Final Project Report

Wenhao Chen & Longfei Wang

Project motivation & Data collection

The price of a new car in the industry is fixed by the manufacturer with some additional costs like taxes. Therefore, customers buying a new car can be assured of the money they invest to be worthy. But duo to the high price of new cars, lots of customers chose to buy used cars. There is a need for a used car price prediction system to effectively determine the worthiness of the car using a variety of features. Even though there are websites that offers this service, their method for prediction may not be the best.

Our goal is to train a linear regression model with a subset of features from our dataset to predict the value of a used car.

We collected the data from CarMax (https://www.carmax.com/), which is a big famous dealer. There are 13 columns in our dataset. They are price, lot, vin, state, country, brand, model, year, drivetrain, transmission, mileage, color, and title type.

	price	lot	vin	state	country	brand	model	year	drivetrain	transmission	mileage	color	title_type
0	$2,050	169398362	**********L822139	Georgia	USA	Nissan	Versa S	2017	Front-wheel Drive	AUTOMATIC	33764	BLACK	Salvage Insurance
1	$2,500	169630683	**********3A57609	California	USA	Mini	Cooper	2015	FWD	Automatic	138732	WHITE	Salvage Insurance
2	$3,700	169755411	**********U275900	Georgia	USA	Toyota	Camry L	2013	Front-wheel Drive	NaN	117335	SILVER	Salvage Insurance
3	NaN	169773065	**********P509208	Georgia	USA	Toyota	Corolla Ec	2016	Front-wheel Drive	NaN	0	WHITE	Salvage Insurance
4	$1,950	169785052	**********Z114727	Texas	USA	Toyota	Corolla S/Le/Xle	2009	FWD	Automatic	145136	WHITE	Salvage Insurance

There are websites that offers an estimate value of a car. They may have a good prediction model. However, having a second model may help them to give a better prediction to their users. Therefore, the model developed in this study may help online web services that tells a used car's market value.

Data Cleaning

- Check the data and drop unnecessary columns. The StockNumber and the Vin are too many unique values, so they have no contribution for predicting the price of car. The model is not an independent feature, and it's a sub feature of the Make feature. Therefore, it needs to be dropped.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29900 entries, 0 to 29899
Data columns (total 13 columns):
StockNumber     29900 non-null int64
Vin             29900 non-null object
Year            29900 non-null int64
Make            29900 non-null object
Model           29900 non-null object
Price           29900 non-null float64
Color           29817 non-null object
Miles           29900 non-null object
DriveTrain      29900 non-null object
Transmission    29900 non-null object
Msrp            28479 non-null float64
Cylinders       29900 non-null int64
EngineSize      29894 non-null object
dtypes: float64(2), int64(3), object(8)
memory usage: 3.0+ MB
```

```
StockNumber     29875
Vin             29875
Year               14
Make               36
Model             528
Price             421
Color              13
Miles             131
DriveTrain          2
Transmission        5
Msrp             1359
Cylinders           7
EngineSize         48
dtype: int64
```

- Check and remove duplicates.

```
len(df[df.duplicated()])
```

```
566
```

```
print(df.shape)
df.drop_duplicates(keep="first", inplace=True)
print(df.shape)
```

```
(29900, 13)
(29875, 13)
```

- Check and deal with NaNs.

```
2 df.isnull().sum()
```

```
StockNumber        0
Vin                0
Year               0
Make               0
Model              0
Price              0
Color             83
Miles              0
DriveTrain         0
Transmission       0
Msrp            1421
Cylinders          0
EngineSize         6
dtype: int64
```

There are not too many None values. Because the price of different model cars may be very different, filling missing values of the Msrp with mean or median value will not be helpful. Therefore, they need to be dropped. Filling missing values of the Color feature and the Engine Size feature with mean or median value is not reasonable, so they also need to be dropped.

- Change some features' data-types.

  The Miles and the Engine Size should be number features instead of category features. So, we removed the letter K and L from their values, and then change their data type from object to float.

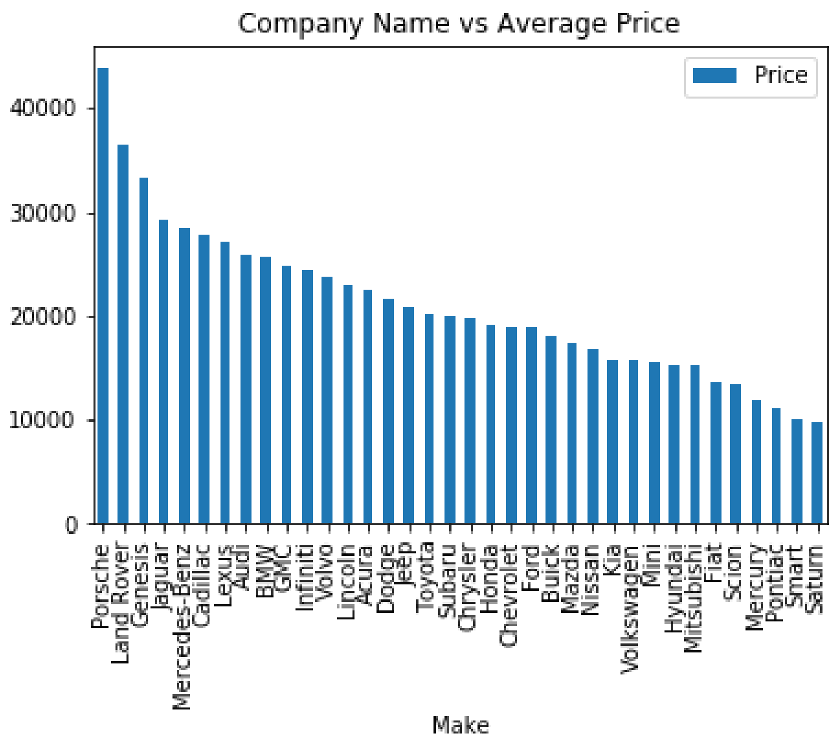
	Year	Make	Price	Color	Miles	DriveTrain	Transmission	Msrp	Cylinders	EngineSize
0	2009	Acura	12998.0	White	55K	2WD	Automatic	35000.0	6	3.5L
1	2014	Acura	17998.0	Black	68K	4WD	Automatic	43400.0	6	3.7L
2	2016	Cadillac	41998.0	Black	32K	4WD	Automatic	79700.0	8	6.2L
3	2015	Cadillac	17998.0	Black	44K	2WD	Automatic	37700.0	6	3.6L
4	2018	Chevrolet	27998.0	White	24K	4WD	Automatic	34000.0	6	3.6L

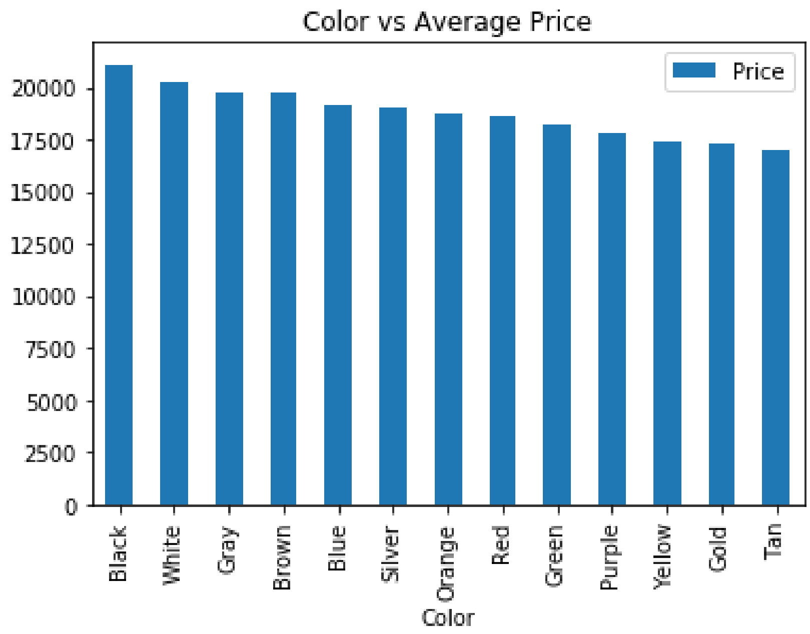
	Year	Make	Price	Color	Miles	DriveTrain	Transmission	Msrp	Cylinders	EngineSize
0	2009	Acura	12998.0	White	55.0	2WD	Automatic	35000.0	6.0	3.5
1	2014	Acura	17998.0	Black	68.0	4WD	Automatic	43400.0	6.0	3.7
2	2016	Cadillac	41998.0	Black	32.0	4WD	Automatic	79700.0	8.0	6.2
3	2015	Cadillac	17998.0	Black	44.0	2WD	Automatic	37700.0	6.0	3.6
4	2018	Chevrolet	27998.0	White	24.0	4WD	Automatic	34000.0	6.0	3.6

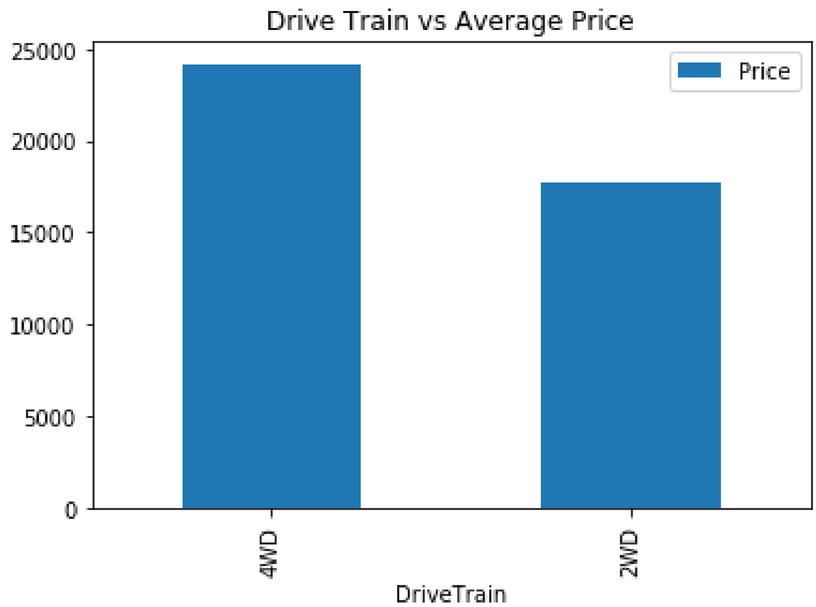
Data Analysis

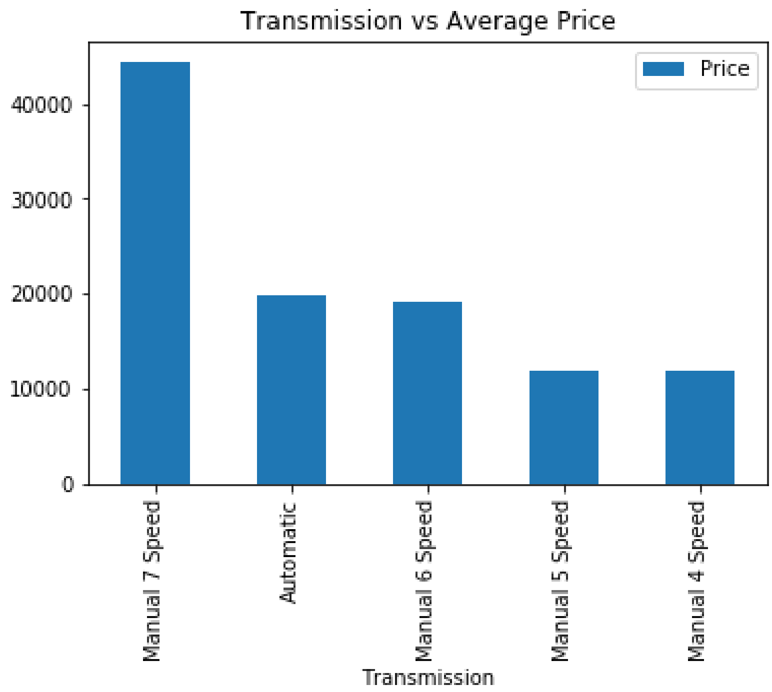
- Check impact some categorical features on the cars' price.

  To predict the price, we need to check the relationship between the categorical variables and the outcome variable to judge whether the categorical variables impact price of the cars.



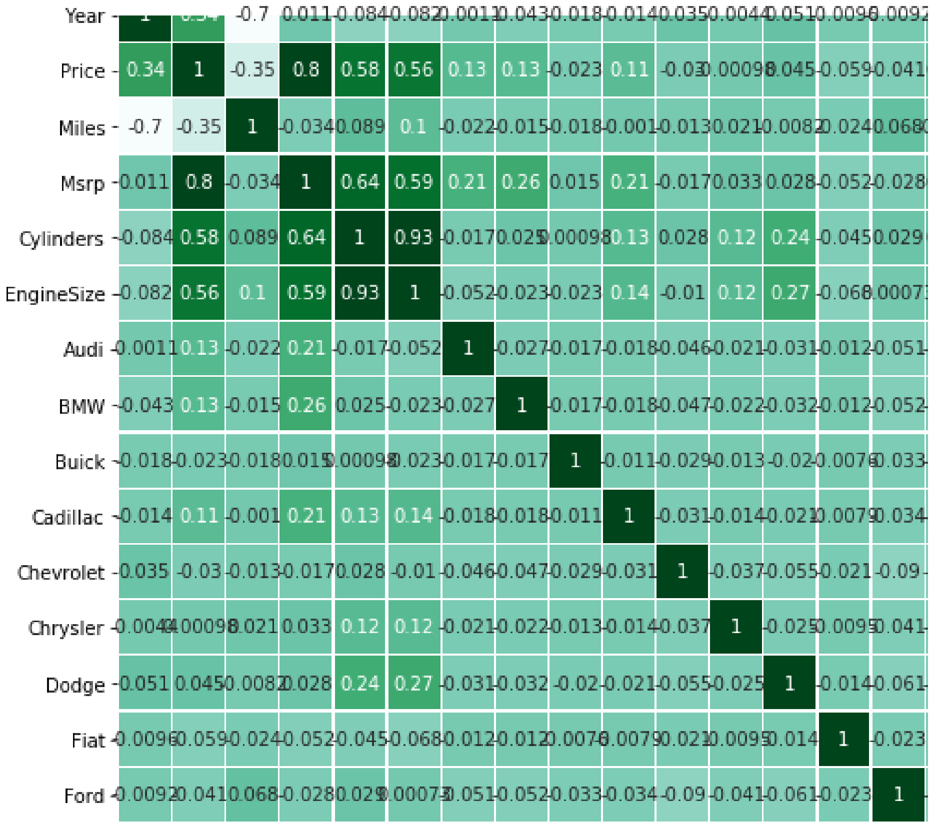

Color vs Average Price
Price
20000
17500
15000
12500
10000
7500
5000
2500
0
Black
White
Gray
Brown
Blue
Silver
Orange
Red
Green
Purple
Yellow
Gold
Tan
Color


Drive Train vs Average Price
Price
25000
20000
15000
10000
5000
0
4WD
2WD
DriveTrain



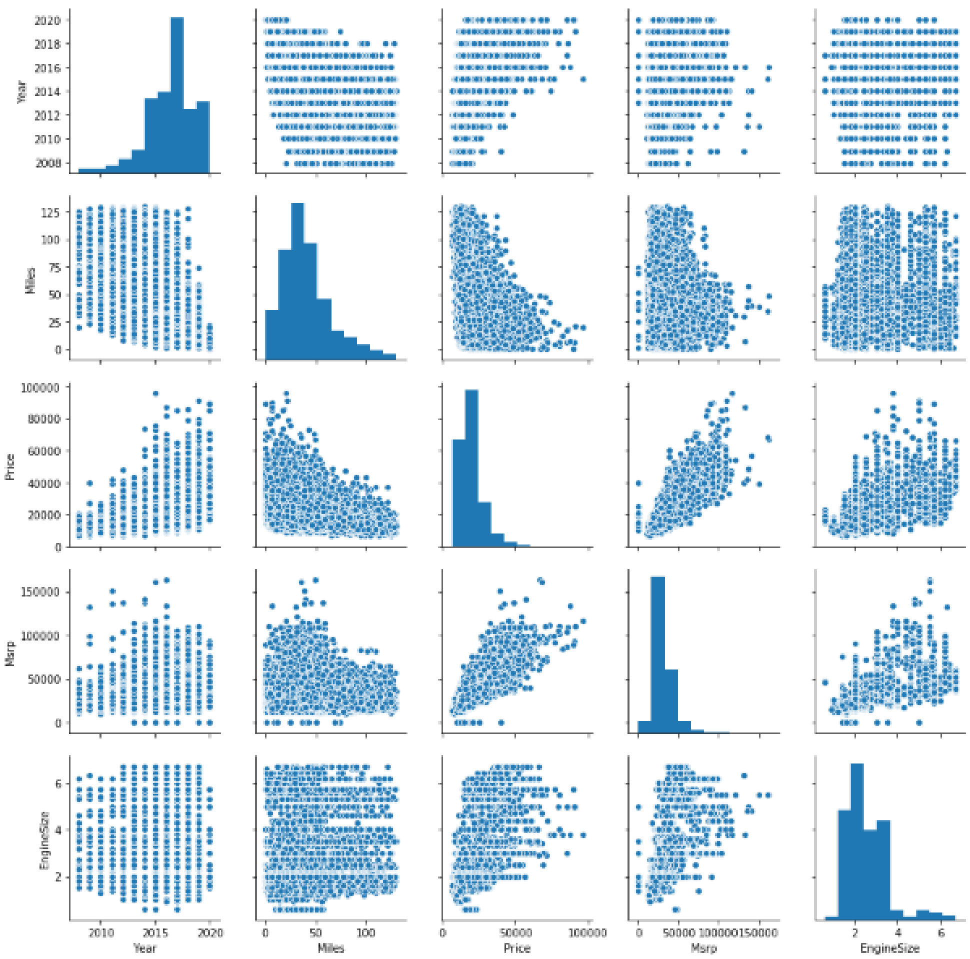
From the plots above, only the colors of the cars have little impact to price. Therefore, we should drop the color variable.

- Calculator correlations and see how features are correlated with the price. First, change remaining categorical variables to dummy variables. Then, check strong correlation among independent variables.



The Cylinders variable has strong correlation with EngineSize Variable in terms of the heatmap above. We need to drop one of them, and we chose drop the Cylinders feature.

- Check the relationship between numerical variables and the outcome variable.



All the Year, Miles, Msrp and EngineSize are quasi-linear with price variable, so we don't need to drop any variables.

- Split the data into training data and testing data.

```
train_data, test_data = train_test_split(df, train_size = 0.7, random_state=1)
```

- Fit the linear model.

```
features = df.drop('Price', axis=1).columns
lm = linear_model.LinearRegression()
lm.fit(df[features], df['Price'])
```

- Show intercept and coefficients of the model.

```
Intercept: -1557007.0230000715
Coefficients: [ 7.74282554e+02 -6.70393045e+01  3.88783719e-01  1.79818317e+03
  1.18099219e+03  1.90078172e+02 -3.51769018e+02 -2.43733193e+03
  9.49755749e+02 -8.01863304e+02 -2.53884362e+02  2.29615361e+02
  1.42401255e+03  1.32917688e+03  1.71192269e+03  3.67456060e+03
  1.26348815e+03 -4.62483201e+02  1.38338738e+03  1.34858794e+03
  8.09734635e+02  3.85180453e+03  3.91748106e+03 -9.52550426e+02
  2.09548712e+03  1.97866499e+03 -1.19032754e+03  2.47596207e+03
  2.95261494e+01  7.58942499e+02  3.91170658e+03  8.37132602e+03
  3.19205098e+03  3.51229847e+03  2.13507803e+03  2.50623156e+03
  3.88268825e+03  1.18509323e+03  1.64389010e+03  1.09804018e+03
 -5.99424590e+02  1.00795840e+03  2.44978628e+03  6.69284814e+03]
```

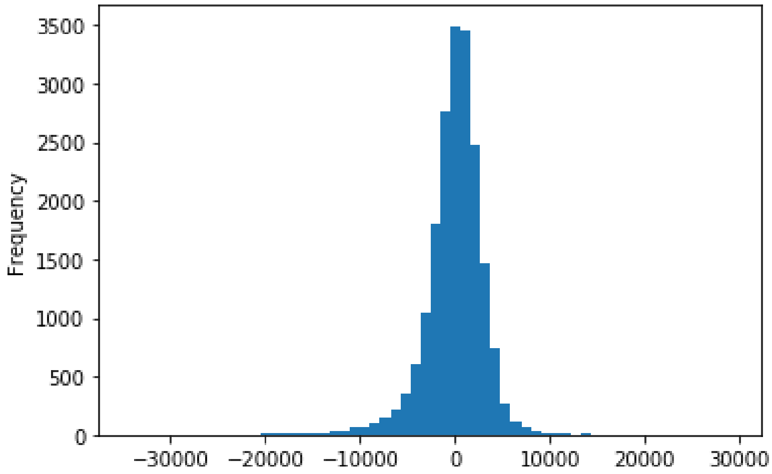
- Make a prediction on testing data and show the model score.

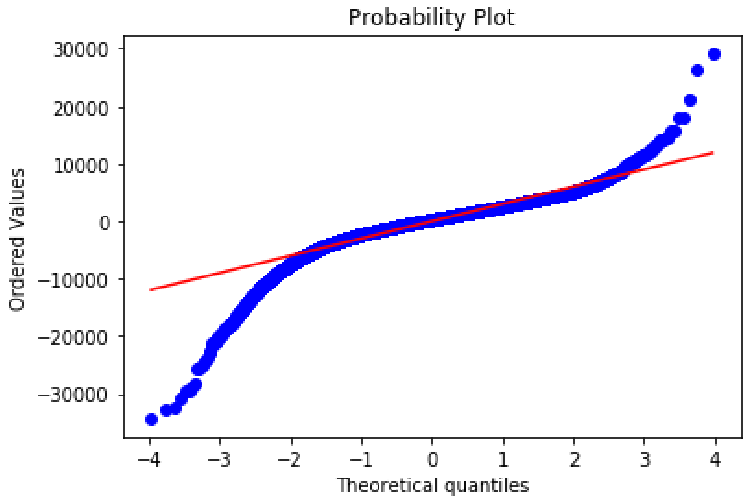
```
pred = lm.predict(test_data[features])

lm.score(test_data[features],test_data['Price'])
```

```
0.8347959231851461
```

- Analyze the residuals.

  Check the distribution of residuals.





The residuals seem to be approximately Normally distributed, so the assumption on the linear modeling seems to be fulfilled. We can see that our model is fairly good. However, here is a problem that the distribution of residuals is a little skewed, not symmetrical, so we want to solve this problem.

The method is adding log transformation to the outcome variable to improve the Normality of residuals.

Model improvement

- Add log transformation to outcome variable.

```
df['Price'] = np.log1p(df['Price'])
```

- Split our data into training data and testing data.

```
train_data, test_data = train_test_split(df, train_size = 0.7, random_state=1)
```

- Fit the new linear model.

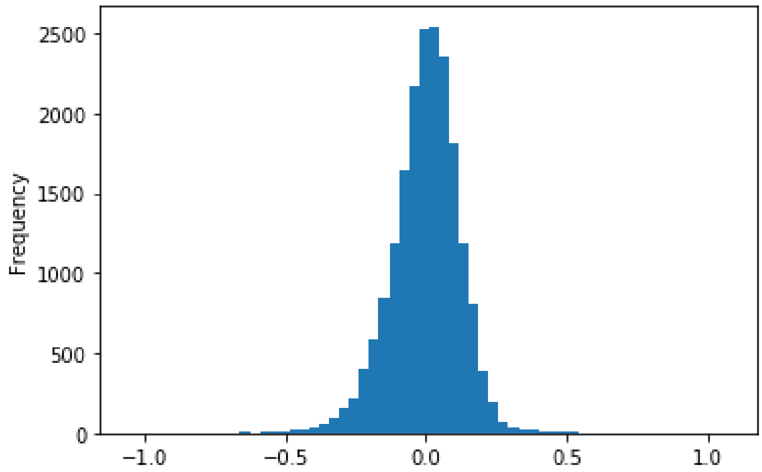
```
features = df.drop('Price', axis=1).columns
lm = linear_model.LinearRegression()
lm.fit(df[features], df['Price'])
```

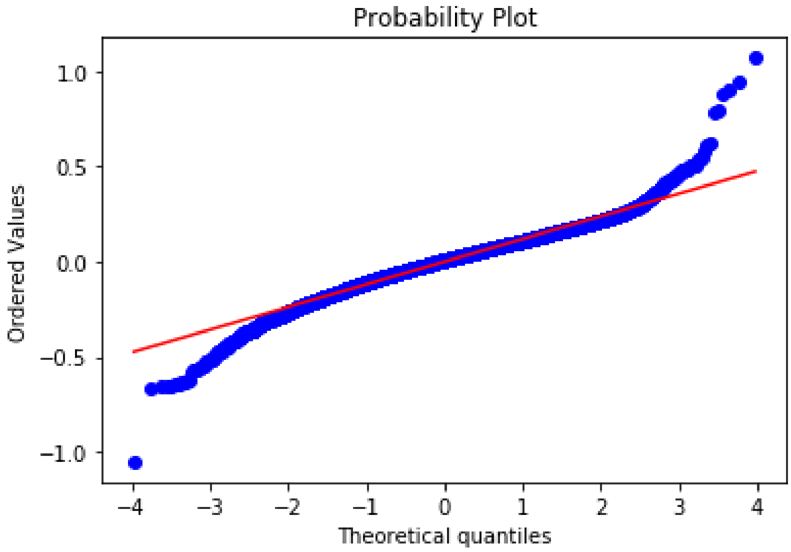
- Show the model score.

```
lm.score(test_data[features],test_data['Price'])
```

```
0.8748337238389039
```

- Analyze the residuals.





We notice that the score is improved, from 0.83 to 0.87, and the Normality is also improved! The model is successfully improved.

Conclusion

The model we improved is fairly good to help us predict the price of used cars and judge whether the price is reasonable.