Car Price Prediction Using Machine Learning

Redi Lleshi (PID: 6282547)

Florida International University

CAP 4612 (Introduction To Machine Learning)

Dr. Mustafa Ocal

July 23, 2023

Introduction

The ability to find the right price for an item is one the most difficult, but yet important topics when it comes to the business world. It can define the ground where the consumer and the producer meet. One such market that depends on finding the right price is the car sale market. The car market is incredibly large and diverse, with different models fitting different needs, from small, practical, and efficient cars, to sports cars, utility cars, and so on. This gives the customer a lot of options, but it also makes it difficult for them to know which model best suits their needs, and what price they should pay. Also, the dealer needs to know how much their car is worth, and what can they expect to get for it. This leaves a grey zone between the customer and the dealer, which we can try and solve using machine learning.

Machine Learning allows us to find patterns in the data, and formulate a model with which we can predict the outcome based on the input given to it. This ability makes machine learning a great tool to attempt to solve the issue that we mentioned above. By analyzing the attributes that each car has, we can generate a price. The more data that we can get for the car prices, the better the outcome will be. The price of a car can vary depending on a lot of factors, like the brand of the car, the model of the car, the fuel type, the exterior color, and so on. Furthermore, there can be spatial and emotional factors that can affect the price of a car, for example, a car that is sold on the East Coast of the United States could have a different price than that of a car sold on the West Coast of the United States, despite having the same attributes. Also, people are willing to pay more for a car brand they love, even though there could be other better options in the market for cheaper. These factors won't take part in our analysis, but it is important to point them out for future directions.

As stated by Tarique Akhtar in his work "Predicting Car Price Using Machine Learning | by Tarique Akhtar", it is important to differentiate the attributes that have an effect on the model, from those that might not have a significant impact. These attributes do not help the model that much, but removing them allows the model to run faster and smoother. Having this in mind, I plan to see how each attribute in the dataset for our model correlates to the price attribute, and from there, we will be selecting the attributes with the highest correlation. From there, we will split the data into 80% test and 20% train, where we will analyze the results and see how they match the actual prices.

Related Works

Car price predictions have been a hot topic forever, but it always has been difficult to know whether the results can be applicable to real life. With the introduction of Machine Learning, there have been many analyses done using it, and my analysis extends from the experience of those other analyses. There are multiple different models that can be used to decide how to predict the price of a car. For example, in the analysis done by Panwar Abhash Anil in his work "Used Car Price Prediction Using Machine Learning | by Panwar Abhash Anil", he uses models like KNearestRegressor, ExtraTreesRegressor, DecisionTreeRegressor Estimator, and so on. They are useful since they can take multiple different values as input, and can produce a continuous result. In our model, the input will be the attributes, and the output will be the price, which is a continuous value.

Furthermore, another issue that is present in such a model is the issue surrounding the separation of categorical and numerical datasets. In the article we mentioned above from Tarique Akhtar, we saw how he separated the attributes into these two categories. But, we still have to deal with the categorical data, since the models we will use don't recognize categorical data as input. Tarique used the dummy variables to represent the categorical values, which does allow for the conversion of the categorical attributes to numerical ones, but it adds a lot of new attributes we have to deal with. An easier way would be to add weights to the categorical attributes, and one way to do this would be by binning the attributes. We can choose to bin by mean, mode, or median. For our model, we will bin the categorical attributes by their corresponding median price, which will allow us to represent them as numerical values.

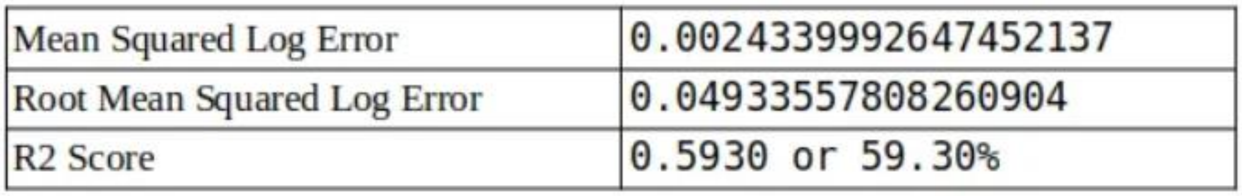
The dataset that we will be using is from Kaggle, from the user Tugberk Karan, and it has multiple attributes which give a good description of all cars. After doing data cleaning and processing, the categorical and numerical attributes are as follows:

Categorical = ['brand', 'model', 'engine', 'transmission', 'fuel_type', 'drivetrain', 'interior_color', 'exterior_color']

Numerical =['min_mpg', 'max_mpg', 'navigation_system', 'memory_seat', 'leather_seats', 'apple_car_play/android_auto'] (These are only the main ones)

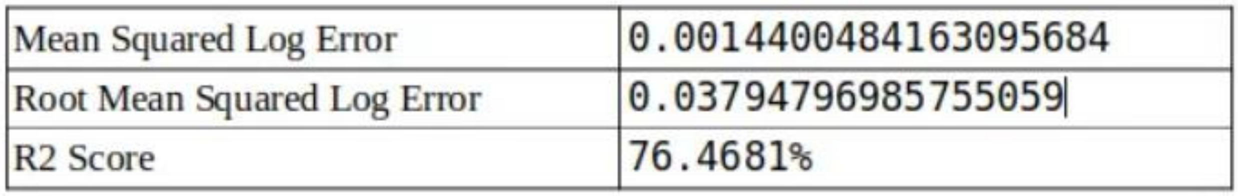
From the dataset that Panwar uses, he has generated these scores of his models.

For the linear regression model:



Mean Squared Log Error	0.002433992647452137
Root Mean Squared Log Error	0.04933557808260904
R2 Score	0.5930 or 59.30%

For the K-Nearest Neighbour model



Mean Squared Log Error	0.0014400484163095684
Root Mean Squared Log Error	0.03794796985755059
R2 Score	76.4681%

The scores that he generated varied between different algorithms, but the K-Nearest Neighbour model was one of the best-performing models for him. We will use these two models to test our dataset, and then we will see how our dataset test results compare to his dataset results.

Data

The dataset that we will be using, from Kaggle by user Tugberk Karan, has more than 20000 values with 36 attributes. After performing data cleaning and processing, we are left with about 14000 tuples, which should provide significant information for our training models. We will be using the attributes that we defined earlier in the categorical and numerical categories. Next, we will add the weights to the categorical values, using binning by mean. Below are the resulting correlation matrixes of the categorical and numerical attributes.

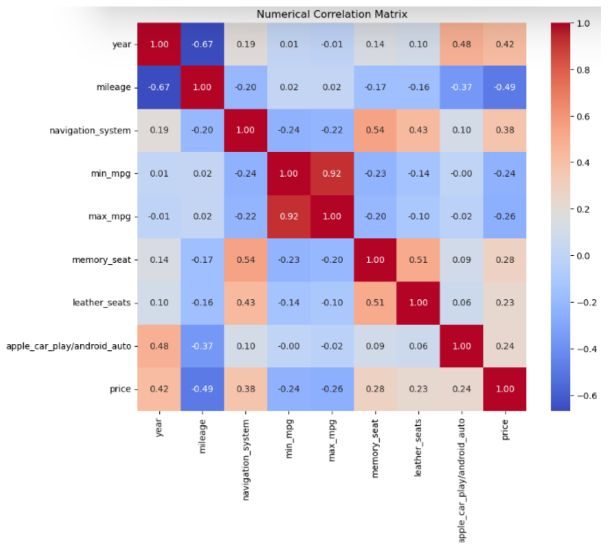


Fig. 1 - Numerical Correlation Matrix

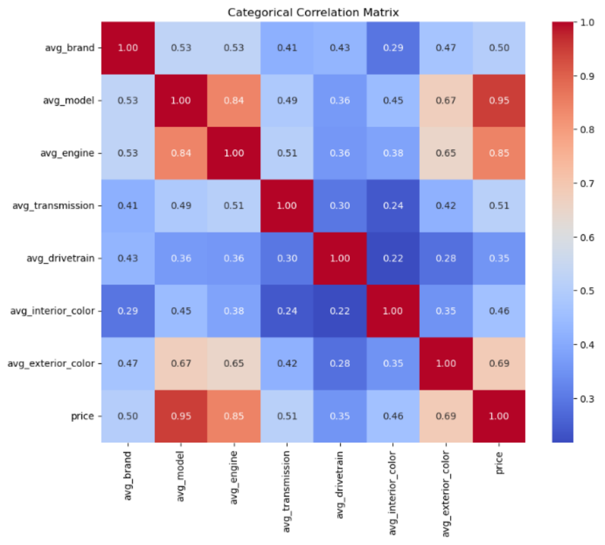


Fig. 2 - Categorical Correlation Matrix

From the above matrixes, we can see how different attributes correlate with the price target. From there we can choose whether we want to drop further attributes or continue with the current setup. With our current setup, those attributes will be included in our models. The train and test data will be divided as shown below

	Test	Train
Percentage (%)	80	20
Number	11200	2800

Implementation

Libraries Used

As for the libraries used, for my models, I will be using the pandas, seaborn, numpy, matplotlib.pyplot, and sklearn libraries. From the sklearn library, we will be using all the tools needed for our models and testing. These libraries were sufficient to implement all the tools needed to design this model. It is worth pointing out that the numpy and seaborn libraries were not used a lot, but it does provide help in a few sections.

The pandas library is a very important library. It allowed us to implement the dataset, and convert it into a data frame, which is the format that Python uses to work on the provided data. It allows us to manipulate the data and format it into a shape that is useful and functional to our plan for the project. The numpy library is used to take data from the data frame and insert it into an array, which is much easier to modify and work on, as well as protects the main body of the data from being accidentally damaged. The seaborn library is only used to draw visual correlations between the attributes, and it allows us to see how they stand in relationship with one another. The matplotlib.pyplot is another very important library, which allows us to design and draw graphs, which can show us how the data is distributed. It can point out whether or not there is a correlation between two attributes, if there are outliers in the data, or if the models we are trying to implement meet the criteria that we have set for our model. The last library used is the sklearn (scikit-learn) library, which is also another very important library. It is used to design and implement all the models, train them, and obtain the output from them. The output provided will mark the predicted price, and we will be using and analyzing it to see how our model performs.

Pipeline

The first step in our code implementation will be to import all the necessary libraries. From there we can continue using the tools that these libraries offer. The second step would be to import the data from the file that it was stored in. From there, it goes into modifying and shaping the data. In between each modification of the data, we will be doing graphical analyses to see if the changes made fit the target we are trying to achieve.

Once the data was formatted to the standard we had set for it, the next step will be getting into the modeling phase. For our modeling algorithms, we will be using the K-Nearest Neighbour algorithm and the Multilinear Regression algorithm. The K-Nearest Neighbour is imported from the sklearn library, under the KNeighboursClassifier module. This algorithm works by taking in numerical attributes and comparing them with the other values in the data set. After it has done comparing itself with them, it calculates the distance between the current value and those other values, and it chooses the K-Nearest values out of all values. From there it chooses which value is closest, using the weight function. This function allows adding weights to the distances, and the closer the distance to the target value is, the more weight it has, and the value with the highest weight is chosen as the value predicted. Meanwhile, the Multilinear Regression algorithm is a little easier to understand, since it is based on mathematical functions. It uses a function in the form of $y = b0 + b1 * x + b2 * x + \ldots$, where the input attributes are taken as x, and the output is delivered as y. In this case, y represents the value of the predicted price. The main issue I encountered with this algorithm is the fact that it produces negative values. Multilinear Regression generates a linear function, but it has no bounds, thus it might be precise, but it can also generate unrealistic values, such as negative values.

Data Processing

When it comes to data processing, it was important to modify the data into a shape that can positively affect our models. First, there was the data cleaning, which we talked more about earlier in the "Data" section. Next, a very big issue I found during my data processing was leftover mistakes from the owner of the dataset. In the price category, there were values in the price attribute stating 'ot Priced', which turned the entire column from an integer datatype to an object datatype, which needed cleaning and conversion.

After the above step, we had to separate the categorical from the numerical attributes. Then we needed to add a numerical value that would represent the categorical attributes in the models that we will use. The following step is to see how the attributes correlate with each other and the price, and from there we will choose the attributes that have the highest correlation with the price attribute. This step is necessary since it will allow our models to run more efficiently and effectively. For further details on these stems, we have covered them in the "Data" section, as well as see how they correlate with each other.

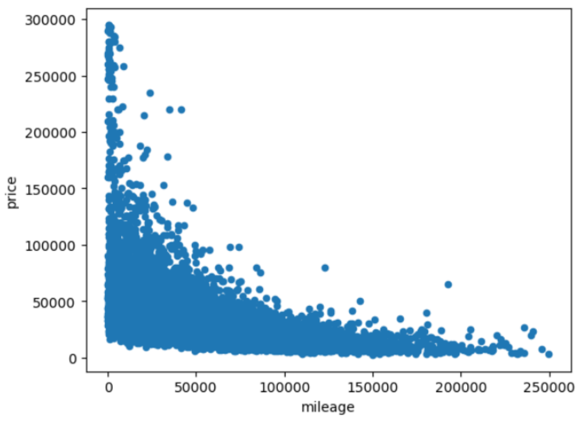


Fig. 3 Mileage vs. Price graph

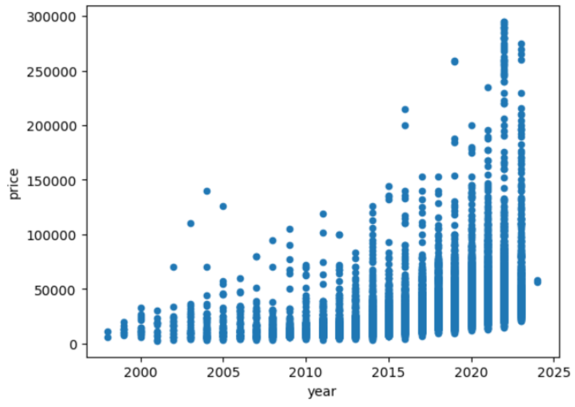


Fig. 4 Year vs. Price graph

Above we can see two images of how the data is disturbed after the data processing. In Fig.3, we can see and understand that the higher the mileage is, the less the price should be, and in Fig. 4, the newer the car is, the more expensive it will probably be. It is worth pointing out that the graph in Fig. 3, represents a hyperbola (y = 1/x) graph, which could lead to complications since the correlation considers the relationship only as linear, and not as hyperbolic.

Performance Measurement Techniques

For the evaluation, since the result is continuous, I used evaluation methods MAE (mean absolute error), MSLE (mean squared logarithmic error), and R^2. For this model, you can also use the MSE (mean squared error) and RMSE (root mean squared error), but these values are disputed across different tests done by other people on the same topic.

Algorithm Used	K-Nearest Neighbour	Multilinear Regression
MAE	4700.8997	4419.6643
MSLE	0.0461	-
R^2	0.9258	0.9331

In the table above, we can see that the Multilinear Regression does not have a representing value for the MSLE. This is because MSLE does not accept negative values, and as we mentioned earlier, Linear Regression models can generate negative values. In some test sets, MSLE worked, but in others, it didn't generate a value. For that reason, I didn't want to include it since it wasn't consistent.

Results

Findings

The results of my tests point out relatively similar values, meaning that the conclusion from the dataset is consistent. The R^2 is also a good sign that the models are performing well, while the MAE is relatively small compared to the average price, which is about 36000$.

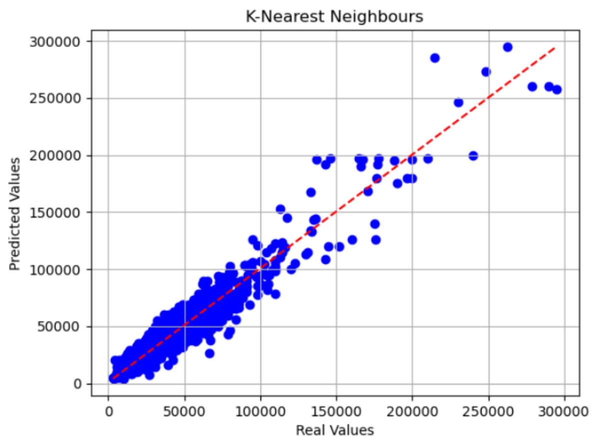


Fig. 5 - K-Nearest Neighbours Graph Graph

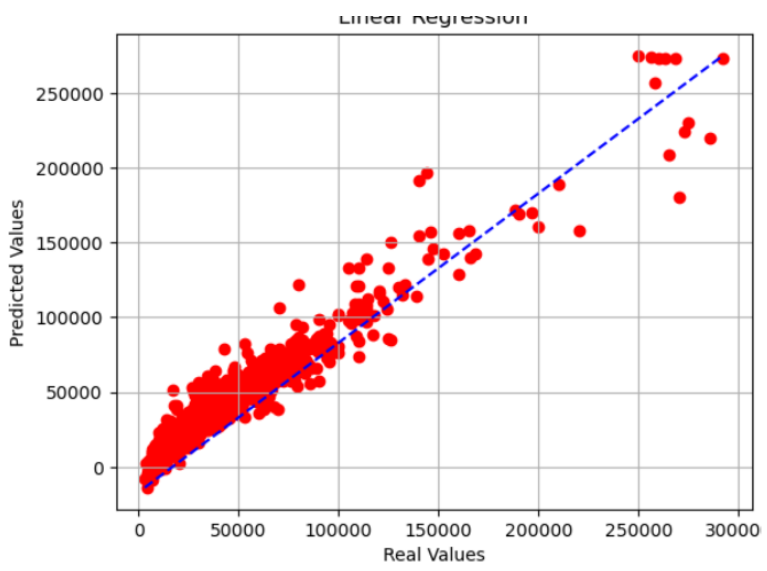


Fig. 6 - Linear Regression Price Graph

The results above show how the predicted price compares to the real prices. We can see that the results from the trendline depict linear growth on both graphs, but in the linear regression we can see that for values between 0 and 50000 for the Real Values, there is a curve, which could have been affected by the hyperbolic relationship between the mileage and price.

Final Discussion

Future Work

In the future, one of the main targets would be to expand the data. This means that we might need to add both in terms of new values, and new attributes. An important attribute would be the location where the car is located at. Different locations have different taxes, different incomes, and so on, so having a location represented in the dataset is important since it holds a lot of information.

Also, I would like to split the data into smaller subcategories, so that we can remove any correlation that is non-linear. As we mentioned above in the mileage vs price graph, if we were to break the graph down into subsections, we could form sections with linear correlations, and from there we would train the models based on these breakdowns. As a result, the entire system would be more complex since we have to use more models for each section, but it would generate more accurate values.

Another improvement would be to work more on outliers. In a dataset of continuous values, there could be many outliers and they could affect the performance of our models in a negative way. I tried removing as many outliers as I could, but there are still outliers that can trick the models. Furthermore, I would like to use other more complex models and see how much could we reduce the MAE and how accurate can we get to the real price of cars.

References

Akhtar, Tarique. “Predicting Car Price using Machine Learning | by Tarique Akhtar.” *Towards Data Science*, 23 November 2020, https://towardsdatascience.com/predicting-car-price-using-machine-learning-8d2df3898f16. Accessed 22 July 2023.

Anil, Panwar Abhash. “Used Car Price Prediction using Machine Learning | by Panwar Abhash Anil | Aug, 2020 | Towards Data Science.” *Towards Data Science*, 3 8 2020, https://towardsdatascience.com/used-car-price-prediction-using-machine-learning-e3be02d977b2. Accessed 22 July 2023.

Karan, Turgberk. “Used Car Listings: Features and Price Prediction.” *Kaggle*, 27 June 2023, https://www.kaggle.com/datasets/tugberkkaran/used-car-listings-features-and-prices-carscom. Accessed 22 July 2023.