Michael Lankford

Machine Learning Term Project

04/18/2022

CSCI 4336 Introduction to Machine Learning

Project Overview

The goal for this project was to construct a linear regression or logistic regression gradient descent algorithm for our choice of data set. I chose the “2022 Fuel Consumption Ratings” dataset off of Kaggle and set out to produce a linear regression gradient descent algorithm that could predict city fuel consumption of a vehicle based off its engine size, number of cylinders, and CO2 emission ratings.

Formulas

Mean Squared Error – MSE =

Mean Squared Error Derivative – Derivative MSE =

Mean Absolute Percentage Error – MAPE = 100 \*

Preprocessing

For this section and the Visualizations section, I used Jupyter notebook to clean and visualize my dataset. I started off this step by looking to see what correlations there were between the columns of my data to see which features would be good to predict city fuel consumption, so I used the Pandas .corr() method. As you can see from Figure 1: Correlation Table, the Fuel Consumption(City (L/100 km) column is what I want to predict and the features Engine Size(L), Cylinders, and CO2 Emissions(g/km) have correlations 0.816440, 0.939394, and 0.949703, respectively. These features will work very well for my linear regression gradient descent algorithm.

Following this, I moved to cleaning my dataset because I had some missing values in the features that I planned to use. I used the .fillna() method from Pandas to fill in the null values for my Engine Size(L), Cylinders, and CO2 Emissions(g/km) features with the mean of their respective columns, which was calculated from the .mean() method and also rounded. I continued by dropping the columns I would not need, which were Model Year, Make, Model, Vehicle Class, Transmission, Fuel Type, Fuel Consumption(Hwy (L/100 km)), Fuel Consumption(Comb (L/100 km)), Fuel Consumption(Comb (mpg)), CO2 Rating, Smog Rating.

Next, I split my data into an 80% train set and a 20% test set and also made a copy of them that would become my train and test data sets with my target values. After that, I dropped the features in the data sets that I only wanted target values in and dropped the target value from the data sets that I only wanted features in. This resulted in two files for my train set, one with my features and one with my target values, and two files for my test set, one with my features and one with my target values. Finally, I scaled my features because there was a great size difference between the features Engine Size(L) and Cylinders compared to the CO2 Emissions(g/km) feature. This can be seen in my Figure 2: 2022 Fuel Consumption Ratings Values which show that Engine Size(L) and Cylinders are small numbers while CO2 Emissions(g/km) are in the hundreds.

Visualizations

Table

Description automatically generatedGraphical user interface

Description automatically generated with low confidence

Figure 1: Correlation Table

A screenshot of a computer

Description automatically generated with medium confidence Graphical user interface, application

Description automatically generated

Figure 2: 2022 Fuel Consumption Ratings Values

Chart, scatter chart

Description automatically generated

Chart, scatter chart

Description automatically generated

Figure 3: Pair plot relations

Chart, scatter chart

Description automatically generated

Figure 4: CO2 Emissions(g/km) vs Fuel Consumption(City(L/100 km))

Chart

Description automatically generated

Figure 5: Cylinders vs Fuel Consumption(City(L/100 km))

Chart, scatter chart

Description automatically generated

Figure 6: Engine Size(L) vs Fuel Consumption(City(L/100 km))

Data

The “2022 Fuel Consumption Ratings” data set contained 15 columns of values and they were Model Year, Make, Model, Vehicle Class, Engine Size(L), Cylinders, Transmission, Fuel Type, Fuel Consumption(City(L/100 km)), Fuel Consumption(Hwy (L/100 km)), Fuel Consumption(Comb (L/100 km)), Fuel Consumption(Comb (mpg)), CO2 Emissions(g/km) , CO2 Rating, Smog Rating and out of these columns I chose to use four, Fuel Consumption(City(L/100 km)) as my target and Engine Size(L), Cylinders, and CO2 Emissions(g/km) as the features I would feed to my algorithm.

As you can see from my Figure 4: CO2 Emissions(g/km) vs Fuel Consumption(City(L/100 km)), Figure 5: Cylinders vs Fuel Consumption(City(L/100 km)), and Figure 6: Engine Size(L) vs Fuel Consumption(City(L/100 km)) visualizations, each of my target features have linear relationships with City Fuel Consumption and this shows that there is good correlation between them. This is good because the features I need to use to predict city fuel consumption would need to be related or else I would result in a high error rate.

You can further see the relationships between all the features and my target feature in Figure 3: Pair plot relations and, like the previous paragraph states, they have linear relationships with each other and are highly correlated.

Implementation

I coded my linear regression gradient descent algorithm in Java through Apache Netbeans. Beginning the implementation phase of this project, my first goal was to be able to read my data from my files and to pass it to another class that would hold my algorithm. I accomplished this through use of the BufferedReader class which was imported. Following this, I passed the BufferedReader variables that held the read files of input features into 2-D Array Lists, and I passed the BufferedReader variables that held the read files of target features into a 1-D Array List. These values would be passed through a class object into the main file.

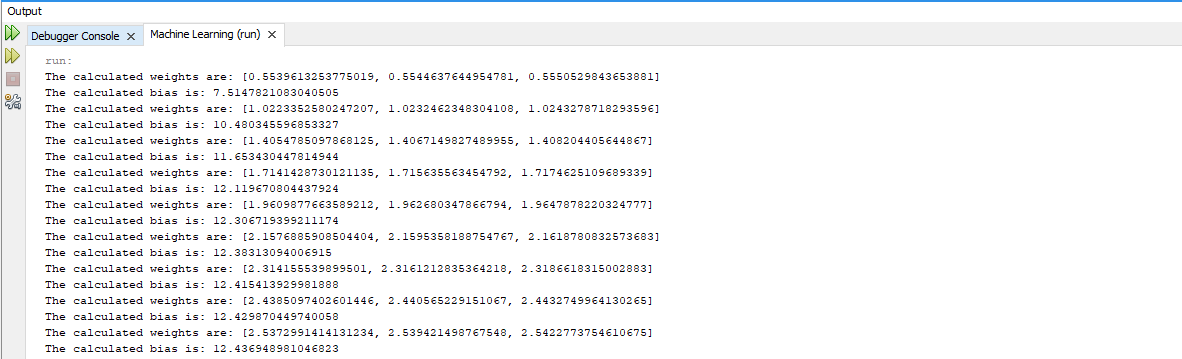
My primary focus when I initially began implementation of my main class was to get a working gradient descent algorithm, which I did. I implemented the code for the Mean Squared Error Derivative, as shown on page 1 under Formulas, using two for loops that could iterate through the input features array list. As the algorithm calculated the Mean Squared Error Derivative for each weight and bias, it would also update the value. As it runs, the algorithm prints the calculated weights and bias for each iteration. The main problem I encountered when implementing this algorithm was an IndexOutOfBoundsException from the for loops and though it did take me some troubleshooting, I was able to correct the error and get the algorithm to work.

My next goal was to implement a method that could test the weights and bias received from my algorithm. Like the gradient descent algorithm, I implemented two for loops to achieve this, but instead of calculating the Mean Squared Error Derivative I multiplied the feature times the calculated weight and added the bias. Each iteration of this method outputs the prediction calculated and also stores it in an array list, which is then returned to the test class. I also ended up encountering the same IndexOutOfBoundsException problem when implementing this method but was able to correct it.

After successfully completing my gradient descent algorithm and test method, I expanded my class to also include a method to solve Mean Absolute Percentage Error, formula shown above. This method would be able to take my calculated value and see if the error rate of my algorithm was egregious or not. I did this using a single for loop and a couple lines of code that would implement the formula for Mean Absolute Percentage Error and pass in the calculated and actual values for my target.

Output

My program outputs the calculated weights and bias as the gradient descent algorithm runs, the calculated predictions as the test method runs, and the Mean Absolute Percentage Error at the end of the program from its respective method. As there are hundreds of values in my set, I cannot show all of the outputs so instead, an example of the outputs are shown below.



Graphical user interface, text, application, email

Description automatically generated

Text

Description automatically generated with low confidence

Instructions to run program

To run my program on your own machine, download my FuelTest and FuelProject programs along with my 4 data sets. The only modification that is needed is to change the file paths for each data set in each BufferReader in the FuelTest file. A picture containing diagram

Description automatically generated

This part

Final Thoughts

In my opinion, the hardest part to this project was finding a good data set that had some correlation between the features that I wanted and the target values. At first, I chose a data set that I was very interested in, but its features resulted in a very high error rate due to low correlation. Because of this, I had to choose a different data set that had more correlation and though I was not as interested in this data set, it was nice to see my gradient descent algorithm output a much small error rate. I definitely believe that this project helped me get a deeper understanding to how gradient descent and linear regression works, especially with implementing the mean squared error formula into code.