Software Design Project 1 Report

In this project, I was tasked with doing an exploratory analysis of a specific dataset and preparing it for predictive analysis of an intelligent system. The goal is to get the highest accuracy percentage for the predictive analysis and fit it to a Linear Regression model. The purpose of this project was to apply my skills on how to best prepare data for ease of user and responsible system manipulation to eventually generate accurate predictive results.

Exploratory Analysis

For this project, I was given a dataset with multiple characteristics about 398 cars. A couple of these characteristics were things such as the mileage (mpg), acceleration, and origin. When first importing this dataset into my notebook I noticed most of the data types were integers or floats which made me very happy because the more numerical types the better for data manipulation!

Although all is not perfect, as I look into the dataset I see that the horsepower is numerical values from using the *head()* but with further investigation with *info()* I was surprised to see that it was an object category type. The values seemed fine but when I tried to convert it to an integer type I was prompted that I couldn't because you cannot convert strings into ints, and apparently there was a '?' somewhere In the horsepower column. My initial instinct was to try and look for the element using *startswith()* or *endswith()* functions but it didn't seem to have any spaces before or after it. I eventually wrote a for loop to go through each row and look for the value(s) and change the '?' to a 0 instead.

I covered all the basic bases, such as null values, duplicates, unnecessary columns, and even one-hot encoding!! There weren't any duplicates in this set nor much unnecessary information but the origin column did seem a bit excessive with having 3 values when one-hot encoding felt a lot more efficient. Of course, I converted the column into a category and from there "removed" the first category to complete the one-hot encoding by using *pd.get_dummies()*.

Uni/Multivariate Analysis Plotting

After preparing my data I performed three plots, two Univariate *boxplots* of the horsepower and acceleration columns from the dataset, and a multivariate *heatmap* of the mpg, displacement, horsepower, weight, and acceleration.

After preparing my data I looked into the descriptions and noticed the standard deviations of the horsepower and acceleration had respectively larger values than the other columns giving me the impression that they would have outliers. This led me to investigate these columns to check my predictions, and after plotting them there are indeed outliers.

Whereas with the heatmap I got some insight into how each of the columns correlates with one another. The displacement, horsepower, and weight correlated with each other relatively well which makes sense realistically seeing for example bigger cars (more weight) tend to need more horsepower. Even then with this connection, I also saw that the mileage doesn't seem to correlate well at all with most of the other columns, which led me to infer that it isn't because they don't affect the mileage because they do but that they just "negatively" affect it. Going back to my earlier example, bigger cars also use more gas than smaller cars.

Fit Regression

Seeing as I was asked to investigate what affects the fuel efficiency of cars I need to train my model and test its accuracy. To train it in order to fit it I used the *train_test_split* function from Sklearn. Seeing as the more data the better the predictions I made sure to include as many possible affecting factors, for this dataset I simply used every column except for the mileage and car names.

In this case the Y is set to be the mileage and the X is the affecting factors. After training, testing, and fitting the data I made my prediction variable (*predict()*) based on the Y values seeing as I want to predict what the horsepower, displacement, and etc. will be for a specific mileage.

From there I was able to check my accuracy scores using the *score()* And compare both the training and testing values!! My accuracy scores were both in the 80 range, being 91% and 85% respectively.

I am relatively confident in my model seeing as with all intelligent systems we can't depend 100% on them to get things right all the time, at least not yet. So there is always the conscious factor of understanding technological error or even just random error and being aware that for every result it is smart to keep in mind personal knowledge of current fuel efficiency trends. Just because as a Jeep owner myself, my car drinks a lot of gas so I have bad mileage but seeing as I don't drive very far when I use it my mileage compared to other jeeps is better on average.