CAR PRICE PREDICTION IN THE USA BY USING LINEAR REGRESSION

Huseyn Mammadov
Carlo Bo University of Urbino, Italy

Received: *November 18, 2021* Accepted: *December 27, 2021* Online Published: *December 29, 2021*

Abstract

This paper studies a linear regression model to predict the car prices for the U.S market, in order to help a new entrant understanding important pricing factors/variables in the U.S automobile industry. The prediction of a car price has become a high-interest research area, as it requires significant knowledge of the field. I have applied to a highly comprehensive analysis with all data cleaning, exploration, visualization, feature selection and model building. The data used for the prediction were collected from the web portal fred.stlouisfed.org using web scraper, written in Python/Jupyter programming language. According to a problem solving approach, I have split it to 5 parts (Data understanding and exploration, Data cleaning, Data preparation: Feature Engineering and Scaling, Feature Selection using RFE and Model Building and Linear Regression Assumptions Validation and Outlier Removal). The points are symmetrically placed along a diagonal line in the former plot and along a horizontal line in the later plot in the examination plots of observed against forecast values or residuals versus projected values. According to the table of Residuals vs. Predicted, many points with extremely high residual values suggest that the model predicts one item adversely. Other well-known raised residual points may possibly be significant outliers.

Keywords: Car Price Prediction, Liner Regression, Data Understanding, Data Cleaning.

1. Introduction

In this paper the given purpose is to explain the price of cars in the US where the liner regression is used, and which helped to estimate predictions. Respectively, an accurate estimation of automobile prices requires specialized expertise, as quality typically relies on several different features and variables. In addition, the amount of gasoline used in the vehicle and the fuel usage per mile have a significant effect on a car's price leading to regular adjustments in a fuel 's demand.

This analysis is organized in this structure:

- Data understanding and exploration
- Data cleaning

- Data preparation: Feature Engineering and Scaling
- Feature Selection using Recursive Feature Elimination (RFE) and Model Building
- Linear Regression Assumptions Validation and Outlier Removal.

2. Literature Review

Noor and Jan (2017) use multiple linear regression to construct a model for forecasting car prices. The dataset was generated during the two-month span and included the following characteristics: size, cubic ability, exterior color, date of posting of the ad, amount of ad views, power steering, kilometer mileage, type of transmission, type of motor, area, registered area, layout, edition, make and model year. With the Results setup researchers were able to reach 98 per cent predictability. The authors have suggested prediction model based on the single machine learning algorithm in the relevant research seen above. Nevertheless, it is notable that a standard approach to machine learning algorithms did not produce impressive predictive outcomes and could be improved by combining multiple methods of machine learning into an ensemble.

Gonggie (2011) suggested a model that would be developed using ANN (Artificial Neural Networks) to estimate the price of a used vehicle. He considered several attributes: passed miles, estimated car life and mark. The new model was developed in order to cope with nonlinear data interactions, which was not the case for prior models using standard linear regression techniques. The non-linear model was able to forecast car prices better than other linear models with greater accuracy.

Wu et al. (2009) performed analysis of car price estimation utilizing a knowledge-based neuro- fuzzy method. They took the following characteristics into account: model, year of production, and engine size. Their model of projection had comparable findings to the simplistic model of regression. They have created a specialist program named ODAV (Optimal Distribution of Auction Vehicles), since there is a strong demand for auto dealers to deliver the vehicles at the end of the leasing year. This method offers information into the best car rates, as well as the place where the best quality can be earned. Regression model focused on neighboring k-nearest machine learning algorithm was used to predict a car's speed. This program appears to be remarkably effective, as it has exchanged more than two million vehicles.

In his thesis research Richardson (2009) offered a specific approach. His expectation was that more robust vehicles should be made by automakers. Richardson implemented multiple regression analyses and found that electric vehicles have maintained their worth longer than regular vehicles. This has origins in urban warming issues and offers greater fuel efficiency.

3. Methodology and Problem Solving

3.1 Data Understanding and Exploration

Let's first have a look at the dataset and understand the size, attribute names etc. Figure 1 shows the data types and names of the columns of the dataset and according to the estimation Python is used where it helps to apply to the liner regression.

Figure 1 − Understanding the features and data Observations on Target Variable- Price

```
cars = pd.read_csv("auto_with_col (1).csv")
cars.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   car_ID            205 non-null    int64
 1   symboling         205 non-null    int64
 2   CarName           205 non-null    object
 3   fueltype          205 non-null    object
 4   aspiration        205 non-null    object
 5   doornumber        205 non-null    object
 6   carbody           205 non-null    object
 7   drivewheel        205 non-null    object
 8   enginelocation    205 non-null    object
 9   wheelbase         205 non-null    float64
 10  carlength         205 non-null    float64
 11  carwidth          205 non-null    float64
 12  carheight         205 non-null    float64
 13  curbweight        205 non-null    int64
```

The target variable price has a positive skew; however, majority of the cars are low priced. More than 50% of the cars (around 105-107 out of total of 205) are priced 10,000 and close to 35% cars are priced between 10,000 and 20,000. So around 85% of cars in US market are priced between 5,000 to 20,000. Based on above observations and graph on right side (KDE/green one) it appears there are 2 distributions one for cars priced between 5,000 and 25000 and another distribution for high priced cars 25,000 and above. (Notice the approximate bell curve from little less than 30000 up to 45,000/50,000).

Data Exploration

To perform linear regression, the target variable should be linearly related to independent variables. Let's see whether that's true in this case. Figure 2 shows the tabular form of the dataset on which we will carry the operations.

Figure 2 − Var indicators

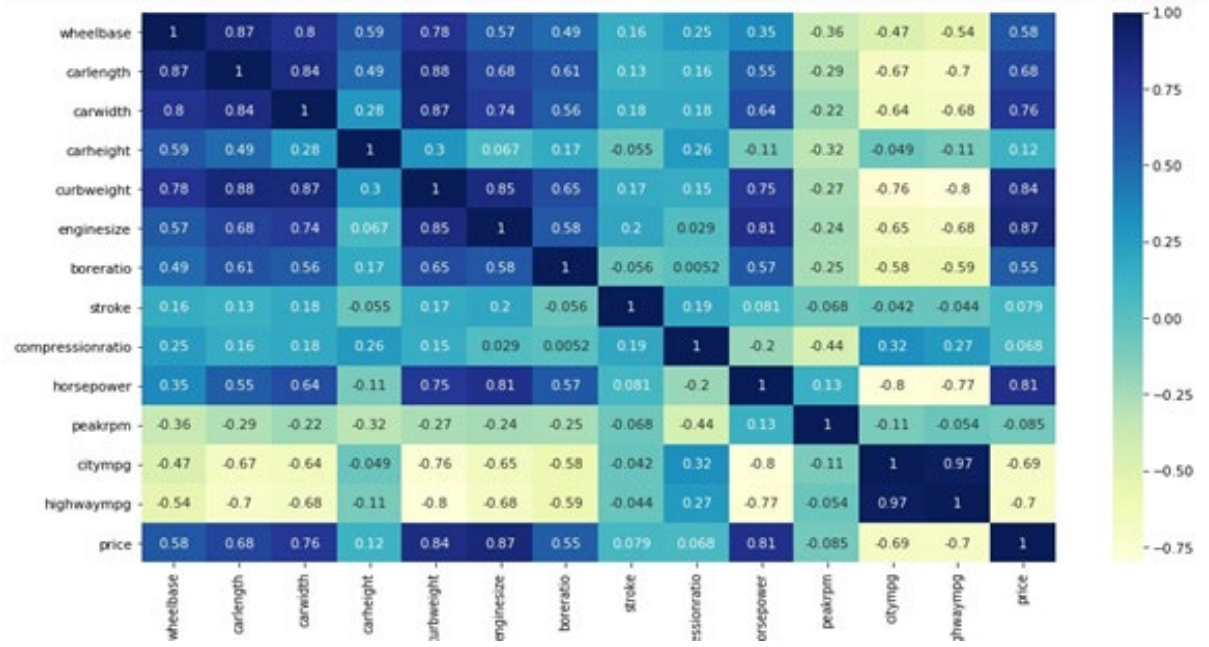
```
#creating df  with numeric var's only
cars_numeric=cars.select_dtypes(include=['float64','int64'])
cars_numeric.head()
```

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	boreratio	stroke	compressionratio	horsepower	peakrpm	citympg
0	1	3	88.6	168.8	64.1	48.8	2548	130	3.47	2.68	9.0	111	5000	21
1	2	3	88.6	168.8	64.1	48.8	2548	130	3.47	2.68	9.0	111	5000	21
2	3	1	94.5	171.2	65.5	52.4	2823	152	2.68	3.47	9.0	154	5000	19
3	4	2	99.8	176.6	66.2	54.3	2337	109	3.19	3.40	10.0	102	5500	24
4	5	2	99.4	176.6	66.4	54.3	2824	136	3.19	3.40	8.0	115	5500	18

These vars appears to have a linear relation with price: carwidth, curbweight, enginesize, horsepower, boreration and citympg. Other variables either don't have a relation with price or relationship isn't strong. None of the variables appear to have polynomial relation with price.

In linear regression assumptions validation section, we will check for linearity assumption in detail. Figure 3 shows the useful insights from Correlation Heatmap (which shows a 2D correlation matrix between two discrete dimensions), dependent variables and independent variables.

Figure 3 — Heatmap Correlation



Positive correlation: price highly correlated with enginesize, curbweight, horsepower, carwidth (all of these variables represent the size/weight/engine power of the car)

Negative correlation: price negatively correlation with mpg var's citympg and highwaympg. This suggest that cars having high mileage may fall in the 'economy' cars category or in other words indicates that Low priced cars have mostly high mpg

Correlation among independent variables: many independent variables are highly correlated; wheelbase, carlength, curbweight, enginesize etc. are all measures of 'size/weight', and are positively correlated

Since independent variables are highly correlated (more than 80% correlation among many of them) we'll have to pay attention to multicollinearity, which we will check in assumptions validation section using VIF score

3.2 Data Cleaning: Missing values and feature data type check

In this section we will check dataset for missing values and check the datatypes of different features. Figure 4 shows the data types and names of the columns of the dataset and meanwhile Figure 5 shows the conversion of desire column.

Figure 4 — The types of columns of the dataset

```
cars.info()

#no missing values

#all var's in correct format, however since symboling is a categorical var we need to change its type
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   car_ID            205 non-null    int64  
 1   symboling         205 non-null    int64  
 2   CarName           205 non-null    object 
 3   fueltype          205 non-null    object 
 4   aspiration        205 non-null    object 
 5   doornumber        205 non-null    object 
 6   carbody           205 non-null    object 
 7   drivewheel        205 non-null    object 
 8   enginelocation    205 non-null    object 
 9   wheelbase         205 non-null    float64
 10  carlength         205 non-null    float64
 11  carwidth          205 non-null    float64
 12  carheight         205 non-null    float64
 13  curbweight        205 non-null    int64  
 14  enginetype        205 non-null    object 
 15  cylindernumber    205 non-null    object 
 16  enginesize        205 non-null    int64  
 17  fuelsystem        205 non-null    object 
 18  boreratio         205 non-null    float64
 19  stroke            205 non-null    float64
 20  compressionratio  205 non-null    float64
 21  horsepower        205 non-null    int64  
 22  peakrpm           205 non-null    int64  
 23  citympg           205 non-null    int64  
 24  highwaympg        205 non-null    int64  
 25  price             205 non-null    float64
dtypes: float64(8), int64(8), object(10)
memory usage: 41.8+ KB
```

Figure 5 − The conversation of the column

```
# converting symboling to categorical
cars['symboling'] = cars['symboling'].astype('object')
cars.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   car_ID            205 non-null    int64  
 1   symboling         205 non-null    object 
 2   CarName           205 non-null    object 
 3   fueltype          205 non-null    object 
 4   aspiration        205 non-null    object 
 5   doornumber        205 non-null    object 
 6   carbody           205 non-null    object 
 7   drivewheel        205 non-null    object 
 8   enginelocation    205 non-null    object 
 9   wheelbase         205 non-null    float64
 10  carlength         205 non-null    float64
 11  carwidth          205 non-null    float64
 12  carheight         205 non-null    float64
 13  curbweight        205 non-null    int64  
 14  enginetype        205 non-null    object 
 15  cylindernumber    205 non-null    object 
 16  enginesize        205 non-null    int64  
 17  fuelsystem        205 non-null    object 
 18  boreratio         205 non-null    float64
 19  stroke            205 non-null    float64
 20  compressionratio  205 non-null    float64
 21  horsepower        205 non-null    int64  
 22  peakrpm           205 non-null    int64  
 23  citympg           205 non-null    int64  
 24  highwaympg        205 non-null    int64  
 25  price             205 non-null    float64
dtypes: float64(8), int64(7), object(11)
memory usage: 41.8+ KB
```

3.3 Data Preparation: feature engineering

In this section we prepare the data for model building and desire operations. Enable to make future operations we prepared the data. Data preparation contains drop, merge, and creating dummies. Scaling features though not necessary in (Multiple Linear regression) MLR but it's good to do it as it makes interpretation of regression coefficients easier

3.4 Model Building and Feature Selection Using RFE (Recursive FeatureElimination)

Since our dependent variable price looks to be linearly related to most of the independent variables, we are using Linear Regression (because of in statistics when dependent variable is linearly related to independent variable then we apply Linear Regression) only and no other types of regression like Polynomial, Random Forest/Boosting regression etc.

Massive overfitting: all features in model is never a good idea unless features are too less and all of them are important, so we used using recursive feature elimination to reduce dimensionality. First, we need to split the data into train and test as shown in Figure 6. Then we perform some R-square and root mean squared error (RMSE) on train and test data and we obtain some values of R-square on train and test data as well and also RMSE on train and test data after performing these operations as these values are clearly shown in Figure 6.

Figure 6 – Data Split

```
In [53]: # Model with all features
         from sklearn import linear_model
         from sklearn.linear_model import LinearRegression

         lm=LinearRegression()
         lm.fit(X_train,y_train)

         y_pred_test=lm.predict(X_test)
         y_pred_train=lm.predict(X_train)
```

```
In [54]: ## Evaluation metrics

         #Rsqaure
         from sklearn.metrics import r2_score

         print('R-sqaure on train data: {}'.format(r2_score(y_true=y_train, y_pred=y_pred_train)))
         print('R-sqaure on test data: {}'.format(r2_score(y_true=y_test, y_pred=y_pred_test)))

         #Standard error/RMSE
         error_train=y_pred_train-y_train
         error_test=y_pred_test-y_test

         print('RMSE on train data: {}'.format(((error_train**2).mean())**0.5))
         print('RMSE on test data: {}'.format(((error_test**2).mean())**0.5))

         R-sqaure on train data: 0.9756892503873289
         R-sqaure on test data: 0.8382621393399081
         RMSE on train data: 1213.0021196738412
         RMSE on test data: 3365.465234388078
```

Feature selection using RFE

First we decide optimal number of features rather than arbitrarily specifying count of features to be used in model in the RFE function.

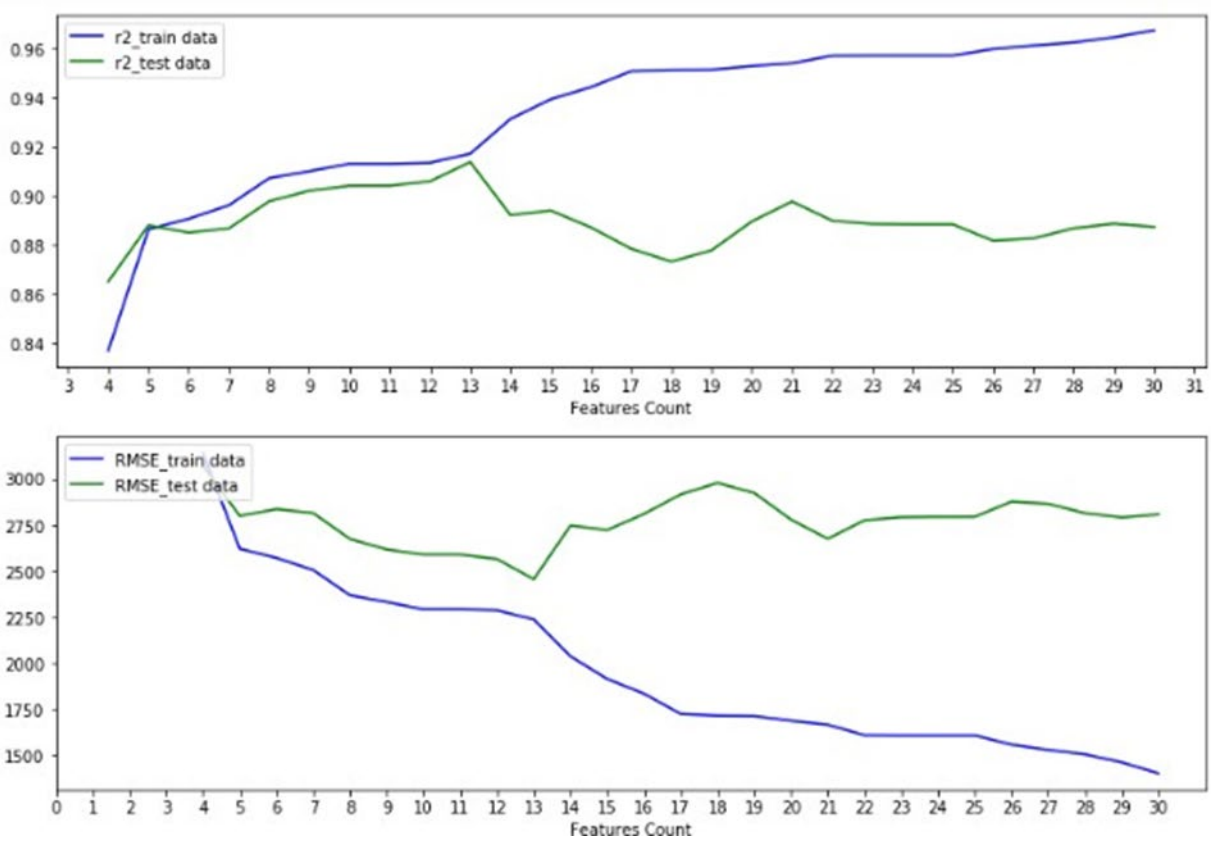
From the graphs as shown in Figure 7 we find:

- R square for test data peaks at 13 features and at this point model generalizes well as train R2 is v close to test. Train R2 keeps on increasing beyond 13 features but R2

keeps increasing as you add more features to train data. We have selected the number of features where model accuracy and generalization both are at satisfactory level.

– RMSE for test data is lowest at 13 features and beyond that it increases. Train RMSE at 13 also looks good, adding more features to train decreases RMSE but again there is always a tradeoff between removing features (aka reducing complexity) and model performance. So, we will go with 13 features (Figure 8).

Figure7 − Features count



3.5 Linear Regression

To detect linearity lets inspect plots of observed vs. predicted values or residuals vs. predicted values. The desired outcome is that points are symmetrically distributed around a diagonal line in the former plot & around horizontal line in the latter one.

From the graphs shown in Figure 9:

1. Obs vs predicted shows that most of the values are closer to the diagonal line, however some are not which is a problem.
2. Resi vs pred graph does not give a conclusive evidence that residuals are evenly scattered around the zero line as Resi values increase with increase in predicted values, so assumption of linearity can't be confirmed.
3. There seems to be presence of outliers, which might be giving a non-conclusive enough Resi vs Predicted graph. Some points have very high residual values; a point (~ -3000, ~ 8000) shows one value is predicted negatively by the model. There are many other prominent high residual points which could be influential outliers.

Figure 8 − Model Building with optimal features

```
----------------------R-square-----------------------------------------
R-sq for test data is 0.9006167424476529
R-sq for train data is 0.9172871967113265
----------------------STANDARD ERROR/RMSE-------------------------------------
RMSE for test data is 2455.655255247883
RMSE for train data is 2237.4263820988704
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.917
Model:                            OLS   Adj. R-squared:                  0.910
Method:                 Least Squares   F-statistic:                     120.1
Date:                Sat, 23 May 2020   Prob (F-statistic):           3.47e-64
Time:                        12:25:33   Log-Likelihood:                -1305.9
No. Observations:                 143   AIC:                             2638.
Df Residuals:                     130   BIC:                             2676.
Df Model:                          12
Covariance Type:            nonrobust
==============================================================================================
                             coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------------------
const                   1.332e+04    200.770     66.366      0.000    1.29e+04    1.37e+04
carwidth                1622.7341    455.659      3.561      0.001     721.266    2524.202
curbweight              2009.8697    574.805      3.497      0.001     872.688    3147.052
enginesize              5314.0047    888.479      5.981      0.000    3556.256    7071.753
boreratio              -1613.9636    427.436     -3.776      0.000   -2459.595    -768.332
stroke                  -945.7383    286.937     -3.296      0.001   -1513.409    -378.068
enginelocation_rear     1178.1547    369.229      3.191      0.002     447.679    1908.630
enginetype_rotor         714.2205    162.095      4.406      0.000     393.535    1034.906
cylindernumber_five      837.7928    283.402      2.956      0.004     277.116    1398.470
cylindernumber_four     1047.0367    526.494      1.989      0.049       5.431    2088.642
cylindernumber_twelve  -629.3369    261.164     -2.410      0.017   -1146.018    -112.655
cylindernumber_two       714.2205    162.095      4.406      0.000     393.535    1034.906
car_company_bmw         1729.5095    211.128      8.192      0.000    1311.819    2147.200
car_company_porsche     1011.0676    289.741      3.490      0.001     437.850    1584.286
==============================================================================
Omnibus:                       30.332   Durbin-Watson:                   2.020
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               61.739
Skew:                           0.922   Prob(JB):                     3.92e-14
Kurtosis:                       5.638   Cond. No.                     2.56e+16
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 8.86e-31. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

Observations from above numbers:

1. R-sqaure for both test & train looks good
2. RMSE for both is closer (test is little < train which is fine and shows that model generalizes well)

Figure 9 − Comparation of Observed and Predicted Values

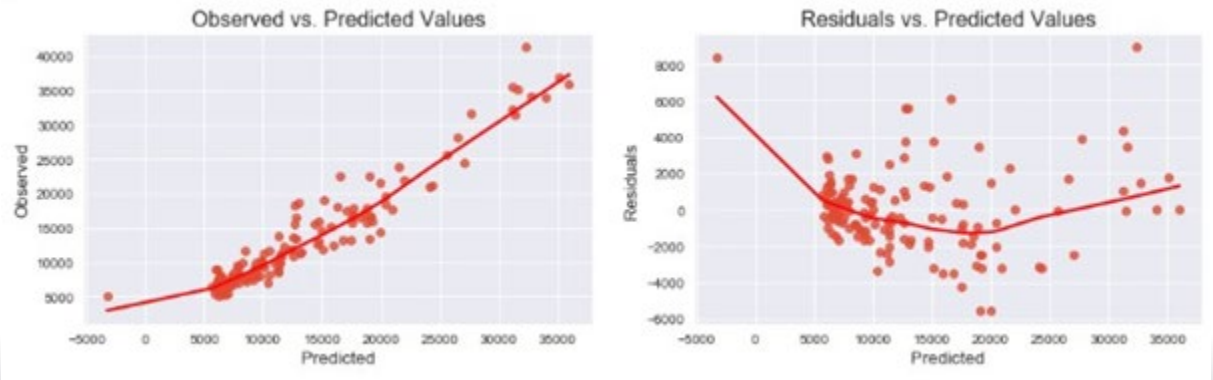
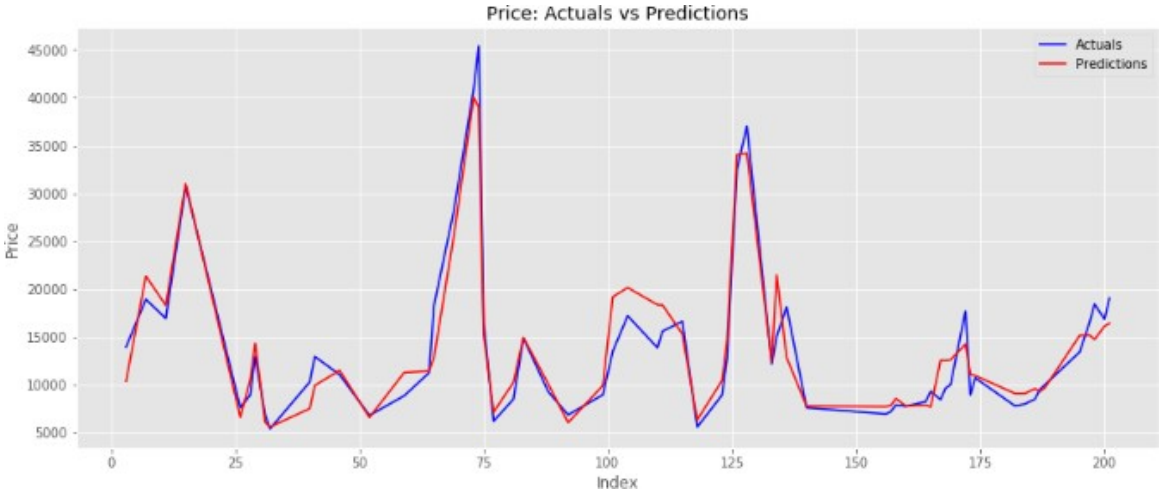


Figure 10 indicates the Actual vs Predictions price. Blue label indicates the actual price of the cars and red label indicates the predicted values of the cars.

Figure 10 − Relation of Actuals and Predictions



4. Conclusions

According to the aim and methodology of this research may apply other countries with using the same statistical analysis. The precise structure explains and indicates poses if the variables where the price fluctuated over the cars with subject to the results and according to the understanding the features and data observations on target variable Price, the estimation illustrates information about, the goal variable price has an optimistic bias because most vehicles are low cost. Over 50 percent of the vehicles are priced at 10,000 and approximately 35 % are priced between 10,000 and 20,000. So, in the US industry about 85 per cent of cars are priced between 5,000 and 20,000.

On the basis of the above findings and graph on the right side there are 2 distributions: one for cars priced between 5,000 and 25,000 and another for high priced cars, at 25,000 and beyond. In the data exploration we have started to perform the liner regression. So, in detail, some var's seem to have a linear price relation: carwidth, curbweight, enginesize, horsepower, boreration, and citympg and certain factors either have no price relation or are not good association. Neither of these variables appear to have a polynomial relation to size. In the segment Validation of linear regression assumptions, we have tested for the linearity assumption. In the correlation heatmap, price is highly correlated with enginesize, curbweight, horsepower, carwidth and negatively correlated with mpg var's citympg and highwaympg, so, this means that high-mileage cars can fall into the 'economy' car category or, in other words, mean that low-priced cars often have high mpgs.

In the examination plots of observed versus forecast values or residuals versus projected values, the intended consequence is that the points are symmetrically arranged in the former plot along a diagonal line and in the latter along a horizontal line. According to the Residuals vs Predicted table, many points have very high residual values, indicating that the model negatively forecasts one value. There are also other elevated residual points that may theoretically be powerful outliers.

References

1. Du, J., Xie, L. Schroeder, S. (2009). Practice Prize Paper − PIN Optimal Distribution of Auction Vehicles System: Applying Price Forecasting, Elasticity Estimation and Genetic Algorithms to Used-Vehicle Distribution. *Marketing Science*, 28(4), 637-644.
2. Gelman, A., Hill, J. (2006). *Data Analysis Using Regression and Multilevel Hierarchical Models*.Cambridge University Press, New York, USA.
3. Gongqi, S., Yansong, W., Qiang, Z. (2011). New Model for Residual Value Prediction of the Used Car Based on BP Neural Network and Nonlinear Curve Fit. *In Measuring Technology and Mechatronics Automation (ICMTMA), 2011 Third International Conference*, Vol. 2, pp. 682-685, IEEE.
4. Listiani, M. (2009). *Support Vector Regression Analysis for Price Prediction in a Car LeasingApplication*. Thesis (MSc). Hamburg University of Technology.
5. Noor, K., Jan, S. (2017). Vehicle Price Prediction System using Machine Learning Techniques.*International Journal of Computer Applications*, 167(9), 27-31.
6. Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kauffmann.
7. Richardson, M. S. (2009). Determinants of used car resale value. Retrieved from: https://digitalcc.coloradocollege.edu/islandora/ 3A1346 [accessed: August 1, 2020.]
8. Used cars database. (n.d.) Retrieved from: https://fred.stlouisfed.org/
9. Wu, J.D., Hsu, C.C., Chen, H.C. (2009). An expert system of price forecasting for used cars using adaptive neuro-fuzzy inference. *Expert Systems with Applications*, 36(4), 7809-817.