpg, engineSize. We remove unnecessary column “brand” because all brand in our sub dataset is “BMW”. Our target value in our project is price.

Feature Name	Description
model	The car model of the brand
year	Car Registration year
price	Car Price
transmission	Type of gearbox
mileage	Distance that used by car
fuelType	Engine fuel
tax	Road tax
mpg	Miles per gallon
engineSize	Size in literes
brand	Brand of car

Tabel_1: Feature Description

c. Missing value in dataset

We check the missing value of the dataset. Fortunately, we do not have any missing values in each column, which our dataset is very clean.

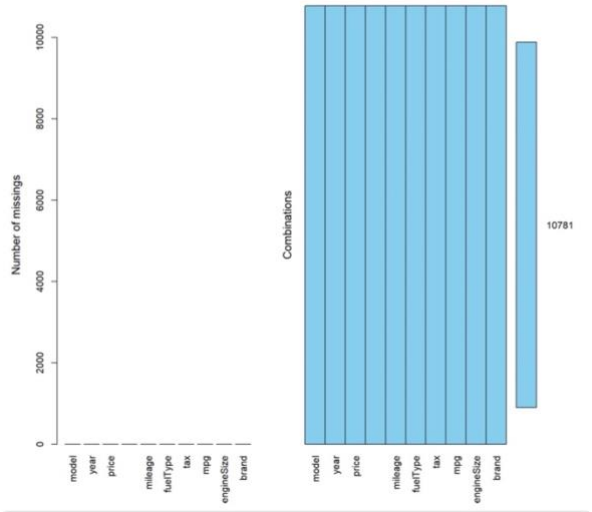


Figure_3: Missing Value in Each Column

2. Data Visualization

In order to explore more about BMW dataset, we perform some data visualizations to find relationship between feature variable and target variable.

a. Box plot of BMW Price in each year

From the Box plot as showing below, we can see that the mean price of BMW increase with the increase of the year.

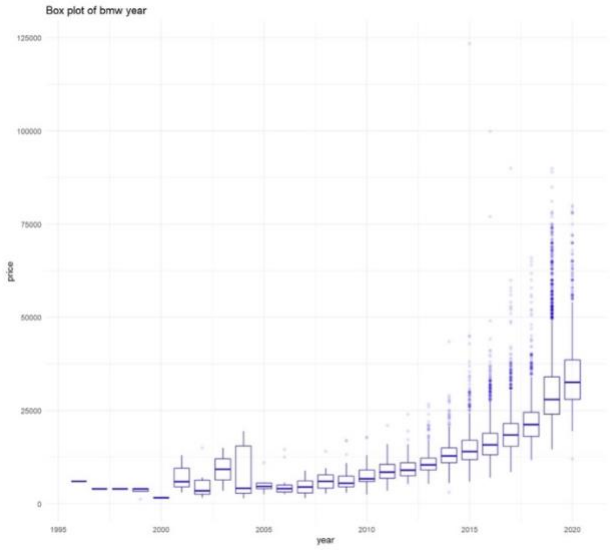


Figure_4: Box Plot of BMW Price in each year

b. Boxplot of BMW price in different Transmission

From the plot, we can see that Automatic and Semi-Auto have higher mean prices than manual transmission.

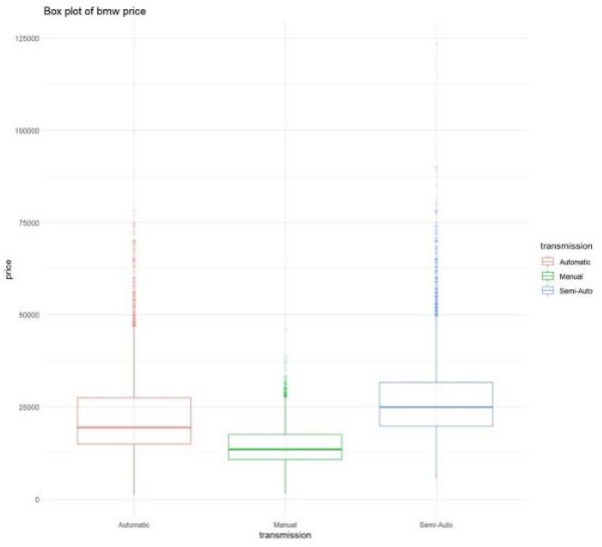


Figure_5: Box Plot of BMW Price in different transmission

c. *Scatter Plot of millage and price in different transmission*

From the plot, we can see that the price decrease with the increase of millage in different transmission.

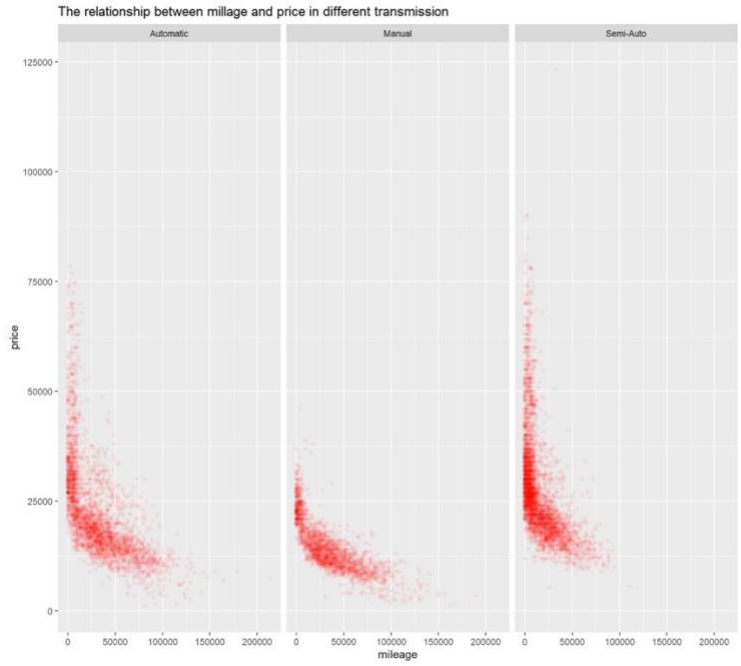


Figure_6: Scatter Plot of millage and price in different transmission

d. *Scatter Plot of millage and price in recent 5 years*

From the plot, we can see that there is linear relationship in 2015, 2016 and 2017 and there seems no linear relationship in 2020 and 2019. The reason could be the car in 2020 and 2019 can be deal as the "New" car, which millage is very low.

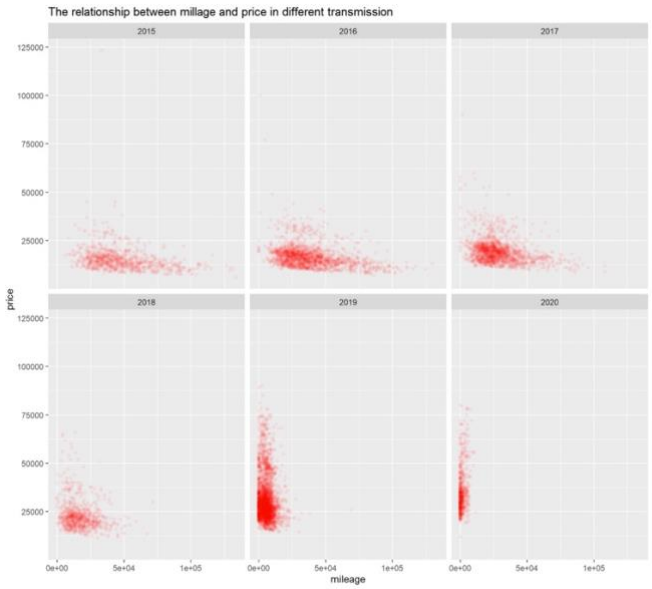


Figure_7: Scatter Plot of millage and price in recent 5 years

IV. Feature Engineering (Data preparation and Preprocessing)

After we remove the unnecessary columns, and we select all features as our input variable. Since we have many categorical data in our dataset, we need to encode these categorical data. In this project, we use the One-Hot encoding method to encode our categorical data. Since some value in some features such as year, mileage is much larger than other columns, we need to normalize these values and also prepare for principal components analysis in the next steps. After encoding our categorical data and data normalization, the dimension of our dataset is too large.

Therefore, we use principal components analysis to reduce the demission of the dataset. From the plot below, we plot the component number and engine value of principal components. From the result, we can see that the optimized number of components is 31. Thus, we choose the first 31 components to implement our model.

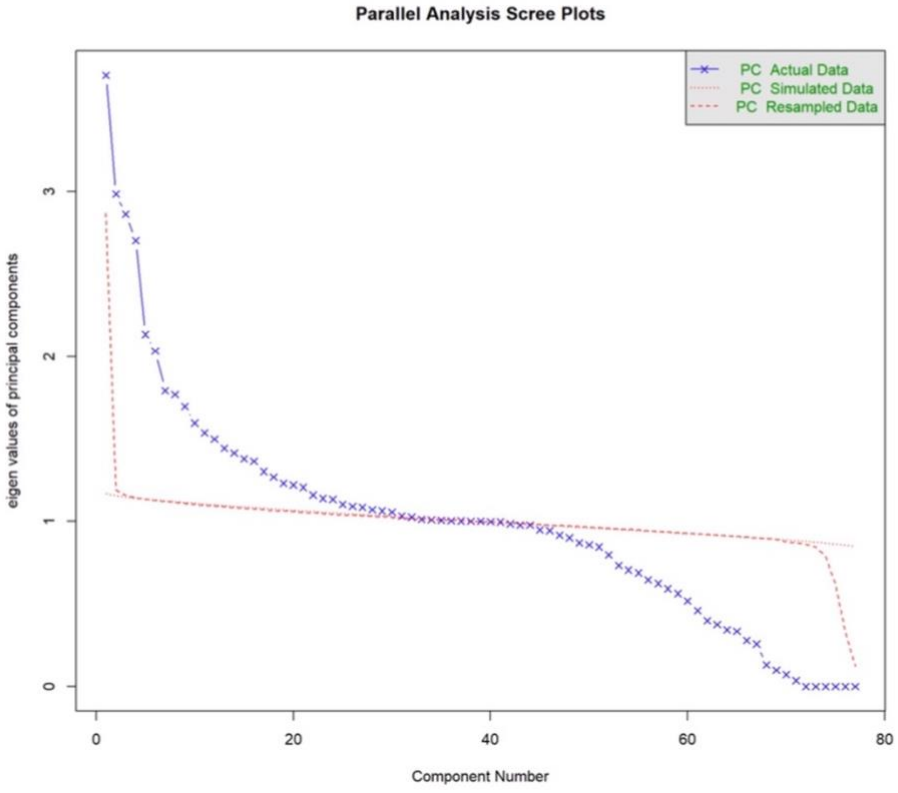


Figure_8: Principal Components Analysis

V. Data Mining Techniques and Implementation

1. Input Variable

Before implementing the model, we need to choose the input variable of the model. In this project, our input variable is the first 31principle components, which are selected by principal components analysis.

2. Split Train Dataset and Test Dataset

   We split the dataset into 80% train dataset and 20% test dataset.

3. Model Selection

   Since our problem is to predict the price based on the given value, we use six regression model in our project.

   The Regression Models are:

   a. Linear Regression
   b. Lasso Regression
   c. Ridge Regression
   d. Elastic Net
   e. KNN Regression
   f. XGBoost

4. Model implementation

   a. *Linear Regression*

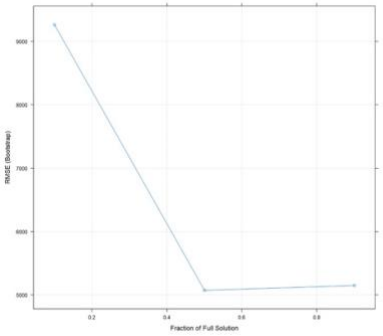
      After we implement linear regression on train data set, we got the coefficient for each principal component. And the R^2 for the Linear Regression model on train dataset is 89.7%, which looks good.

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 22737.80      39.77 571.682  < 2e-16 ***
## PC1          2124.82      10.83 196.247  < 2e-16 ***
## PC2          1766.52      13.43 131.515  < 2e-16 ***
## PC3           868.48      14.30  60.747  < 2e-16 ***
## PC4            55.16      14.62   3.772 0.000163 ***
## PC5          -676.06      18.69 -36.164  < 2e-16 ***
## PC6          -953.21      19.79 -48.164  < 2e-16 ***
## PC7          -648.42      21.97 -29.510  < 2e-16 ***
## PC8           -34.48      20.54  -1.678 0.093343 .
## PC9          -189.38      23.40  -8.093 6.62e-16 ***
## PC10         -489.52      40.16 -12.189  < 2e-16 ***
## PC11         1382.27      27.53  50.219  < 2e-16 ***
## PC12          319.12      30.81  10.359  < 2e-16 ***
## PC13          -50.47      26.20  -1.926 0.054140 .
## PC14          103.12      29.97   3.440 0.000584 ***
## PC15          900.41      29.68  30.341  < 2e-16 ***
## PC16          600.78      29.73  20.206  < 2e-16 ***
## PC17          226.99      30.58   7.423 1.26e-13 ***
## PC18          619.40      32.29  19.185  < 2e-16 ***
## PC19          544.67      32.32  16.850  < 2e-16 ***
## PC20          365.94      34.76  10.529  < 2e-16 ***
## PC21         1293.34      33.67  38.409  < 2e-16 ***
## PC22         -145.73      33.92  -4.297 1.75e-05 ***
## PC23           42.33      37.66   1.124 0.261097
## PC24          143.17      35.54   4.028 5.66e-05 ***
## PC25        -1039.04      36.20 -28.704  < 2e-16 ***
## PC26          686.64      36.51  18.805  < 2e-16 ***
## PC27          -11.53      36.66  -0.315 0.753064
## PC28          180.31      36.96   4.879 1.09e-06 ***
## PC29          -95.96      37.13  -2.584 0.009773 **
## PC30          569.36      37.91  15.019  < 2e-16 ***
## PC31         -330.95      37.44  -8.839  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3649 on 8390 degrees of freedom
## Multiple R-squared:  0.897,  Adjusted R-squared:  0.8966
## F-statistic:  2356 on 31 and 8390 DF,  p-value: < 2.2e-16
```

Figure_9: Linear Regression on Train Dataset

b. *Lasso Regression*

After we implement lasso regression on train data set, we got the plot of the fraction of solution and RMSE. From the plot, we can see the most optimized fraction is 0.5, and the RMSE, R^2 and MAE are 5073.275, 82.36% and 3533.407.

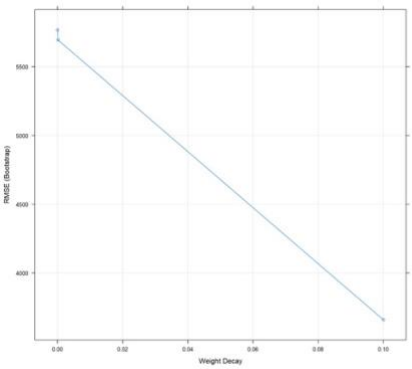



```
## The lasso
##
## 8422 samples
##   31 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 8422, 8422, 8422, 8422, 8422, 8422, ...
## Resampling results across tuning parameters:
##
##   fraction  RMSE      Rsquared   MAE
##   0.1       9266.520  0.5580142  6736.382
##   0.5       5073.275  0.8236085  3533.407
##   0.9       5151.493  0.8585858  2688.957
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was fraction = 0.5.
```

Figure_10: Lasso Regression on Train Dataset

c. *Ridge Regression*

After we implement ridge regression on train data set, we got plot of the weight decay of solution and RMSE. From the plot, we can see the most optimized lambda is 0.1, and the RMSE, R2 and MAE are 3659.588, 89.52% and 2613.91.

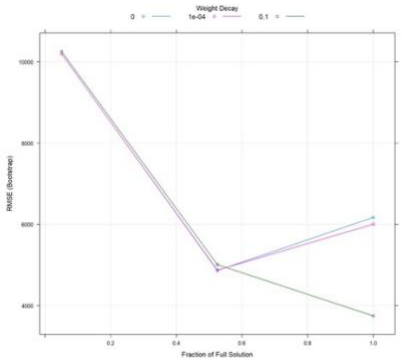



```
## Ridge Regression
##
## 8422 samples
##   31 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 8422, 8422, 8422, 8422, 8422, 8422, ...
## Resampling results across tuning parameters:
##
##   lambda  RMSE      Rsquared   MAE
##   0e+00   5769.355  0.8297025  2678.747
##   1e-04   5695.735  0.8303368  2676.700
##   1e-01   3659.588  0.8951695  2613.910
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was lambda = 0.1.
```

Figure_11: Ridge Regression on Train Dataset

d. *Elastic Net*

After we implement Elastic Net on train data set, we got plot of both weight decay, fraction of solution and RMSE. From the plot, we can see the most optimized fraction is 1 and lambda is 0.1, and the RMSE, R2 and MAE are 3749.853, 89.07% and 2629.293.

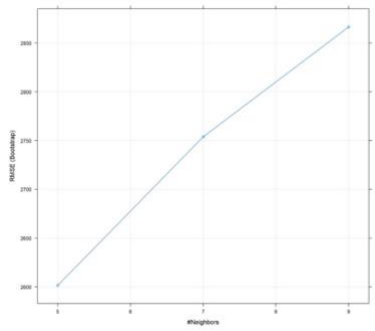



```
## Elasticnet
##
## 8422 samples
##   31 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 8422, 8422, 8422, 8422, 8422, 8422, ...
## Resampling results across tuning parameters:
##
##   lambda  fraction  RMSE       Rsquared   MAE
##   0e+00   0.050     10198.447  0.4622664  7482.738
##   0e+00   0.525      4867.566  0.8348269  3395.957
##   0e+00   1.000      6172.367  0.8051181  2732.327
##   1e-04   0.050     10202.661  0.4613966  7486.154
##   1e-04   0.525      4874.250  0.8345093  3400.482
##   1e-04   1.000      6009.335  0.8067995  2725.737
##   1e-01   0.050     10260.675  0.4600051  7532.487
##   1e-01   0.525      5011.316  0.8272891  3490.770
##   1e-01   1.000      3749.853  0.8907413  2629.293
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were fraction = 1 and lambda = 0.1.
```

Figure_12: Elastic Net on Train Dataset

e. *KNN*

After we implement KNN on train data set, we got plot of number of K and RMSE. From the plot, we can see the most optimized number of K is 5, and the RMSE, R2 and MAE are 2601.653, 94.71% and 1231.675.

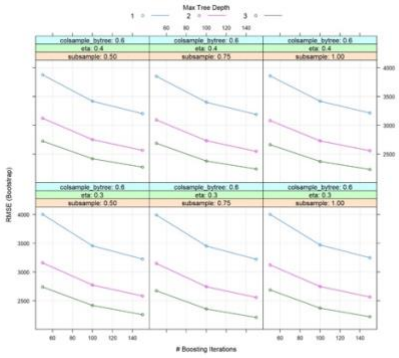



```
## k-Nearest Neighbors
##
## 8422 samples
##   31 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 8422, 8422, 8422, 8422, 8422, 8422, ...
## Resampling results across tuning parameters:
##
##   k  RMSE      Rsquared   MAE
##   5  2601.653  0.9470779  1231.675
##   7  2753.984  0.9408119  1348.401
##   9  2866.505  0.9359094  1446.883
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was k = 5.
```

Figure_13: KNN on Train Dataset

f. *XGBoost*

After we implement XGBoost on train data set, we got plot of boosting iterations and RMSE in different depth, eta, subsample and col samples. From the plot, we can see the most optimized iteration is 150, max depth = 3, eta = 0.4, gamma = 0, colsample_bytree = 0.8, min_child_weight =1 and subsample = , and the RMSE, R2 and MAE are 2133.722, 96.49% and 1254.251, which has the best performance in six models on train dataset.




```
##  parameter 'min_child_weight' was held constant at a value of 1
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were nrounds = 150, max_depth = 3, eta
##  = 0.4, gamma = 0, colsample_bytree = 0.8, min_child_weight = 1 and subsample
##  = 1.
```

RMSE	R^2	MAE
2133.722	0.9649395	1254.251

Figure_14: XGBoost on Train Dataset

VI. Performance Evaluation

After implementing each model on train dataset, we implement model on test dataset to evaluate the performance of each model.

1. *Plot the Prediction Value and True Value in Test Dataset*

We plot the true value of price in test dataset and the prediction price that got from regression model. For each plot, we plot a straight line that means predict value and true value. From the distribution of scatter, we can measure the performance of each model. Form the result, we can see the linear regression, lasso regression, ridge regression is good, but the variance is still a little large. However, the performance of KNN and XGBoost are much better than other models. We also can see that the variance of XGBoost and KNN is much lower than other models.

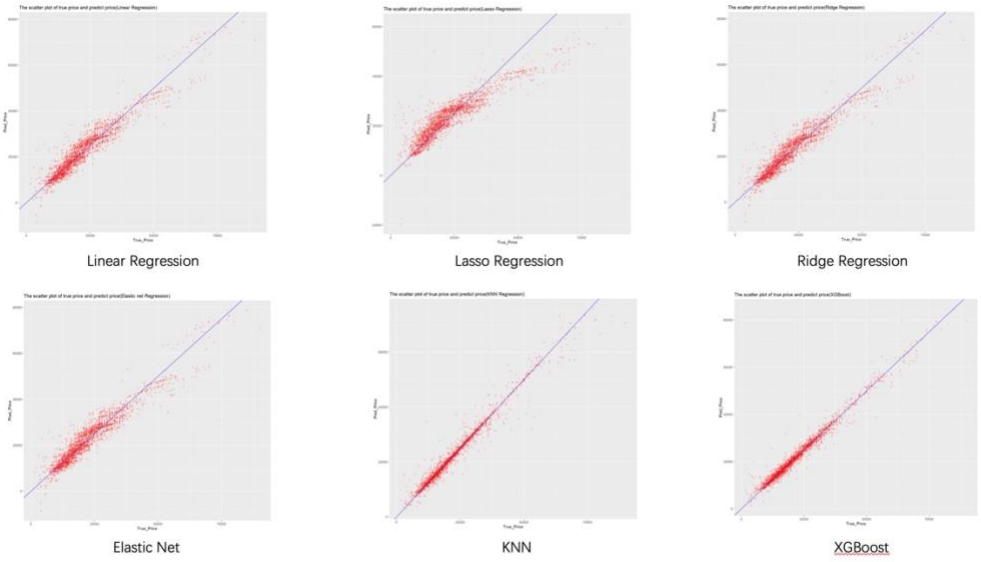


Figure_15: Scatter Plot of Prediction Value and True Value

2. *RMSE, R^2, MAE*

Also, we make a table that contains RMSE, R^2 and MAE on test dataset from each model. From the result, we can see that XGBoost has the best performance, which RMSE, R^2 and MAE are 1848.355, 97.49% and 1179.92. The worst model is Lasso Regression, in which RMSE , R^2 and MAE are 5256.0043,82.29% and 3593.14. From these results, we can see XGboost and KNN perform much better than linear models such as linear regression, lasso regression, ridge regression and Elastic Net.

	RMSE	R^2	MAE
Linear Regression	3535.5402315	0.9081011	2583.9095914
Lasso Regression	5256.0043747	0.8292439	3593.1494320
Ridge Regression	3521.6345848	0.9088323	2578.9292709
Elastic Net	3521.6345848	0.9088323	2578.9292709
KNN	2107.9561276	0.9675403	1056.1253734
XGBoost	1848.3558484	0.9749145	1179.9157517

Figure_16: RMSE, R^2 and MAE of test set

VII. Discussion and Recommendation

From the result, we can see the XGBoost and KNN perform much better than other linear Models, the R^2 for XGBoost and KNN are 97.49% and 96.75% The worst model is Lasso Regression, which R^2 is only 82.92%. The reason for worst performance of Lass could be L1 norm. Since Lasso Regression use L1 norm as penalty term, this could cause some feature weight to 0, which causes overfitting problem. In the future work, we will also try more regression models such as Bayesian Ridge Regression and try other demission reductio methods such as LDA.

VIII. Summary

In this project, we try six regression models including linear model, KNN model, and tree model such as XGBoost on this regression task. After comparing the result in test dataset, we can conclude that XGBoost and KNN have the best performance in this project.

IX. Appendix: R Code for use case study

````
---
title: "Car Price Prediction"
author: "Jiachen_Liu and Thinh_Nguyen"
date: "4/15/2021"
output:
   html_document:
      fig_width: 10
      fig_height: 9
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```
########################################
```{r}
library(ggplot2)
library(tidyverse)
library(dplyr)
```
## EDA
```{r}
car = read.csv('raw_data.csv')
```
########################################
```{r}
head(car)
```
########################################
```{r}
## choose subset of bmw data
bmw = car[which( car$brand == 'bmw' ),]
bmw = subset(bmw,select = -1 )
head(bmw)
```
########################################
```{r}
## summary of bmw data
summary(bmw)
```
########################################
```{r}
## check missing value for each column
````
````

```
library(mice)
library(VIM)
md.pattern(bmw)
aggr(bmw, prop=FALSE, numbers=TRUE)
```
```{r}
colnames(bmw)
```
##
```{r}
a = as.matrix( subset( bmw ,    select =c(price,mileage,tax,mpg )    ))
heatmap( a )

```
##
Exploary Data Analysis
##
```{r}
# box plot for the bmw year
ggplot(data = bmw, aes( x = year, y = price, group =year ) ) + geom_boxplot(color = "blue",alpha = 0.1 ) + ggtitle("Box plot of bmw year")+ theme_minimal()
```
##
```{r}
# box plot of bmw    price
ggplot(data = bmw, aes(x = transmission, y = price )) + geom_boxplot( aes(color = transmission), alpha = 0.1) + ggtitle("Box plot of bmw price")+ theme_minimal()
```
##
```{r}
# boxplot for car model
ggplot(data = bmw, aes( y = price, x = model, group = model ) ) + geom_boxplot( aes(color = model), alpha =0.8 ) + ggtitle("Box plot of car mileage in different model")

```
##
```{r}
# scatter plot of millage and price
ggplot(data = bmw, aes(x = mileage , y = price)) + geom_point(color ="red", alpha= 0.05) + ggtitle("The relationship between millage and price in different transmission")+ facet_wrap(~ transmission)
```
##

```{r}
#scatter plot for car mpg and price
ggplot(data = bmw, aes(x = engineSize , y = price , group = engineSize)) + geom_boxplot(color ="blue", alpha= 0.05) + ggtitle("The relationship between engine size and price")
```
##
```{r}
ggplot(data = bmw[which(bmw$year>2014),], aes(x = mileage , y = price)) + geom_point(color ="red", alpha= 0.05) + ggtitle("The relationship between millage and price in different transmission")+ facet_wrap(~ year )
```
##
```{r}
# bar plot of "Total Number of each transmission"
ggplot( data = bmw , aes(x = transmission ), color = "red" ) + geom_bar(aes(color = transmission)) + ggtitle( "Total Number of each transmission")+ theme_minimal()

```
###
Feature Engineering
###
encoding categrical data
```{r}
# Feature Engineering
library(caret)
library(psych)
library(fastDummies)
```
##
```{r}
bmw_data = dummy_cols(bmw, select_columns =c('year','model','transmission','fuelType','engineSize' ) )
head(bmw_data)
```
##
```{r}
bmw_df = subset(bmw_data ,select =c(-year,-model, -transmission, - fuelType, -brand , -engineSize) )
head(bmw_df)
```
##
Normalize data and prepare for PCA
```{r}
```

```
bmw_scale = as.data.frame(scale(bmw_df[    , -2]))
head(bmw_scale)
```
##
Apply PCA on scale data set and choose apporiate number of components
```{r}
fa.parallel(bmw_scale,fa = "pc",n.iter = 100,show.legend = TRUE )

```
##
Choose components number is 31 and create pca data frame for modeling part
```{r}
# From the result from above , we choose components number = 31
pc = principal(bmw_scale, nfactors = 31 , rotate = "none", scores = TRUE )
```
```{r}
pca_df = cbind(bmw_df$price , as.data.frame(pc$scores))%>%
rename(price = "bmw_df$price")
head(pca_df)
```
###
Split dataset as train and test data set
```{r}
# split train and test data , which 80% train data and 20% test data
library(caTools)
pca_split_data = sample.split(pca_df,SplitRatio = 0.8)
pca_train_data = subset(pca_df,pca_split_data==TRUE)
pca_test_data = subset( pca_df,pca_split_data ==FALSE)

```
###
implementing model
###
linear regression with PCA
```{r}
## linear regression
linear = lm(price ~. , data = pca_train_data)
summary(linear)
plot(linear)
```
###
Lasso Regression with PCA
```{r}
## Lasso Regression
```

```
lasso_pca <- train(price ~ . ,
                   data = pca_train_data,
                   method = "lasso")
lasso_pca
plot(lasso_pca)
```
##
Ridge Regression with PCA
```{r}
## ridge regression
ridge_pca <- train(price ~ . ,
                   data = pca_train_data,
                   method = "ridge")
ridge_pca
plot(ridge_pca)
```
###
Elasticnet with PCA
```{r}
## Elasticnet
enet_pca <- train(price ~ . ,
                   data = pca_train_data,
                   method = "enet")
enet_pca
plot(enet_pca)
```
###
KNN regression with PCA
```{r}
## KNN regression
knn_pca <- train(price ~ . ,
                   data = pca_train_data,
                   method = "knn")
knn_pca
plot(knn_pca)
```
###
XGBoost with PCA
```{r}
## xgboost
xgboost_pca<- train(price ~ . ,
                   data = pca_train_data,
                   method = "xgbTree")
xgboost_pca
```

plot(xgboost_pca)
```
##########################################################
## Model Evaluation
### Linear Regression on test set
```{r}
## RMSE, R2 and MAE in test data set of Linear Regression
linear_pca_pred <- predict(linear, pca_test_data)
as.table(postResample(pred = linear_pca_pred, obs = pca_test_data$price))
ggplot() + geom_point(aes(x = pca_test_data$price , y = linear_pca_pred), color = "red" ,alpha =0.2) + xlab("True_Price") + ylab("Pred_Price") + ggtitle("The scatter plot of true price and predict price(Linear Regression)")+geom_abline (slope= 1, color = "blue")
```
##
Lasso Regression on test set
```{r}
## RMSE, R2 and MAE in test data set of Linear Regression
lasso_pca_pred <- predict(lasso_pca, pca_test_data)
as.table(postResample(pred = lasso_pca_pred, obs = pca_test_data$price))
ggplot() + geom_point(aes(x = pca_test_data$price , y = lasso_pca_pred), color = "red" ,alpha =0.2) + xlab("True_Price") + ylab("Pred_Price") + ggtitle("The scatter plot of true price and predict price(Lasso Regression)")+geom_abline (slope= 1, color = "blue")

```
##
Ridge Regression on test set
```{r}
## RMSE, R2 and MAE in test data set of Ridge Regression
ridge_pca_pred <- predict(ridge_pca, pca_test_data)
as.table(postResample(pred = ridge_pca_pred, obs = pca_test_data$price))
ggplot() + geom_point(aes(x = pca_test_data$price , y = ridge_pca_pred), color = "red" ,alpha =0.2) + xlab("True_Price") + ylab("Pred_Price") + ggtitle("The scatter plot of true price and predict price(Ridge Regression)")+geom_abline (slope= 1, color = "blue")
```
##
Elasticnet on test set
```{r}
## RMSE, R2 and MAE in test data set of Elastic net
enet_pca_pred <- predict(enet_pca, pca_test_data)
as.table(postResample(pred = enet_pca_pred, obs = pca_test_data$price))
ggplot() + geom_point(aes(x = pca_test_data$price , y = enet_pca_pred), color =

"red" ,alpha =0.2) + xlab("True_Price") + ylab("Pred_Price") + ggtitle("The scatter plot of true price and predict price(Elastic net Regression)")+geom_abline (slope= 1, color = "blue")

```
```
##################################################################
### KNN Regression on test set
```{r}
## RMSE, R2 and MAE in test data set
knn_pca_pred <- predict(knn_pca, pca_test_data)
as.table(postResample(pred = knn_pca_pred, obs = pca_test_data$price))
ggplot() + geom_point(aes(x = pca_test_data$price , y = knn_pca_pred), color = "red" ,alpha =0.2) + xlab("True_Price") + ylab("Pred_Price") + ggtitle("The scatter plot of true price and predict price(KNN Regression)")+geom_abline (slope= 1, color = "blue")
```
##
XGboost on test set
```{r}
## xgboost evaluation
xgboost_pca_pred <- predict(xgboost_pca, pca_test_data)
as.table(postResample(pred = xgboost_pca_pred, obs = pca_test_data$price))
ggplot() + geom_point(aes(x = pca_test_data$price , y = xgboost_pca_pred), color = "red" ,alpha =0.2) + xlab("True_Price") + ylab("Pred_Price") + ggtitle("The scatter plot of true price and predict price(XGBoost)")+geom_abline (slope= 1, color = "Blue")
```
```

# X. Reference


1. https://www.kaggle.com/adityadesai13/used-car-dataset-ford-and-mercedes?select=audi.csv
```