CIS 662: Assignment 2

Instruction: This is an individual assignment. You are welcome to look anything up online but do NOT collaborate with your classmates or get significant help from anyone outside. The due date can be checked on Blackboard and is approximately a week from when this is assigned. You need to respond to all the questions on Blackboard. Also, upload to the online portal both the Jupyter notebook as well as PDF versions of the python code with run outputs visible (you may want to write a long code so that the various parts are in a single file). Check to see that when you print to PDF, parts of the code do not get cut off.

1. **Get data and pre-process**: use the CSV file `diamonds.csv` for this assignment. The dataset is obtained from Kaggle. As the name suggests, the dataset is a collection of information about diamonds including the *price* which we will use as a dependent variable (i.e. Y), and most of the remaining columns as features. Here is a description of the columns:
   `rownames`: Just an index column and will not be used as a feature;
   `price`: The dependent variable denoting the price of that diamond in US dollars;
   `carat`: weight of the diamond;
   `cut`: quality of the cut (Fair, Good, Very Good, Premium, Ideal);
   `color`: diamond color, from D (best) to J (worst);
   `clarity`: a measurement of how clear the diamond is (I1 (worst), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (best));
   `x, y, z`: length, width, and depth respectively, all in mm;
   `depth`: total depth percentage $= z/mean(x, y) = 2 * z/(x + y)$;
   `table`: width of top of diamond relative to widest point.
   As a first step, read the CSV file into a data frame in your python code.

   (a) Using the CSV file you read into your python code, check it looks good and see the names of the columns, display the first five lines of the file. Then check the column data types.

   (b) Drop the column 'rownames' as it is simply an index of the diamonds. Make sure to check afterwards that it got dropped.

   (c) Next, check if there are any NaN values in any of the files.

   (d) How many rows of data (n) are in the data frame?

   (e) What are the set of values taken by the three categorical variables in the data set?

2. **Simple Linear Regression**: This is similar to what we did in class for SLR.

   (a) Recall that for SLR we use just a single feature (X or independent variable). Create a heat map and say which of the quantitative (i.e. not categorical) features has the highest correlation with *price*.

   (b) Split the data (with just two columns, i.e. the single independent variable and the dependent variable) into train and test by randomly selecting 25% of the data to be in the test set with 'random_state' of 50. *Write down the length of the train and the test data. (as well as the first row's index for both).*

(c) Perform a simple linear regression using the train data. *In Blackboard write down* R^2 *and* $\hat{\beta}_0$. *Comment on fit, and also on what the coefficients are indicating.*

(d) Create a dataframe to predict the *price* for the test set using the independent variables. Also include the ground truth (actual *price*). *Write down the mean absolute error (MAE), and also the fraction of MAE to the average* price *seen in the test set (this gives an idea of how good the predictions were). Based on how good the fit was, did you expect the errors to be so?*

3. **Multiple Linear Regression**: This is similar to what we did in class without using categorical variables.

(a) For the independent variables, use all six numerical columns and no categorical variables. For the dependent variable continue to use *price*. Split the data into train and test by randomly selecting 25% of the data to be in the test set with 'random_state' of 50. *Did the length of the test and train data change from SLR? What about the first row's index for both, did they remain the same?*

(b) Perform a multiple linear regression using the train data. *In Blackboard write down* R^2 *and* $\hat{\beta}_0$. *Comment on fit, and whether having more features improved the fit compared to SLR.*

(c) Create a dataframe to predict the *price* for the test set using the independent variables. Also include the ground truth (actual *price*). *Write down the mean absolute error (MAE), and also the fraction of MAE to the average* price *seen in the test set (this gives an idea of how good the predictions were). Did the predictions in the test set improve compared to SLR?*

4. **Multiple Linear Regression with Categorical Variables**: This is similar to what we did in class, this time using the categorical variables.

(a) For the independent variables, besides the 6 columns considered earlier, also use the three categorical variables. *Display the first five rows of the data with the independent variables and the dependent variable.*

(b) Do a one-hot encoding to convert the three categorical variables into coded binary variables. *How many columns are in the eventual set? Can you explain how we went to that many columns?*

(c) Split the data into train and test by randomly selecting 25% of the data to be in the test set (again use 'random_state' of 50). Perform a linear regression using the train data. *Write down* R^2 *and* $\hat{\beta}_0$, *and in the code output display* $\hat{\beta}_1$, ..., $\hat{\beta_m}$ *for the m features. Comment on fit, and also on what the coefficients are indicating.*

(d) Create a dataframe to predict the *price* for the test set using the independent variables. Also include the ground truth (actual *price*). *Write down the mean absolute error (MAE), and also the fraction of MAE to the average* price *seen in the test set (this gives an idea of how good the predictions were). Based on how good the fit was, did you expect the errors to be so? Did the test predictions improve when we added the new variables? Was it significant in your opinion?*

5. **Quantile Regression with Categorical Variables**: In SLR and multiple regression our goal was to predict the expected value (or mean) of the dependent variable. Here we switch our attention to obtain the median instead of the mean as it is a more robust estimate of the middle value especially not getting affected by outliers. We will then use that median prediction and compare against the ground truth in the test data. For the independent variables, besides the 6 columns considered earlier, also use the three categorical variables. This is the same as in the most recent case. Do a one-hot encoding to convert the three categorical variables into coded binary variables. Split the data into train and test by randomly selecting 25% of the data to be in the test set (again use 'random_state' of 50). Up until this stage, it is identical to the previous case.

   (a) While there is a quantile regression function in `sklearn`, as an opportunity to study a different API, run the following:

   ```
   import statsmodels.formula.api as smf
   ```

   if needed by first running a pip install. Then run the following:

   ```
   mod = smf.quantreg("insert model here")
   res = mod.fit(q=0.5)
   ```

   after reading about quantile regression in `statsmodels`. Note that in the above lines you will actually need to put down the relation between the dependent variables and the independent variables. Then $q = 0.5$ is used to get the 50th percentile, i.e. median. *In the code display the model you used.*

   (b) Create a dataframe to predict the *price* for the test set using the independent variables and suitably multiplying with the quantile regression parameters (use `res.params[]` to get the parameters, the documentation has these). Also include the ground truth (actual *price*). *Write down the fraction of MAE to the average* price *seen in the test set (this gives an idea of how good the predictions were). Did the test predictions improve when we use median (i.e. quantile regression)?*