**Project 4: Regression Analysis**

**Team Members:**

**Rachel Menezes: 905508013**

**Diplav : 605627748**

**Evyn Chiappe : 605201321**

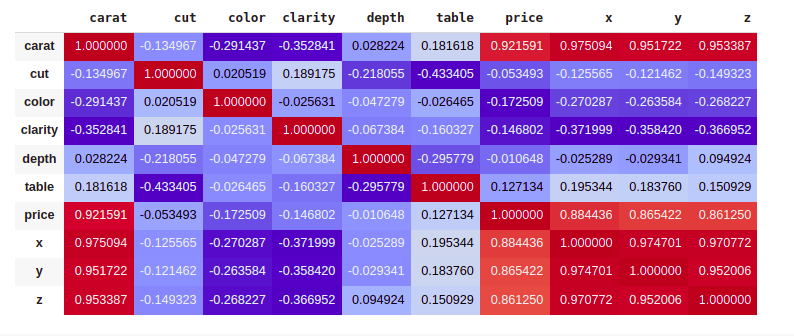
We decided to work on the **Diamonds** dataset

Prior to question 1.1 the cut, color and clarity columns which are currently categorical are given numeric values. The code for the same can be seen in the notebook.

**QUESTION 1.1:**

Plot a heatmap of the Pearson correlation matrix of the dataset columns. Report which features have the highest absolute correlation with the target variable. In the context of either dataset, describe what the correlation patterns suggest.

**ANS:**

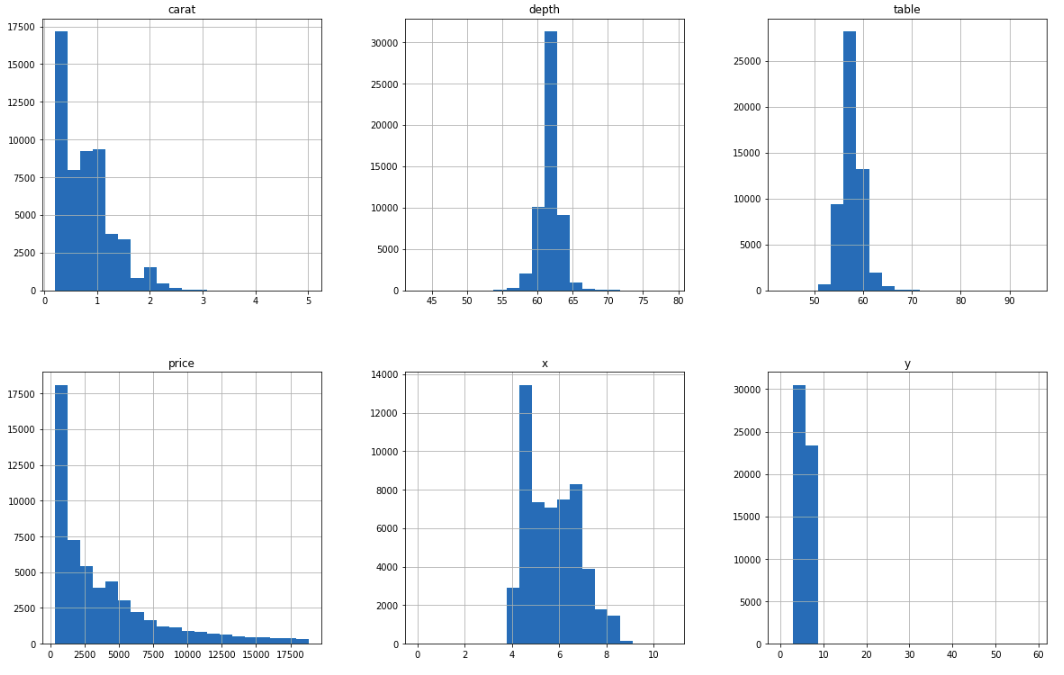


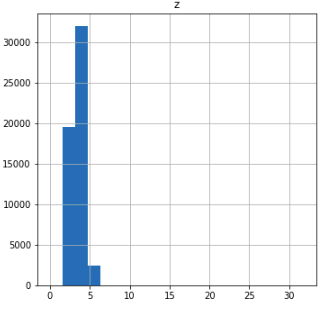
Price is the target variable if we look at the row/column of the price variable we can see the features that have the highest absolute correlation with price are carat(0.921591), and then x(0.884436), y(0.865422) & z(0.861250). The other features have an absolute correlation coefficient of less than 0.2 with respect to price  
  
Carat is the weight of the diamond & x,y,z are the dimensions and the larger the diamond dimensionally and in weight the more expensive it is makes sense.

**QUESTION 1.2:**  
Plot the histogram of numerical features. What preprocessing can be done if the distribution of a feature has high skewness?

**ANS:**

The histogram of the numerical features, i.e., 'carat', 'depth', 'table', 'price', 'x', 'y', 'z' are plotted.  
We see that the histograms of depth and x are centrally distributed. The histograms of carat, table, price, y and z are more right skewed.





In regression modeling, features that exhibit high skewness are undesirable because many data analysis tools, such as regression analysis, require the dependent variable's distribution to be Gaussian. Models are more likely to converge when the features follow a Gaussian distribution. When using squared error to train the model, parameter estimation on skewed distributions can lead to disproportionate influence on the parameter estimates.

To eliminate skewness, one can perform several transformations on the dataset, including log transformation for right-skewed data (a strong transformation) and left-skewed data (a weak transformation) (excluding values of 0), square root transformation for right-skewed data (a weak transformation) and left-skewed data (a strong transformation), Box-Cox transformation, cube root transformation, reciprocal transformation (excluding values of 0), and transformations combined with min-max or z-score normalization and scaling. Removing outliers can also help by removing points that are far from the distribution's mean.

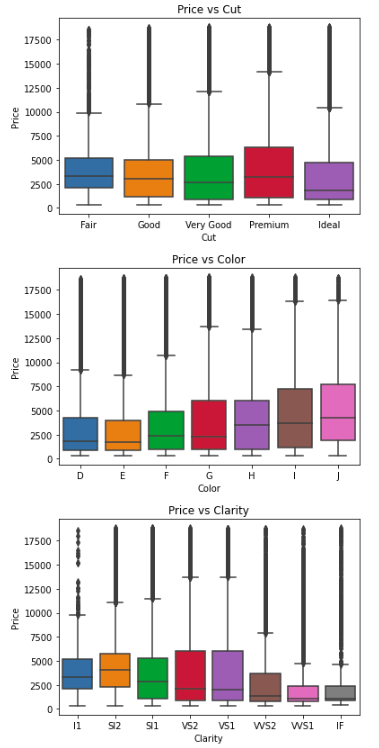
**QUESTION 1.3:**  
Construct and inspect the box plot of categorical features vs target variable. What do you find?  
  
**ANS:**

Below are the box plots for price versus the categorical features (cut, color, and clarity).

For price versus cut, the median value is similar across all categories except for "ideal," which has a slightly lower value. The maximum price for each category is roughly the same. Data points for each category exhibit a right-skewed distribution, and there are outliers at one end of each category.

In the case of price versus color, the median value varies across categories, with color J having the highest median price. Median price increases from category D (best) to category J (worst). The maximum price for each category is roughly the same. Data points for each category exhibit a right-skewed distribution, and there are outliers at one end of each category.

For price versus clarity, the median value decreases as clarity increases. The maximum price for each category is roughly the same. Data points for each category exhibit a right-skewed distribution, and there are outliers at one end of each category.

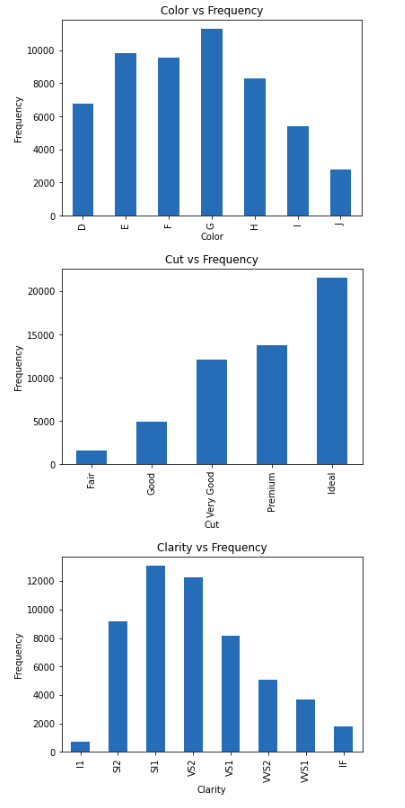


**QUESTION 1.4:**

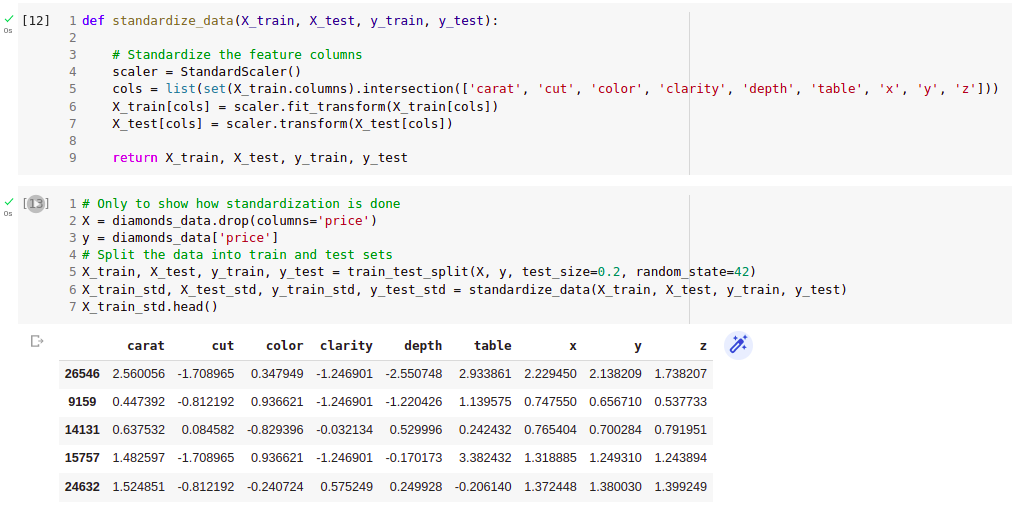
For the Diamonds dataset, plot the counts by color, cut and clarity.

**ANS**:

Plots for the counts by color, cut and clarity are shown below:



**QUESTION 2.1:**Standardize feature columns and prepare them for training.

**ANS:**While there is a separate function written to standardize across all 10 folds in thw pipeline. Here is a snippet showing standardiation of feature columns.  
  


**QUESTION 2.2:**

**sklearn.feature selection.mutual\_info\_regression** function

returns estimated mutual information between each feature and the label.

**sklearn.feature selection.f\_regression** function provides F scores

You **may** use these functions to select features that yield better regression results (especially in the classical models). Describe how this step qualitatively affects the performance of your models in terms of test RMSE. Is it true for all model types? Also list two features for either dataset that has the lowest MI w.r.t to the target.

**ANS:**Feature selection involves selecting a subset of features from the original set of variables that are most relevant to the target variable.

By selecting the most informative features, we can reduce the noise in the data and improve the generalizability and interpretability of the model. This step can significantly affect the performance of the models in terms of test RMSE. Typically, selecting informative features leads to a decrease in test RMSE because the model is trained on a more relevant and less noisy set of features. However, in some cases, selecting too many features can lead to overfitting and an increase in test RMSE.

This step is not necessarily true for all model types. For example, some models such as decision trees and random forests can handle irrelevant features without compromising their performance. However, in classical models such as linear regression, Lasso regression, and Ridge regression, feature selection can have a significant impact on performance.

For the diamond characteristics dataset, two features that have the lowest MI w.r.t the target are "table" and "depth".

The graph below shows the test RMSE vs the top k features for (linear, lasso, and ridge regression) using F-Score and mutual information. For the diamond characteristics dataset we see that test RMSE improvement is not a lot after k=6, F-Score and MI converge to similar test-RMSE values when k equals 1, 3, 4, 6, 8, and 9. When k equals 2, F-Score outperforms MI for all three models, while MI outperforms F-Score when k equals 5 and 7.



**QUESTION 3:**

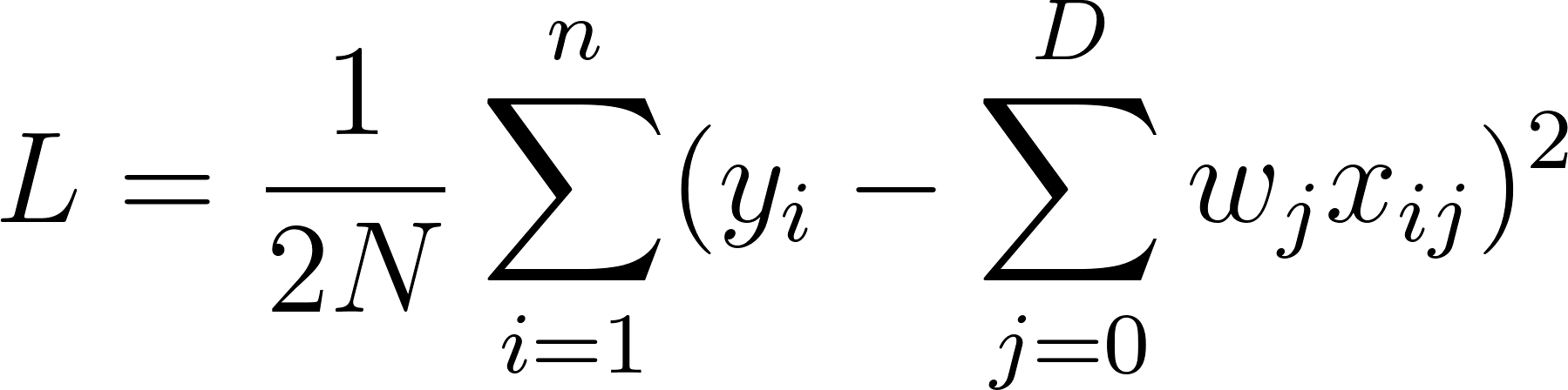
For random forest model, measure “Out-of-Bag Error” (OOB) as well as explain what OOB error and R2 score means given this link.  
  
**ANS**:  
The Out-of-Bag (OOB) error is a way to estimate the performance of a random forest model without the need for a separate validation dataset. The OOB error is calculated by training each tree in the forest using a bootstrap sample of the original dataset, leaving out around one-third of the data. This left-out data is called the OOB sample. The OOB error is then calculated by predicting the OOB samples using the trees that were not trained on them and calculating the mean squared error between the predicted and actual values. The OOB error gives an estimate of the generalization error of the model.

The R2 score is a metric used to evaluate the performance of regression models. It measures the proportion of variance in the target variable that is explained by the model. An R2 score of 1 indicates that the model explains all the variance in the target variable, while an R2 score of 0 indicates that the model does not explain any variance.

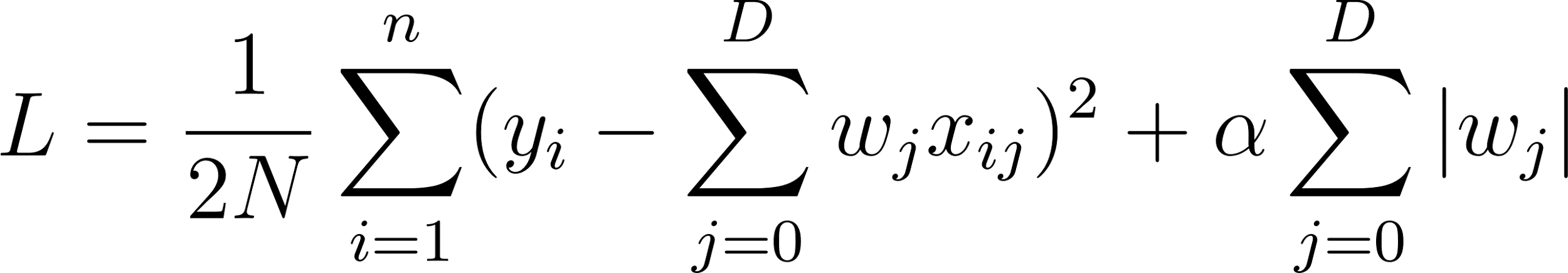
**3.3.1 Linear Regression**

What is the objective function? Train three models: (a) ordinary least squares (linear regression without regularization), (b) Lasso and (c) Ridge regression, and answer the following questions.

The objective function in linear regression is to minimize the sum of squared errors between the predicted values and the actual values of the target variable. This is achieved by finding the best set of parameters that minimize the difference between predicted and actual values.

[](https://www.codecogs.com/eqnedit.php?latex=L%20%3D%20%5Cfrac%7B1%7D%7B2N%7D%20%5Csum%5Climits_%7Bi%3D1%7D%5E%7Bn%7D(%7By%7D_i-%5Csum%5Climits_%7Bj%3D0%7D%5E%7BD%7Dw_jx_%7Bij%7D)%5E2#0)

In Lasso regression, the objective function adds a penalty term equal to the sum of absolute values of the coefficients multiplied by a constant alpha. This penalty term encourages sparse solutions by driving some of the coefficients to zero, which can help with feature selection and prevent overfitting.

[](https://www.codecogs.com/eqnedit.php?latex=L%20%3D%20%5Cfrac%7B1%7D%7B2N%7D%20%5Csum%5Climits_%7Bi%3D1%7D%5E%7Bn%7D(%7By%7D_i-%5Csum%5Climits_%7Bj%3D0%7D%5E%7BD%7Dw_jx_%7Bij%7D)%5E2%20%2B%20%5Calpha%20%5Csum%5Climits_%7Bj%3D0%7D%5E%7BD%7D%7Cw_j%7C#0)

In Ridge regression, the objective function adds a penalty term equal to the sum of squared values of the coefficients multiplied by a constant alpha. This penalty term also encourages sparse solutions, but it doesn't drive the coefficients all the way to zero. Instead, it shrinks them towards zero, which can help with preventing overfitting and improving the model's generalizability.

[](https://www.codecogs.com/eqnedit.php?latex=L%20%3D%20%5Cfrac%7B1%7D%7B2N%7D%20%5Csum%5Climits_%7Bi%3D1%7D%5E%7Bn%7D(y_i-%5Csum%5Climits_%7Bj%3D0%7D%5E%7BD%7Dw_jx_%7Bij%7D)%5E2%20%2B%20%5Calpha%20%5Csum%5Climits_%7Bj%3D0%7D%5E%7BD%7Dw_j%5E2#0)

**QUESTION 4.1:**Explain how each regularization scheme affects the learned parameter set.

**ANS:**

Regularization schemes affect the learned parameter set by adjusting the trade-off between fitting the data well and preventing overfitting. In linear regression without regularization, the model can fit the data exactly, but it may not generalize well to new data. In Lasso regression, some of the coefficients are driven to zero, leading to a simpler model with fewer features that may be easier to interpret and generalize. In Ridge regression, the coefficients are shrunk towards zero, leading to a smoother model with less variance and potentially better generalization performance.

**Table Compiling all results**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **Standardized** | **Feature Selection Scheme** | **Optimal penalty** | **Train RMSE** | **Validation RMSE** |
| Linear Regression | Yes | F-regression | N/A | -1221.6042206336504 | -1228.2203041711432 |
| MI -regression | N/A | -1221.6042206336504 | -1228.2203041711432 |
| Lasso Regression | Yes | F-regression | 1e-05 | -1221.6042206336576 | -1228.2203123582412 |
| MI -regression | 1e-05 | -1221.6042206336576 | -1228.2203123582412 |
| Ridge Regression | Yes | F-regression | 1e-05 | -1221.6042206336504 | -1228.2203047136793 |
| MI -regression | 1e-05 | -1221.6042206336504 | -1228.2203047136793 |
| Ridge Regression | No | F-regression | 1e-05 | -1221.6042206336504 | -1228.2203070191495 |
| MI -regression | 1e-05 | -1221.6042206336504 | -1228.2203070191495 |

**QUESTION 4.2:**Report your choice of the best regularization scheme along with the optimal penalty parameter and explain how you computed it.

**ANS:**

Looking at the table above we can see that the validation scores are extremely close on comparing to the maximum precision Ridge Regression with an optimal penalty of 1e-05 seems to have the lowest validation error.

**QUESTION 4.3:**Does feature standardization play a role in improving the model performance (in the cases with ridge regularization)? Justify your answer.

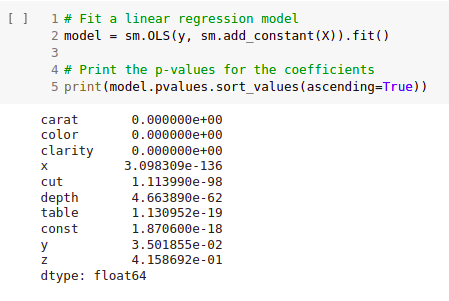
**ANS:**

Based on the provided results table, we can see that for both F-regression and mutual information regression feature selection methods, the RMSE values for the ridge reegularization model with and without standardization are very close to each other for both the training and validation datasets.

Feature standardization did not help in the ridge regression case for the diamonds dataset because the scale of the features was already relatively similar. Ridge regression penalizes the sum of squared weights, so if there is no significant difference in the scale of the features, the regularization effect will be spread out more evenly among the features, and the effect of feature scaling on the model will be limited.

**QUESTION 4.4:**Some linear regression packages return p-values for different features. What is the meaning of these p-values and how can you infer the most significant features?

**ANS:**

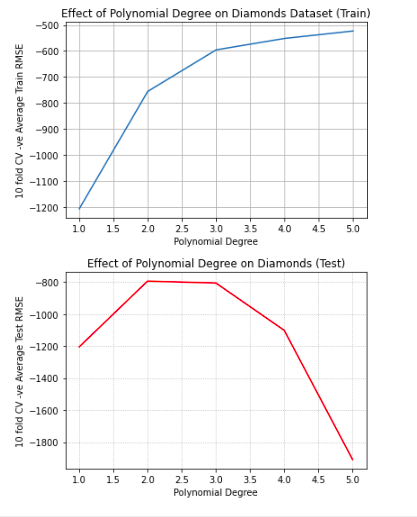
****

In some linear regression packages, such as the statsmodels python package, p-values are returned for different features. These p-values indicate whether we can reject or fail to reject the null hypothesis that the feature has no correlation with the target variable at the population level. A high p-value (> 5% or > 0.05) means that there is insufficient evidence to find a correlation between the feature and the target variable, while a low p-value (< 5% or < 0.05) indicates that the feature is significant and contributes to changes in the target variable.

To determine the most significant features, we look for the features with the lowest p-values, as they are most likely to have a strong correlation with the target variable. In the diamonds dataset, the top 5 significant features based on their p-values are carat, color, clarity, x, and cut, as they all have p-values of 0.05 or less. These results partially match the salient features obtained from the heatmap in Question 2, with some differences possibly due to the influence of categorical variables over numerical variables. However, it's worth noting that the p-values alone do not necessarily determine the most significant features. Other factors, such as the correlation between features, should also be taken into consideration.

**3.3.2 Polynomial Regression**

Perform polynomial regression by crafting products of features you selected in part 3.1.4 up to a certain degree (max degree 6) and applying ridge regression on the compound features. You can use scikit-learn library to build such features. Avoid overfitting by proper regularization. ANSwer the following:



**QUESTION 5.1:**What are the most salient features? Why?

**ANS:**Based on the Ridge regression model with polynomial features, the top salient features (in descending order of significance) for predicting the diamond price are:

‘carat clarity’, ‘carat color’, ’carat y’, ‘carat x’, ‘carat z’, ‘color clarity’, ‘carat cut’, ‘depth x’,’x’. These features are deemed most significant based on their corresponding coefficients in the Ridge regression model. The higher the absolute value of the coefficient, the more influential the feature is in determining the target variable (price).

**QUESTION 5.2:**What degree of polynomial is best? How did you find the optimal degree? What

does a very high-order polynomial imply about the fit on the training data? What about its performance on testing data?

**ANS:**To determine the optimal degree of polynomial for the diamond dataset, we created a pipeline that uses PolynomialFeatures, FoldStandardScaler(standardization function), and Ridge Regression. The pipeline takes a range of degree values (1 to 5) and alpha values (10^-5 to 10^5) as parameters for PolynomialFeatures and Ridge Regression respectively.

Using GridSearchCV with 10-fold cross-validation, we trained the pipeline on the diamond dataset and evaluated its performance based on the negative root mean squared error (RMSE) score. The best parameters were found to be a **degree of 2** and an alpha value of 1000.0. The pipeline achieved a train RMSE of -523.151557853977 and a test RMSE of -794.164314834404.

A very high-order polynomial may fit the training data very well, but it may also result in overfitting. Overfitting occurs when the model is too complex and has learned noise or random fluctuations in the training data, which does not generalize well to the testing data. As a result, a high-order polynomial model may perform poorly on the testing data, even though it fits the training data well. It is important to strike a balance between the model's complexity and its ability to generalize to new data, which is why we used cross-validation to determine the optimal degree for our polynomial regression model.

**3.3.3 Neural Network**

You will train a multi-layer perceptron (fully connected neural network). You can simply use the sklearn implementation:

**QUESTION 6.1:**Adjust your network size (number of hidden neurons and depth), and weight decay as regularization. Find a good hyper-parameter set systematically (no more than 20 experiments in total).

**ANS:**To evaluate the performance of MLP for this task, we experimented with various hyperparameters such as weight decays, network sizes (hidden neurons and depth), and model activation functions. We used the top 6 features selected by F-Score (MI selected features were the same). The hyperparameter space included:

Network sizes: [[32],[64],[8,16,32,64]]. We kept the number of network sizes small to reduce training time. We used 32 and 64 deep MLP to analyze the effect of increasing the number of neurons in a layer and 32 and [8,16,32,64] to analyze the effect of increasing the number of layers in MLP.

Model activation: relu, tanh, logistic

Weight decay (for regularization): 0.001, 0.01, 0.1

We performed a 10-fold Grid Search Cross-validation using the pipeline() function of Scikit learn to systematically find the optimal hyperparameters for the diamond characteristic dataset. The best set of hyperparameters for this dataset was achieved with activation as **'relu'**, alpha as **0.001**, and layer\_structure as **[8, 16, 32, 64]**. The resulting train and validation RMSE for this setting were **-684.873 and -651.281**, respectively.

**QUESTION 6.2:**How does the performance generally compare with linear regression? Why?

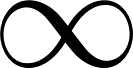
**ANS:**The test RMSE of the best linear regression model was -1228.22030, while the test RMSE obtained by the MLP model with best hyperparameters is -651.28062.

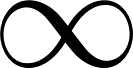
This indicates that the MLP model performs significantly better than the linear regression model. This is due to several reasons, such as the MLP model having a larger number of trainable parameters, which provides it with more capacity to learn information from the data. The MLP model can also capture complex mathematical relationships via hidden layer connections, which is not an option in linear regression. Additionally, neural networks can capture non-linear relationships since they have non-linear activation functions, unlike linear regression. Furthermore, MLPs can automatically extract the most important non-linear features without requiring explicit transformations or human intervention into the data. This is a capability that is not available in linear regression.

However as a con, it is worth noting that neural networks can be more computationally expensive and may require more careful tuning of hyperparameters compared to linear regression. Additionally, neural networks can be more prone to overfitting if not properly regularized.

**QUESTION 6.3:**

What activation function did you use for the output and why? You may use none.

**ANS:**The appropriate activation function for the output variable depends on the range of the output values and the range of the activation function. ReLU activation function outputs values in the range [0,[](https://www.codecogs.com/eqnedit.php?latex=%5Cinfty#0)], while tanh outputs values between -1 and 1, and sigmoid / logistic outputs values between 0 and 1.

As the output variable in this case is price, which has a range of [0,[](https://www.codecogs.com/eqnedit.php?latex=%5Cinfty#0)], we require an activation function whose range matches the desired range of the output. Therefore, we should use ReLU activation function for the output as it produces values within the desired range.

**QUESTION 6.4:**What is the risk of increasing the depth of the network too far?

**ANS:**Increasing the depth of a neural network beyond a certain point can lead to overfitting, which is a common problem in machine learning. Overfitting occurs when the model learns the noise in the data instead of the underlying patterns, resulting in poor performance on new, unseen data. Deep neural networks with too many layers can also suffer from the vanishing gradient problem. This occurs when the gradient signal becomes too small as it backpropagates through many layers, making it difficult for the network to learn and update the weights properly. Another issue with deeper networks is that they require more computational resources and training time, which can lead to longer training times and higher hardware costs.

**QUESTION 9:**

**About the Data:**

**9.1 Report the following statistics for each hashtag:**

**ANS:**

**Average number of tweets per hour:**

Average number of tweets per hour #goHwaks: 292.09326424870466

Average number of tweets per hour #gopatriots: 40.888695652173915

Average number of tweets per hour #nfl: 396.97103918228277

Average number of tweets per hour #patriots: 750.6320272572402

Average number of tweets per hour #sb49: 1275.5557461406518

Average number of tweets per hour #superbowl: 2067.824531516184

**Average number of followers:**

Average number of followers of users posting tweet #goHwaks: 2217.9237355281984

Average number of followers of users posting tweet #gopatriots: 1427.2526051635405

Average number of followers of users posting tweet #nfl: 4662.37544523693

Average number of followers of users posting tweet #patriots: 3280.4635616550277

Average number of followers of users posting tweet #sb49: 10374.160292019487

Average number of followers of users posting tweet #superbowl: 8814.96799424623

**Average number of retweets per tweet:**

Average number of retweets per tweet #goHwaks: 2.01320939913198

Average number of retweets per tweet #gopatriots: 1.408191910169

Average number of retweets per tweet #nfl: 1.534460265554

Average number of retweets per tweet #patriots: 1.78528712884

Average number of retweets per tweet #sb49: 2.5271344411

Average number of retweets per tweet #superbowl: 2.3911895819207

**9.2 Plot “number of tweets in hour” over time for #SuperBowl and #NFL:**

**ANS:**

**Chart

Description automatically generated**

**Chart, histogram

Description automatically generated**

**QUESTION 10:**

**ANS:**

**QUESTION 10:**

**ANS:**

I have performed following task on the provided twitter dataset.

1. Fan Base analysis before, during and after the match.
2. Analyzing the average Sentiment trend during the game.
3. Identifying the Key Moment and Key Player in a Game at different duration
4. Predicting Fanbase from Tweet whether Massachusetts or Washington.

**Fan Base analysis before, during and after the match**

**Describe your task?**

The aim for this task was to find which team has better fanbase in term of volume by analyzing the volume of tweet in support for both the teams at different time interval before, during and after the game.

**Explore the data and any metadata?**

For this we explored tweets\_#gohawks.txt and tweets\_#gopatriots.txt which contains the relevant positive tweets for patriots and hawk team.

The tweets\_#gohawks.txt contains around 169122 tweets while tweets\_#gopatriots.txt contains around 23511 tweets.

**Describe the feature engineering process?**

Both the file are quite large and are thus read one line at a time and processes to get only relevant datafield in a dataframe.

Each tweet is processed to remove the special character, hyperlink and web artifact.

The timestamp was converted from UNIX timestamp to PST for the convenience of understanding.

The tweets are divided according to the start time of super bowl game 2015-02-01 15:00:00-08:00 and end time of super bowl 2015-02-01 20:00:00-08:00 and are divided into 3 sections.

* Tweets before the game started - before 2015-02-01 15:00:00-08:00
* Tweets during the game - between 2015-02-01 15:00:00-08:00

and 2015-02-01 20:00:00-08:00.

* Tweets after the game – after 2015-02-01 20:00:00-08:00

The number of tweets were scaled by number of days on which these tweets were received and are plotted for the different time section below.

Chart, bar chart

Description automatically generated

From the graph it can be observed that number of supporters for hawks are in larger volume as compared to patriots and number of supporting tweets are much higher in volume during the game than before and after the game.

**Analyzing the average Sentiment trend during the game**

**Describe your task?**

Over here I tried to analyze the average sentiment trend during the super bowl game for hawks and patriots supporters. Analyzing the the sentiment score we can infer the key moments in the game, peaks in the average sentiment score that the supporters are quite excited and the team in performing well and drop in the sentiment score indicate that their team is not doing well in the game.

**Describe the feature engineering process?**

For analyzing the average sentiment, we have used two methods. In the first one we have used custom built sentiment analyzer where we first defined positive and negative set of words which includes football and sports specifics slangs and words for better performance. The tweets are first processed to remove the special character, hyperlink and web artifact. The cleaned tweets are then passed by a CountVetorizer to get the frequency of each words.

**Generate baselines for your final ML model?**

After this we basically count the number of positive or negative words present in the tweets to decide the average sentiment score by computing the mean the difference between the positive and negative words present in that tweet to decide the average sentiment score at that particular time point.

The graph below shows the average sentiment for patriot and hawk supporters during the game time.

Chart, histogram

Description automatically generated

We can observe an inverse trend of the average sentiment score for the supporter i.e., when the sentiment score for hawks supporter is high the sentiment score for patriots supporters is lower which is sort of expected.

In the second one we used TextBlob which is lexicon based sentiment analyzer with predefined rules and uses a dictionary kind of thing to define weight for different words and uses it to create average sentiment score. As it uses a predefined rule to compute the score and doesn’t incorporate special rules for sports related slang it doesn’t perform that well. Here we don’t see the inverse trend which was observed in custom defined sentiment analyzer. It fails on sarcasm and some sports related word. We are merely using it for baseline comparison.

Chart, line chart

Description automatically generated

**Identifying the Key Moment and Key Player in a Game at different duration**

**Describe your task?**

By Analyzing the frequency of the tweets we can infer about the key moments in the game. We can observe increase in the frequency of tweets whenever there is an important moment in the game. By analyzing the frequency of the tweet one can assume that key moment of the game occurs at the peak of the frequency distribution of the tweet.

**Describe the feature engineering process?**

We plotted the frequency of cleaned tweet during the game by only considering tweets between 2015-02-01 15:00:00-08:00

and 2015-02-01 20:00:00-08:00. For finding the peak of the frequency distribution we have used find\_peaks function in the sklearn library.

The frequency distribution and corresponding key moment during the game for #gopatriot and #gohawks tweets are plotted below.

Chart, histogram

Description automatically generated

Chart, histogram

Description automatically generated

From the graph, we can observe for both patriots and hawks are mutually exclusive which is inline with our sentiment analysis study indicating the moments which are high point of one teams are low point for the other.

For identifying the key player, we first collected the data for different player in both hawks and patriots from <https://www.sbnation.com/nfl/2015/2/1/7957703/super-bowl-rosters-2015-seahawks-patriots> . Then for both patriot and hawks corresponding to each key moment we found which of the players were mentioned at that time in the tweet and arranged them according to descending order of the number time they were mentioned in the tweet, and the top 5 player at that key moments were identified as the key player.

Below are the key player for patriots at different key moments

Chart, bar chart

Description automatically generated

Chart, bar chart

Description automatically generated

Chart, bar chart, histogram

Description automatically generated

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

Below are the key player for **hawks** at different key moments

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Predicting Fanbase from Tweet whether Massachusetts or Washington**

**Describe your task?**

Over here we aimed to predict weather a tweet is from Massachusetts or Washington. We considered only Massachusetts and Washington as patriot is the home team of Massachusetts and hawks is the home team of Washington. The hashtag used for this task is #superbowl. Once may assume is a tweet is positive about patriots then it is posted from Massachusetts and if it is positive for hawks then it is posted from Washington.

**Explore the data and any metadata?**

GT labels for the tweets were obtained from ['user']['location'] attribute for the tweet and tweet were further filtered on the basis of language and the tweets in English language were only considered.

The ['user']['location'] attribute in the tweet contain the cities and the corresponding state is obtained using geonamecache python library. Tweets corresponding to only WA and MA is considered and rest are filtered out.

In total we have 16574 tweets for MA and 15784 tweets for WA.

We even amount of data point for both the label and there is no class imbalance.

We have used 20% of the total datapoint for testing and remaining 80% for training the model.

**Describe the feature engineering process?**

For generating the feature from tweets we have used TFIDF for each word in the tweet. Tweets were preprocesses to remove digits, stop words, special character, hyperlink and web artifact and then lemmatized to get root word.

There were a total of around 6340 feature per datapoint and thus we used TruncatedSVD to reduce the number of feature. By analyzing the explained variance below we to reduce number of feature from 6340 to 200.

Chart

Description automatically generated

**Generate baselines for your final ML model?**

We trained Logistic Regression and SVC model on our dataset and used grid search to find suitable hyper parameters

Grid Search Parameter for Both Models

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **C** | **Penalty** | **Loss** |
| Logistic Regression | [10^-4, 10^4] | [l1, l2, elasticnet] | Logistic Loss |
| SVC | [10^-3, 10^4] | [l1, l2, elasticnet] | ['hinge', 'squared\_hinge'] |

* **Logistic Regression**

Best Parameter

Penalty – L1, C = 1

Test Accuracy for logistic regression = 0.7456736711990112

Scores for logistic classifier

Accuracy = 0.7456736711990112

Recall = 0.7413389559645515

Precision = 0.766136376284409

F1\_Score = 0.7382772392430639

Chart, treemap chart

Description automatically generated

* **SVC**

Best Parameter

C = 100, Penalty = L2, Loss = squared\_hinge

Test accuracy for SVC classifier = 0.7469097651421508

Scores for SVC classifier

Accuracy = 0.7469097651421508

Recall = 0.7421999372997606

Precision = 0.7719416369220731

F1\_Score = 0.7383590910127489

Chart, treemap chart

Description automatically generated

There seems to be a lot of false positive MA for both classifiers. In term of performance SVC performs slightly better than Logistic Regression.