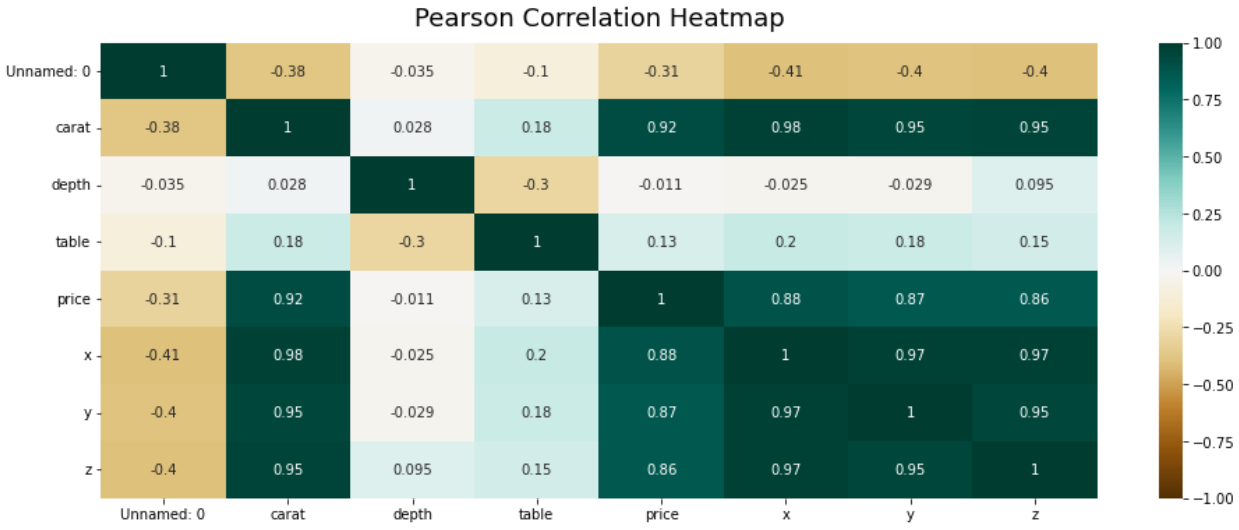
Project 4 Regression Analysis and Define Your Own Task

EECS 219 Winter 2023

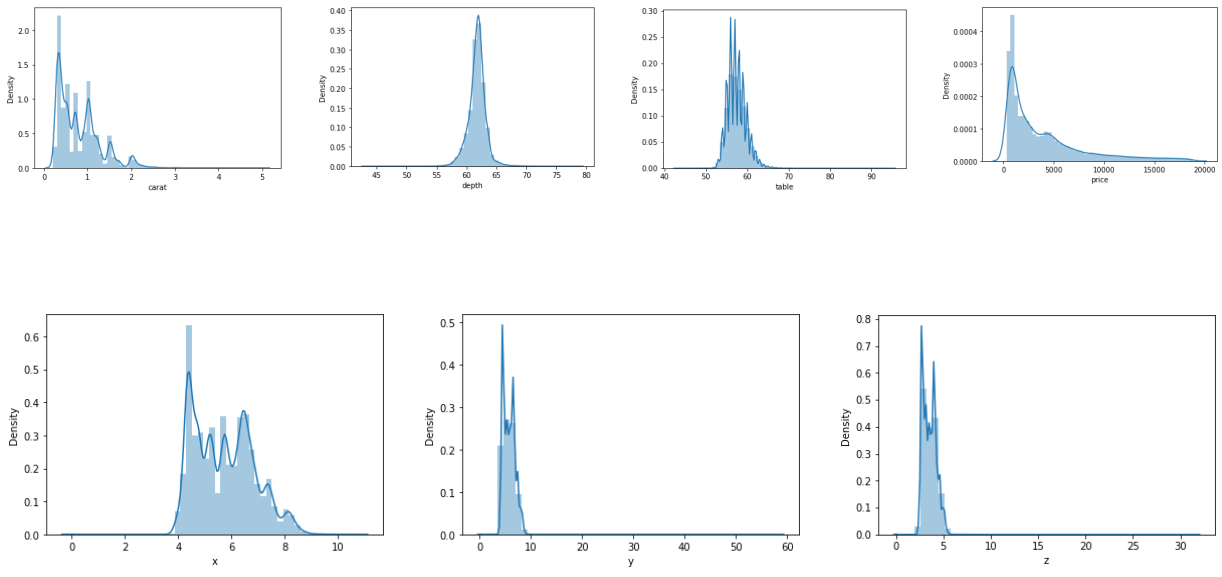
Zheng (Alice) Lu, Fengyuan Heying, Dadian Zhu

Question 1 Data Inspection:

1.1 Heatmap of Pearson correlation matrix:

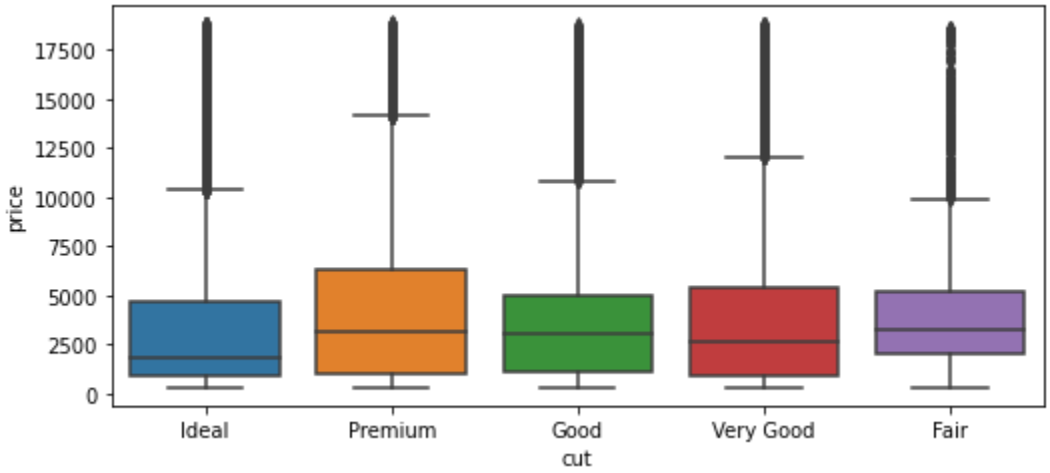


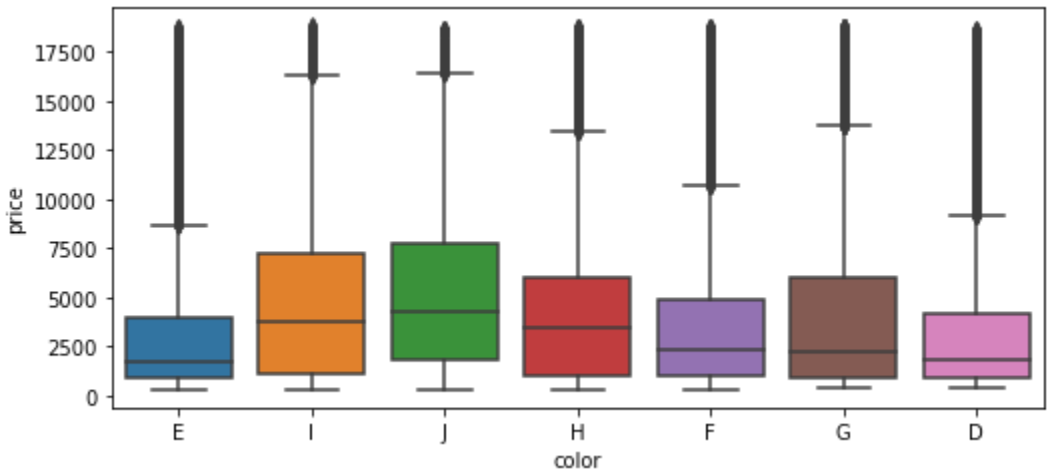
1.2 Plot of numerical features:

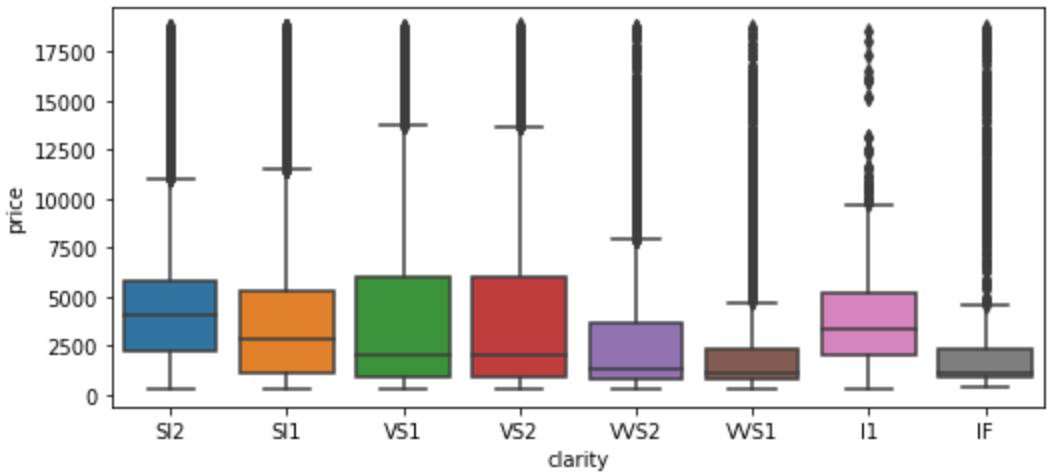


For high skewed data, we can do transformation like log transform, square-root transform, and box-cox transform. We also can replace extreme values, trimming data, and standardizing the data.

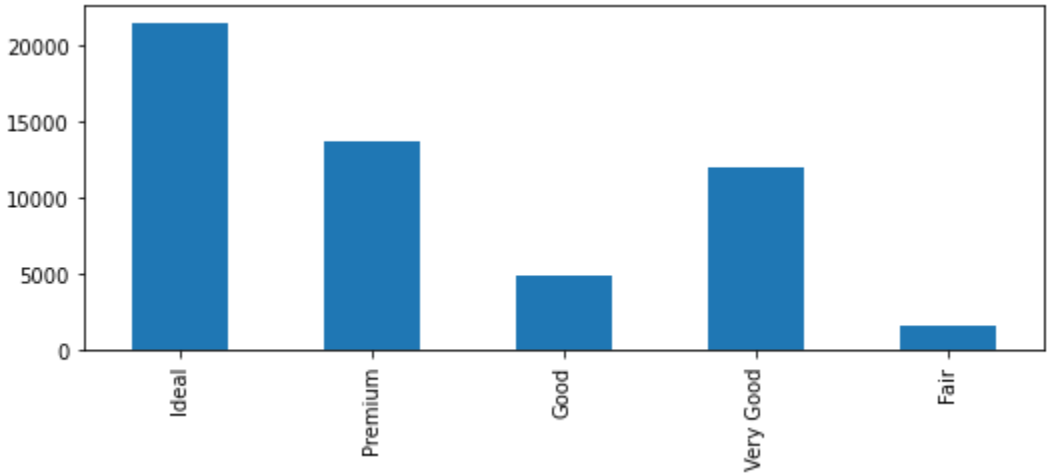
1.3 Plot of category features:

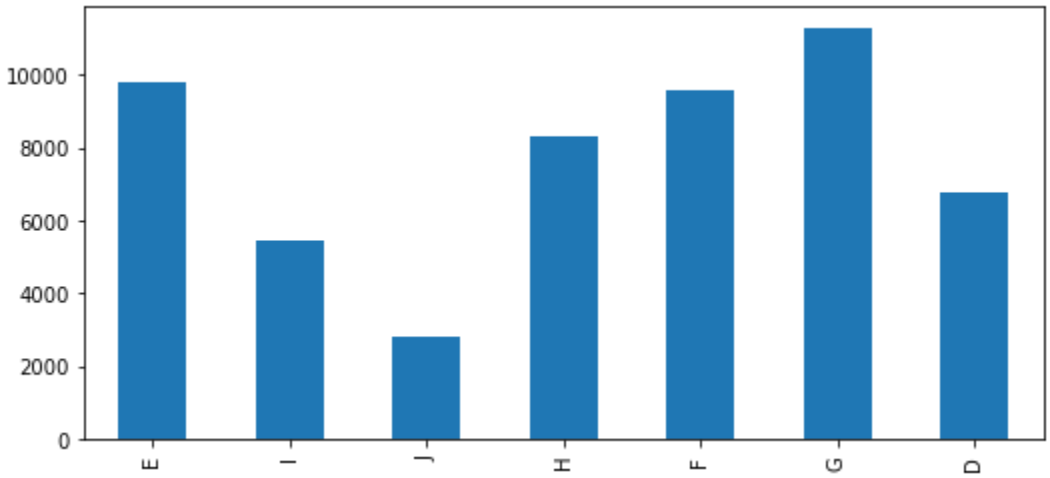


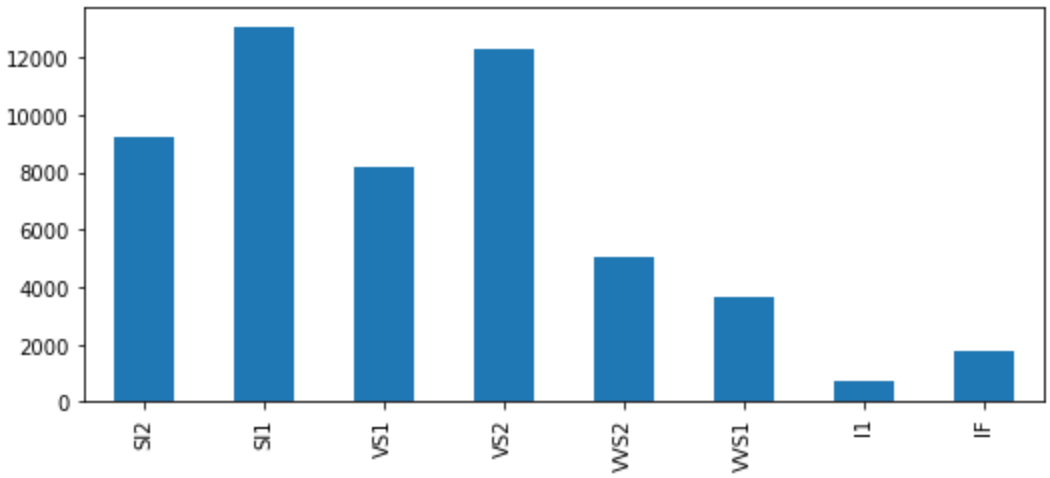




1.4 Plot the counts by color, cut and clarity:







Question 2:

2.1 Standardize feature columns and prepare them for training:

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaled_data=scaler.fit_transform(processed_data)
```

2.2

Describe how this step qualitatively affects the performance of your models in terms of test RMSE.

This step helps us select features that are relevant so that noise and irrelevant features that are used to training can be reduced. By reducing irrelevant features and focusing more on relevant features, the impact of irrelevant data can be reduced and thus the model has better performance.

Is it true for all model types?

This is not true for every model, for models that are simple and contain lots of data may not be largely impacted by irrelevant data.

Also list two features for either dataset that has the lowest MI w.r.t to the target.

Depth and **table** have lowest MI with respect to target.

Question 3:

For a random forest model, measure "Out-of-Bag Error" (OOB) as well. Explain what OOB error and R2 score means:

OOB Error is the average prediction error rate across all tree in the random forest, which is 1-self.oob_score_ given the link

R2 is the proportion variance explained in the "Price" that is predicted by independent variables. R2 score measures how the changes of target variable is represented by the model, which is just self.oob_score_ given the link.

Question 4:

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

Two best-known techniques for shrinking the regression coefficients towards zero are ridge regression and lasso. For both regularizations, they're less flexible, decrease the variance, and increase the bias. But increase in bias is smaller than decrease in variance, so accuracy improved.

Lasso:

Lasso shrinks the coefficient estimates towards zero. L1 penalty has the effect of forcing some of the coefficient estimates to be exactly equal to zero when the tuning parameter λ is sufficiently

large. Lasso performs variable selection, so it yields sparse models.

Ridge:

Ridge regression puts further constraints on the parameter estimates in the linear model. It is less flexible than Lasso with highly-variable parameters that are shrunk, and so less variability.

	Lasso	Ridge
Penalty	$\|\beta\|$	β^2
Best parameter	$\alpha = 0$	$\alpha = 0$

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

Regularization	Penalty
Linear Regression	None

Because we use pipeline and Grid Search to find the best parameter with lowest negative root mean squared error which is alpha=0. It indicates that there is no ridge or lasso which is the same as linear regression.

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

All variables should be standardized so that they are essentially measured in the same units.

Also in Ridge regression, it requires that the predictors be standardized to have SD=1 before beginning. This is because they react to the units of measurement, since large numbers will play a bigger role.

The standard least squares coefficient estimates are scale equivariant: multiplying X_j by a constant c simply leads to a scaling of the least squares coefficient estimates by a factor of 1/c. In other words, regardless of how the jth predictor is scaled, $X_j\hat{\beta}_j$ will remain the same.

In contrast, the ridge regression coefficient estimates can change substantially when multiplying a given predictor by a constant, due to the sum of squared coefficient terms in the penalty part of the ridge regression objective function.

Therefore, it is best to apply ridge regression after standardizing the predictors, using the formula:

$$\widehat{x_{ij}} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij}-\bar{x_j})^2}}$$

4.4: Some linear regression packages return p-values for different features2 . What is the meaning of these p-values and how can you infer the most significant features?

Theoretically, p values in regression help determine whether there is a relationship between each predictor variable and response variable. In our case, we are looking for a relationship between “depth”, “table”, “cut” and response variable, “price” is significant or not.

H_0: $price\ has\ no\ correlation\ with\ depth,\ table,\ cut\ variables$

H_1: $there\ is\ non\ -\ zero\ correlation\ with\ the\ price$

We usually reject the null hypothesis when p-values is less than the level of significance $\alpha = 0.05$ because there is sufficient evidence to conclude that the H1 is significant.

Question 5:

5.1: What are the most salient features? Why?

“Carat” is the most salient feature because we use pipes to find coefficients and Carat has the highest coefficient. It shows that carat has the highest positive correlation with the response variable, Price. As carat increases one unit, the mean of price tends to increase approximately 2444 units while holding other variables in the model constant.

	PolynomialFeatures	coeffient
0	1	-0.002752
1	carat	2444.249501
2	x	-264.517051
3	y	1310.071958
4	z	306.166006
...	...	...
457	color^4 clarity	24.885995
458	color^3 clarity^2	10.630920
459	color^2 clarity^3	17.631649
460	color clarity^4	15.919480
461	clarity^5	6.025050

462 rows × 2 columns

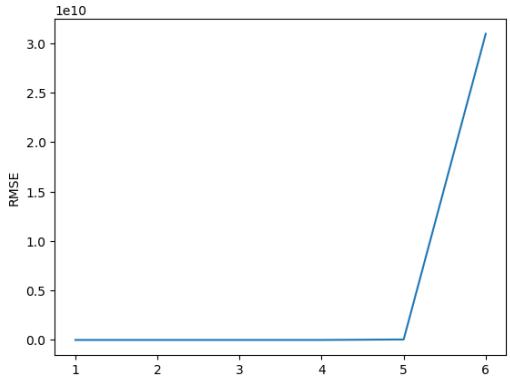
Polynomial Features’ Coefficient

5.2:

What degree of polynomial is best?

Best Degree of Polynomial	5

How did you find the optimal degree?



Different degree with RMSE

From the plot above, we can see after polynomial degree 5, RMSE increases rapidly. So degree 5 is the best polynomial with lowest RMSE and avoid overfitting.

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

It also will lead to overfitting problems which means training RMSE is very low but testing RMSE will be very high.

What about its performance on testing data?

High-order polynomials will have overfitting problems and test error will be very high. It will not perform well in testing data.

Question 6:

6.1 Find a good hyper-parameter set systematically:

model_alpha	model_hidden_layer_size
1	(100,1)

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

Function	No-op activation (Linear Regression)	Logistic sigmoid	Hyperbolic tan	Rectified linear unit

RMSE	1213.8348	4247.2039	4246.0381	632.0160

Multi-layer perceptrons (MLP) is an artificial neural network that has many layers of perceptrons. These layers are a single input layer and single output layer of perceptrons. It's follow back propagation which is MLP receives feedback on the error and MLP adjusts the weights so that to make better prediction.

We could consider No-op Activation as linear regression. As the table above shows, we can see MLP performs much better than linear regression. It's because MLP has multiple hidden units and can learn complex non-linear relationships.

6.3 What activation function did you use for the output and why?

We used rectified linear unit function (RELU) as an activation function because it has the best performance under the best parameter set. The RELU function produces sparsity in the activation output. This means that only a subset of the neurons will be activated for a given input, resulting in a more efficient use of computation and a reduction in overfitting.

Besides, RELU is a good activation function for gradients that can be small, which can be caused by deep neural networks. Unlike the sigmoid function, the RELU function does not saturate when the input is large, thus avoiding the vanishing gradient problem.

The RELU function is computationally efficient to compute and differentiate. This makes it suitable for large-scale neural networks, where computation time is a critical factor.

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

If we increase the depth of the network too far, the model can be overfitting and make the performance worse. Overfitting occurs when a neural network is too complex and has too many parameters relative to the amount of training data available. This can cause the network to fit the

training data too closely, which can result in poor generalization to new, unseen data.

Besides, it is possible that as depth increases, gradients become too small and hard to compute. Vanishing gradients occur when the gradients become too small as they backpropagate through the deep layers of the network, making it difficult for the network to learn meaningful representations of the input data. This can result in slower training times and poor performance on the test data.

Additionally, increasing the depth of the network can increase the computational cost of training and inference, which can be a significant practical consideration.

Question 7:

7.1

Explain how these hyper-parameters affect the overall performance.

Maximum number of features:

It indicates the maximum number of features to be considered when splitting into forest. It may bring underfitting or overfitting when performing a model. A smaller number of features can facilitate the randomness of the forest and may give better performance. But it is possible that a too low number of features may neglect important features and give a bad performance number of trees. It also has a bias-variance problem because when increasing the maximum number of features will reduce bias but increase variance.

Number of trees:

More trees allows taking more trees into consideration to give predictions. But too many trees may cause overfitting. As the number of trees increases, it also increases the computational cost. Model also may converge after a certain number of trees.

Depth of tree:

The deeper a tree can be the better the model can fit the data. But because of that, a deeper tree can cause overfitting and affect the performance of the model. Generally, increasing depth will reduce bias but also increase variance. For high-dimensional data, setting depth of trees to be a smaller value will help prevent the curse of dimensionality and improve performance.

Describe if and how each hyper-parameter results in a regularization effect during training:

Basically is similar to the above question which will bring overfitting, underfitting, bias-variance tradeoff, dataset size, and dimensionality of data problems. If numbers are too high, the model may overfit the training data. If parameters are too low, it will underfitting. When decreasing the numbers, it will reduce variance but increase bias. For a small data set, a small number may be sufficient while larger data need a larger maximum number. For high-dimensional data requires smaller maximum numbers to prevent curse and improve performance.

Maximum number of features:

It will have a regularization effect. Setting a small maximum number of features will take less number of features into consideration and avoid overfitting

Number of trees:

It will have a regularization effect. By having more trees, the effect of overfitting that may happen in one tree will be reduced and avoid the impact of overfitting.

Depth of tree:

It will have a regularization effect as well. Setting a smaller depth of tree will avoid overfitting data

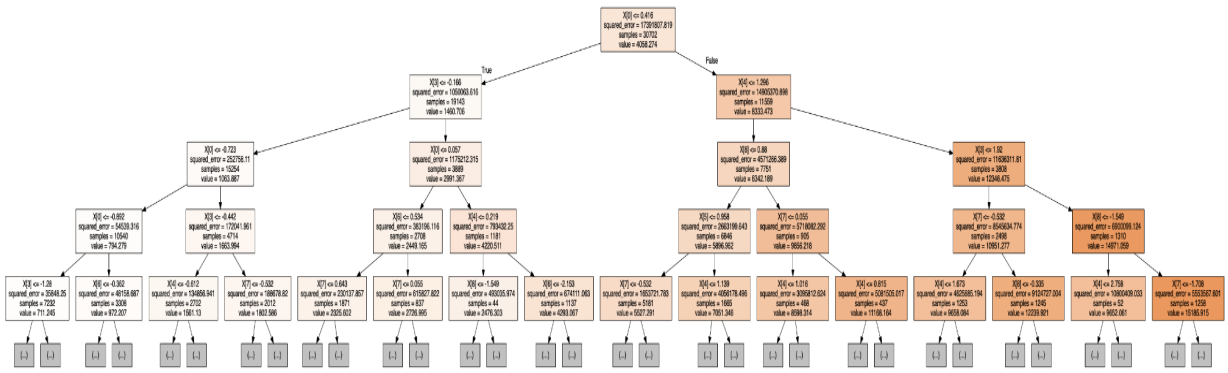
7.2 How do random forests create a highly non-linear decision boundary despite the fact that all we do at each layer is apply a threshold on a feature?

Random forests create highly non-linear decision boundaries by combining many individual decision trees, each of which applies a threshold on a feature, but the combination of all the trees in the forest results in a highly non-linear decision boundary that can capture complex interactions between features.

When a threshold is applied on a feature at each layer, a linear boundary is formed to split into groups on each tree. But each tree will apply such a threshold on a feature and form a boundary to make predictions. Predictions from multiple trees combined will form a highly non-linear boundary.

7.3

Randomly pick a tree in your random forest model (with maximum depth of 4) and plot its structure.



Plot of a tree in random forest

Which feature is selected for branching at the root node?

As the plot shown above, X[0]is selected for branching at the root node which is carat

What can you infer about the importance of this feature as opposed to others?

The feature selected for branching at root is more important than other features, though it is possible that the feature selected for branching at root is not the most important one.

Do the important features correspond to what you got in part 3.3.1?

As the feature_importances_ shown, the most important ones are X[0],X[4],X[8],X[3], which correspond to carat, y, clarity, x. This is not exactly what I got in 3.3.1 but it covers most of the important features like carat, y, x.

7.4 Measure “Out-of-Bag Error” (OOB). Explain what OOB error and R2 score means.

OOB	0.02304389892974079

OOB error indicates how the model performs in unseen data by measuring prediction error rate on unseen data.

R2 score indicates how much variance explained by the model which shows how the model fits the data provided.

Question 8:

8.1 Read the documentation of LightGBM OR CatBoost and determine the important hyperparameters along with a search space for the tuning of these parameters (keep the search space small).

Parameter	Learning_rate	n_estimators	num_leaves	subsample_for_bin

Learning_rate:

A smaller learning rate can lead to a more accurate model, but at the cost of slower convergence and longer training times. On the other hand, a larger learning rate can lead to faster convergence but may also result in overshooting the optimal weights and lower model performance.

N_estimators:

n_estimators parameter specifies the number of boosting iterations or trees to be built in the ensemble. Each iteration adds a new tree to the ensemble, with the goal of improving the model's predictive power. Increasing the number of trees can improve the model's performance, but at the cost of longer training times and a higher risk of overfitting.

Num_leaves:

num_leaves parameter specifies the maximum number of leaves in each decision tree. A larger value of num_leaves can lead to a more complex model with higher capacity, which may result in better performance on the training set. However, a higher value can also lead to overfitting, longer training times, and increased memory usage.

Subsample_for_bin:

subsample_for_bin parameter controls the number of samples used to construct the bins for numerical features. The algorithm constructs bins to discretize numerical features, which can improve the model's performance and generalization.

8.2

Apply Bayesian optimization using skopt.BayesSearchCV from scikit-optmize to find the ideal

hyperparameter combination in your search space.

Parameter	Learning_rate	n_estimators	num_leaves	subsample_for_bin
Value	0.0373	344	39	352416

Report the best hyperparameter set found and the corresponding RMSE.

RMSE	665.0184

8.3 Qualitatively interpret the effect of the hyperparameters using the Bayesian optimization results: Which of them helps with performance? Which helps with regularization (shrinks the generalization gap)? Which affects the fitting efficiency?

Performance	Regularization	Fitting efficiency
Accuracy	Prevent overfitting	Training Speed

Performance:

Learning rate: The learning rate determines the step size at which the model updates its weights during training. Based on the Bayesian optimization results, we can observe that a higher learning rate can generally lead to better performance, up to a certain point where the model becomes unstable and fails to converge.

Regularization: parameter C and parameter gamma

Dropout rate: Dropout is a popular regularization technique that randomly drops out a fraction of the units in each layer during training. The dropout rate determines the fraction of units to drop out. Based on the Bayesian optimization results, we can observe that a higher dropout rate can generally lead to more regularization and better generalization performance, up to a certain point where the model starts to underfit.

L1 and L2 regularization: L1 and L2 regularization add a penalty term to the loss function of the model, which helps to limit the weights' magnitude. The hyperparameters that control the strength of the regularization are lambda1 and lambda2, respectively. Based on the Bayesian optimization results, we can observe that increasing the values of lambda1 and lambda2 can lead to more regularization and better generalization performance, up to a certain point where the model starts to underfit.

Fitting Efficiency: subsample_for_bin

We could estimate the model fitting efficiency by dividing the sum of the effective sample size (ESS) of all the chains by the elapsed time to run iterations excluding the warmup time.

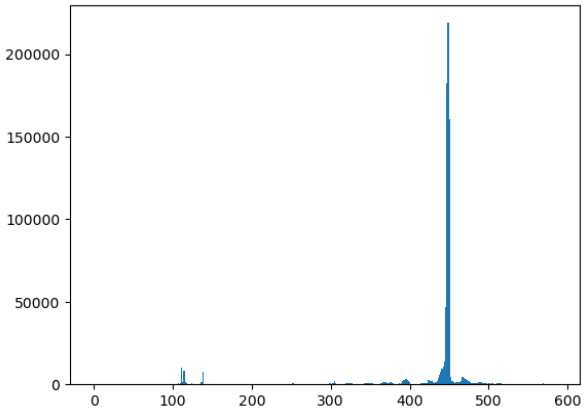
Question 9:

9.1 Statistics for each hashtag:

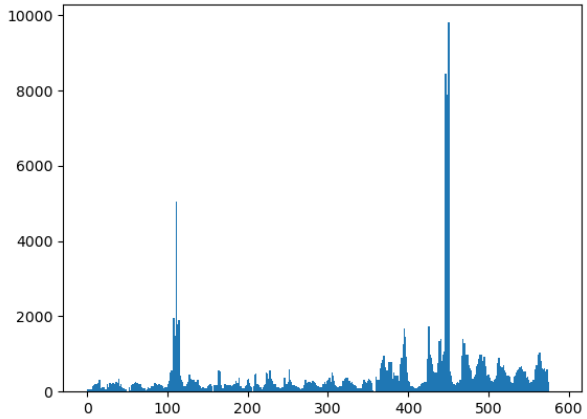
```
1 tags_stats
```

```
{'#gohawks': {'avg_twt_perhour': 0.0034189454292693376,
  'avg_follower': 0.0034189454292693376,
  'avg_retwts': 2.0132093991319877},
 '#gopatriots': {'avg_twt_perhour': 0.02441722314377667,
  'avg_follower': 0.02441722314377667,
  'avg_retwts': 1.4081919101697078},
 '#nfl': {'avg_twt_perhour': 0.0025187559782147415,
  'avg_follower': 0.0025187559782147415,
  'avg_retwts': 1.5344602655543254},
 '#patriots': {'avg_twt_perhour': 0.0013317454229371728,
  'avg_follower': 0.0013317454229371728,
  'avg_retwts': 1.7852871288476946},
 '#sb49': {'avg_twt_perhour': 0.000783173020246559,
  'avg_follower': 0.000783173020246559,
  'avg_retwts': 2.52713444111402},
 '#superbowl': {'avg_twt_perhour': 0.00048259790520542383,
  'avg_follower': 0.00048259790520542383,
  'avg_retwts': 2.3911895819207736}}
```

9.2



Number of tweets in hour for SuperBowl



Number of tweets in hour for NFL

Question 10:

Tasks:

1.Predict the number of retweets/likes/quotes.

2.Predict the hashtags or how likely it is that a tweet belongs to a specific team fan.

Task 1: Predict the number of retweets/likes/quotes

- Explore the data and any metadata (you can even incorporate additional datasets if you choose).

  Data exploration check whether retweet has relation to popular tags and also check range of retweet count

- Describe the feature engineering process. Implement it with reason: Why are you extracting features this way - why not in any other way?

  After exploring the dataset, I found a follower, a hashtag, a tweet included, who is the author, and how influential is the author and the tweet can be useful for training models. A user with more followers is more likely to get more retweets. A tweet with a popular hashtag is more likely to get more retweets. The more influential the author is, the more

retweets a tweet can get. The more influential the tweet is, the more retweets a tweet can get.

- Generate baselines for your final ML model.

  Baseline 1: take advantage of the interaction between user and tweet using NMF and check whether a user is more likely to get more retweets.

  Baseline 2: use linear regression with different regularization to predict retweet count model with best performance is neural network with MLP to predict retweet count.

- A thorough evaluation is necessary:

	Baseline1	Baseline2 with regularization	Best Model (MLP)
MSE score	161.51096155337777	134.5170710481433	127.63968988015372

Task 2: Predict the hashtags or how likely it is that a tweet belongs to a specific team fan.

- Explore the data and any metadata (you can even incorporate additional datasets if you choose).

  We first clean the data and split the text by "#". Data exploration checks whether a retweet has relation to text and also inspect the tag.

- Describe the feature engineering process. Implement it with reason: Why are you extracting features this way - why not in any other way?

  Tested user description, tweet text, geolocation, and Tags

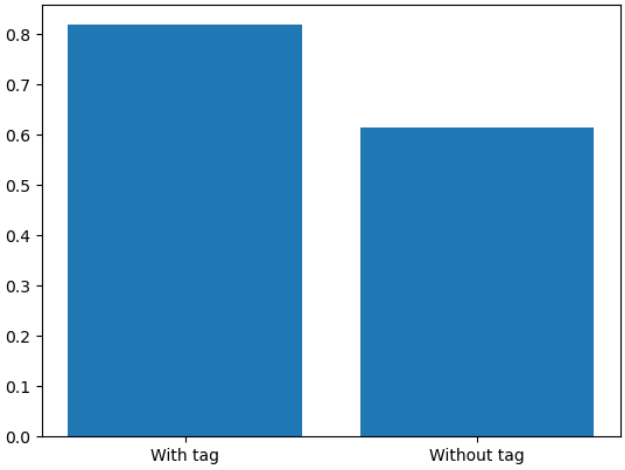
  Tweet text: highly correlated with the team

  Geolocation: not or mild correlated with the team

  Tags: correlated with the team

  Description: not or mild correlated with the team

So, we used **text** and **tags** as the features.



Compare with tag and without

- Generate baselines for your final ML model.

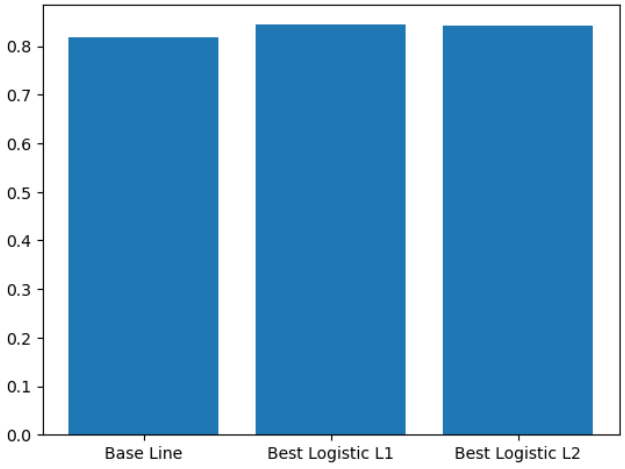
  We use logistic regression and SVM as our final ML model.

  Baseline1: Logistic regression with L1 regularization

  Baseline2: Logistic regression with L2 regularization

  Baseline3: SVM model

- A thorough evaluation is necessary:



Compare accuracy with best Baselines

Best model	Accuracy
Logistic Regression with L1 regularization	0.8438400046526638