MPG Calculator

Khunsa Umer, kumer@gmu.edu,

Scott Morgan. smorga@gmu.edu

Volgenau School of Engineering
George Mason University
Fairfax, VA, United States

ABSTRACT

Automotive companies are always trying different ways to improve the efficiency of the car both in cost and environmental effects because car buyers prefer cars that cost less to maintain and do not harm the environment. The emphasis on making the environment healthy is increasing due to current pandemic and global warming. That is why this calculator was created, so automotive companies or people who are interested in building the most efficient car can get help from it. Users can pick the parts of the car as they wish, and the model would give them the projected miles per gallon (MPG) of their vehicle on the highway or in a city. The question arises of why not just buy electric vehicles? As of now, electric vehicles cost much more than other vehicles due to their efficiency and building cost which the average person cannot afford.

The first step to building a smart regression model was to find relevant data. Then that data would be preprocessed, and finally, the best regression model would be chosen and tuned to increase the accuracy of the predictions for the city and highway MPG's.

The goal was to make program that can help people predict the miles per gallon (MPG) of a car they are trying build with specific choices, so they can build the most fuel-efficient car. The model calculates MPG for city and highway separately because cars will naturally have a lower average MPG when driving through a city due to traffic lights. The model also has the user interaction feature, so they can decide their own parameters like the type of fuel they want their car to use, how much horsepower they want in their car, how many cylinders they want to put in their car, the transmission type, etc., to build the most efficient car.

KEYWORDS

Gradient boosting, mean absolute error, fine tune, hyperparameter, measured, efficiency, lowest error.

APPROACH

1. Getting Data from Kaggle:

The first step was looking for data that had all the information related to cars with their MPG, so that the final model would be able to give an accurate prediction. The data set was collected from Kaggle. [2]

2. Preprocessing:

The data set that we got form Kaggle was relatively clean, there were a few columns that only contained words that described a class of something (i.e., transmission type), so a label encoder was used to convert those strings into numeric form, so it is readable for machine learning model. All the electric cars were removed from the dataset, as this model is about predicting the MPG of cars that use gasoline to power their engine due to the lack of affordability of electric vehicles.

3. Calculating accuracy:

Calculating accuracy to test the model is the most important part of any machine learning algorithm. Since the project was using regression model, mean absolute error was used to determine how accurately the model was able to predict the MPG of a car. Mean absolute error is commonly used to describe data and to explain the relationship between one dependent binary variable and other independent variables in regression models. This was a suitable method to calculate the accuracy of our regression model due to our method of prediction.

4. Choosing the best regression model:

Choosing the best regression model was arguably the longest part of the project. There were several regression models that were tested with the data set to see which one would produce the lowest Mean Absolute Error (MAE). All the tests run with the different regressors set the "random state" to 10, to avoid having to deal with randomness by setting the random seed. The following regression models were tested:

a. Logistic regressor.
b. Linear regression.
c. Ridge regressor.
d. SGD regressor.
e. K nearest neighbor regressor.
f. Decision tree regressor.
g. Gradient boosting regressor.

a. Logistic regressor:

Logistic regression model is a predictive analysis tool that is used to describe the relationship between multiple independent variables and one binary dependent variable. Since it was exactly what we were trying to achieve, it made sense to try this model on our data set. [3]

Results from logistic regressor:

The goal of this project was to get the lowest miles per hour (MPG) for city and highway based on the values entered by the user. Logistic regression model produced mean absolute error of 2.25 for city and 3.18 for highway. The error was not that high, but we wanted to still try to decrease the error percentage further to get the lowest possible error.

b. Linear regressor:

Linear regression is the simplest regression model that is used for predictive analysis. It is commonly used for determining the strength and effects of independent variables on the binary dependent variable. That is why we chose this model because our data also has multiple independent values which we are using to get the dependent value. [3]

Results from Linear regressor:

Since our data was complex, we did not get good results from linear regression model. The mean absolute error for city miles per hour was 3.26, and 3.69 for highway.

c. Ridge regressor:

Ridge regression is another variation of linear regression model. It is also used for similar purposes. Hence, we decided to use this model for our data set.

Results from Ridge regressor:

We got the exact same results from ridge regression as linear regression for both city and highway, so we decided not to use this model for as the final model.

d. SGD regressor:

This model is another simple yet very efficient approach for predictive analysis. It is an optimized method that usually produces good results because it defines a loss function. It gave good results in previous similar data sets, so we decided to try this model as well. [3]

Results from SGD regressor:

The mean absolute error from this model was better than linear regressor and ridge regressor models as it was 3.19 for city and 3.68 highway but it was still lower than logistic regression model. Hence, this model was not used as the final model.

e. K nearest neighbor regressor:

K nearest neighbor regressor is another very efficient predictive analysis model. This model produced good results in the past for similar regression models, so we decided to use this model. It usually uses a simple technique of calculating the average of the numerical target of the k nearest neighbors.

Results from K nearest neighbor regressor:

The results from this model were good as compared to the previous models that we had tried earlier. We got 1.92 mean absolute error for city and 2.3 mean absolute error for highway. We tried changing different hyper parameters to lower the error, but it did not change that much. It was a good result, but we wanted to try a couple more regression models before choosing the final model.

f. Decision tree regressor:

Decision tree regressor is also another great model used to solve problems that require predicting dependent value from different independent values. The reason for testing the data with this model was that if we add boosting to this model, it produces great results, and if we do not get the desired result from models, we will use this model and add boosting to it to improve the results.

Results from decision tree regressor:

The results from decision tree model were promising and could get better after adding boost to it. We got the mean absolute error of 1.93 for city and 2.34 for highway.

g. Gradient boosting regressor:

Gradient boosting is a great method that makes weak models perform better. This model is derived from SGD which was a weak model because it produced the lowest results out of all the models tested. That is why we thought it would be a good idea to try this model. [3]

Results from gradient boosting regressor:

The results that we got from this model were the best out of all the models we had tried. We got 1.52 mean absolute error for city and 1.95 mean absolute error for highway without fine tuning the model. Hence, we decided to fine tune this model to get better result and use it as our final model.

4.1 Testing gradient boosting with different parameters:

After choosing gradient boosting as the final model, we then decided to further fine tune hyperparameters to see if we can get a better result. We started this process first by removing columns that did not contribute to improving the result, for example, make, model, and popularity. Then we tried several hyperparameters, but the following are the only parameters that contributed to improving the accuracy.

a. Max_depth (city 5, highway 4):

This parameter determines the maximum depth of the tree to avoid overfitting. Sometimes if the maximum depth is not specified, it is common to have overfitting in some models. By adding this hyperparameter, we improved the error from 1.52 to 1.31 for city and 1.95 to 1.82 for highway. [1]

b. N_estimators (city 1000, highway 500):

This parameter determines the number of trees which can also help to ovoid overfitting in the model. By adding this parameter on top of max_depth, the error rate went down from 1.31 to 1.19 for city and 1.82 to 1.71 for highway.

c. Min_samples_leaf:

This parameter determines the minimum number of sample leaves. This parameter again also helps to avoid overfitting in the model. This parameter does not always help because we tried adding this parameter to calculate highway MPG, and it did not help instead it increased the error rate. So, we decided to only use it to calculate city MPG because it changed the city error from 1.19 to 1.17. [1]

4.2 Final table:

Table 1 shows the result of all the regression models tested with their error rates to choose the best model. Table 2 shows the further fine tuning of the hyperparameters for the final model to get the lowest error.

Table 1: Regression models with error rates

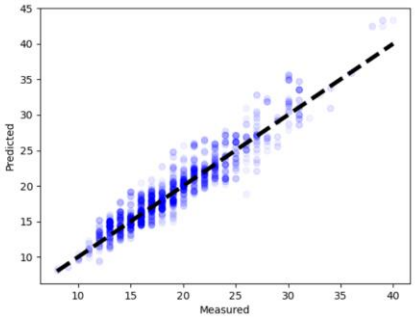
Regressor	MAE city	MAE Highway
logistic	2.25	3.18
ridge	3.26	3.69
SGD	3.19	3.68
Linear	3.26	3.69
Gradient Boosting	**1.53**	**1.88**
knn	1.92	2.3
Decision Tree	1.93	2.34

Table 2: Adding hyperparameters to improve the accuracy

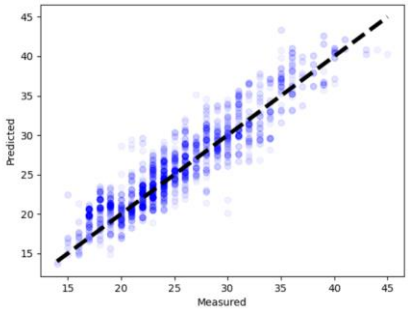
Gradient Boosting parameters	MAE city	MAE Highway
Gradient Boosting(criterion='mae')	1.52	1.95
max depth, city 5, highway 4	1.31	1.82
depth city=5, highway=4 n_estimators 1000,500	1.19	1.71
same, min_sample_leaf= 4 for city	1.17	1.73

RESULTS

The results that we got from the final model was better than expected because the mean final mean absolute error came out to be 1.13 for city and 1.73 for highway. Below is the graph showing the result from predicted and measured value which is almost a line. This indicates that the predicted values were almost the same or close as the measured values. There was also a higher density of values closer to the line, than farther away.



Graph 1 City



Graph 2 Highway

City MPG error rate was less then highway error rate, the datapoints are closer in Graph 1 as compared to Graph 2.

CONCLUSION

Overall, this is a great model for someone/company looking for ways to improve the efficiency and cost of the car and contribute to making the environment healthy at the same time by using less fuel. The best part about this model is that user can pick the parameters themselves and the model will tell the most efficient result. Users can also trust this model because it has extremely low error rate of only 1.13 for city MPG and 1.73 for highway MPG. The low error is the result of testing multiple regression models, dropping unused columns, and fine tuning hyperparameters of gradient boosting regression model which was used as the final model.

ACKNOWLEDGMENTS

We thank our professor Sanmay Das who taught us this valuable material to be able to do this project. We also thank our GTA who was always available to answer questions whenever needed. Lastly, Edmunds and Twitter for information on Kaggle. And CooperUnion for making a useful data set.

Video Presentation link

https://www.youtube.com/watch?v=shBPKmH5uEQ&feature=youtu.be

Please click the link on the final report.

REFERENCES

[1] Aarshay JainAarshay. *Gradient Boosting: Hyperparameter Tuning Python*. Retrieved December 12, 2020, from https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/

[2] CooperUnion. (2016, December 21). Car Features and MSRP. Retrieved December 12, 2020, from https://www.kaggle.com/CooperUnion/cardataset

[3] Learn. (n.d.). Retrieved December 12, 2020, from https://scikit-learn.org/stable/