Ajay Iyer

DAT 402

Project 2: Diamond Price Prediction Model

The dataset I chose for this project is Diamond Price Analysis dataset from Kaggle by Shivam Agrawal. This is something similar to what was done in the first project, but I also wanted to see how accurate the model would be when there are more than 300 observations which was not in the first project. After reading in the csv dataset in Python, I knew I wanted to do is look for a predictive model that will predict the price of the diamond based on the features in the dataset. I wanted to use Linear Regression with subset selection and logistic regression and compare the results to see which one would predict the price better. The first method/model was Linear Regression with subset selection. The first thing I did was remove the index column of the dataset since it was not a feature/column I needed. After dropping that column, I checked for any NA values in the dataset and there were none so there was not much I had to do there. Then, I renamed some of the columns so it would be easier to know what some of the columns stand for. I changed the depth column to depth percent which was the total depth percentage. Same thing was done for table which was percentage of the diamond’s average diameter. The other columns I changed were x, y, and z since no one would know what those were. Those variables were changed to length, width, and depth which would be easier to understand. Then, I used the label encoder to change some of the values that were categories to numbers so I could use them in the model. The label encoder was used for the cut, color, and clarity columns/features of the dataset. Next, I did not like where the price column was which was in the middle of the dataset. I wanted it either in the first or the last column so it would be easier to check. Then, I checked to make sure that all the values in each column were numbers and they had to be either float or int so they could be used for the model. Then, I saved the updated dataset so I could do subset selection in R. After reading the updated dataset into R, I wanted to find the best model using the exhaustive selection method. I wanted to find the best model with 8 features/predictors for the price thinking there had to be one that was not that useful. When trying to find the best features, I found a better model with only 7 features, and I decided to go with it. The features I got were carat, cut, color, clarity, depth percent, table percent, and length. Next, I did a summary of the model which found the coefficients and some statistics of the model. The multiple r-squared is 0.8851 while the adjusted r-squared is 0.885. This means that there is 88.50% variation statistically and 90% or higher would have been better but this will still work. Next, I went into Python again so I could remove what features I did not need to perform the train-test split. I performed the train test split with the test size being 20% so I could see if the data was a good fit for the model and I got an accuracy score of 0.887056 which is right around what I thought it would be after retrieving the adjusted r squared. I printed the intercept and coefficients to see if it was right around what I got when I got the summary of the model counting the error in R. After comparing both equations, I was getting something close accounting for the error which was a good sign saying I am on the right track. I used the testX data to predict what the output would be, and it was close to what the actual output. There were some predictions that went higher, and some went lower than the price listed in the data. Next, I took that updated dataset and read in R so I plot the bias-variance tradeoff and see how that looks for the model. After going back to R one last time, I used the KNN algorithm to plot the bias-variance tradeoff and that made 2 plots which were looking at k and Complexity of k vs outRMSE and inRMSE. I found that as the complexity increases, the graph was also increasing meaning there was high variance which is not what we desired for it to be in the model. After that I used Python to calculate the variance and bias, and the result was that the variance was around 13,000,000+ which is ridiculously high, and the bias was 2,000,000+. With such high variance, we can say that it does fit the data does fit but it might be underfitting to what we wanted when looking at the plot.

<https://www.kaggle.com/datasets/shivam2503/diamonds>