219 - Large-Scale Data Mining:
Models and Algorithms

Report on

Project 4: Regression Analysis and Twitter Data Analysis

Authors
Jayanth Shreekumar (805486993)
Narayan Gopinathan (605624014)
Yuheng He (505686149)

Winter 2022

Part 1 - REGRESSION ANALYSIS

Introduction

Regression analysis enables us to find the relationships between variables in a real-life system. For example, in a thermal power plant, regression can help us find the strength of the relationship between ambient temperatures, power output, and pollutant output. Using regression analysis, we can find the strength and the signs of the relationships between different variables, to see how strongly they affect each other. Regression analysis has two main components:

1. Dependent Variable: what we're trying to understand or predict. It can be the price of a house, the price of a stock at a certain point in the future, and so on.
2. Independent Variables / Features are what we believe have an impact on the dependent variable and influence its fluctuation.

Regression models primarily find the correlation between the dependent variable and the features and fit a best-fit curve to the data for predictions. In the first part of the project, we perform regression analysis using several different models and compare the results on two different datasets, the diamond dataset where we predict the price of a diamond given its features, and the gas emissions dataset where we predict the level of CO in the air based on several factors such as temperature, pressure, etc..

Diamond Dataset

The diamond dataset contains information on the price and other characteristics of diamonds. This dataset contains information about 53,940 synthetic round-cut diamonds. There are ten features for each diamond, which specify the diamond's properties. These are are:

- **carat:** The carat of the diamond represents its mass. One carat is equal to .200 grams.
- **cut:** The quality of the cut of the diamond can be classified as fair, good, very good, premium, or ideal. These qualitative categories can be coded quantitatively on a 1-5 scale.
- **color:** The color of a diamond is graded on a qualitative scale from J (worst) to D (best). These can be coded quantitatively on a 1-8 scale.
- **clarity:** The clarity of a diamond is measured qualitatively into categories: I1 (worst), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (best). These can be coded quantitatively on a 1-8 scale, respectively.
- **x:** The x-dimension is its length in millimeters.
- **y:** The y-dimension is its width in millimeters.
- **z:** The z-dimension is its height in millimeters.

- **depth:** This measures the depth as a percentage.

$$Depth = \frac{z}{mean(x, y)} = \frac{2 \times z}{x + y}$$

- **table:** This represents the width of the top of the diamond relative to its widest point.
- **Price:** The price in US dollars is the dependent variable, and what we want our model to predict based on the other characteristics.

Gas Emissions Dataset

The dataset on emissions of gases from a gas turbine in Northern Turkey has eleven categories. Each row of data represents the conditions at the turbine for one hour. The categories are:

- **AT:** The ambient temperature outside, in degrees C.
- **AP:** The ambient air pressure outside, in mbar.
- **AH:** The ambient humidity outside, in percentage.
- **AFDP:** The air filter difference pressure, in mbar.
- **GTEP:** The gas turbine exhaust pressure, in mbar.
- **TIT:** The turbine inlet temperature, in degrees C.
- **TAT:** The turbine outlet temperature, in degrees C.
- **CDP:** The compressor discharge pressure, in mbar.
- **TEY:** The turbine energy yield, in megawatt-hours.
- **CO:** Carbon monoxide output, in mg/m^3. This is the dependent variable that we are trying to predict based on the other variables.
- **NOx:** Nitrogen oxide output, in mg/m^3. This variable will be ignored, as we will focus solely on emissions of carbon monoxide.

Standardization

Question 1

Standardization is the process of changing the values of a dataset such that each column is distributed with mean 0 and variance 1. This is a very important step in training machine learning models as unstandardized datasets can have columns that have values that range from a very small value to very large values which affects the learning. For example, consider a twitter dataset where one of the features is the number of retweets and another is year that the tweet was tweeted. Clearly, the year value is a four digit number, but the number of retweets can range anywhere from 0 to millions. This disrupts learning.

For each of the datasets, we performed feature standardization using sklearn's StandarScaler() class.

For the gas emissions dataset, we chose to work on CO levels.

Data Inspection

Question 2

Pearson's correlation coefficient measures the linear correlation between two features in the dataset. A value of 1 means that the two features are perfectly correlated, if one increases, the other also increases. A value of -1 means that the two features are still perfectly correlated, but now if one of them increases, the other decreases. A value of 0 indicates no correlation. The formula is given by:

$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$ where cov(X, Y) is the covariance given by:

$cov(X, Y) = E[(X - \mu_X)(Y - \mu_Y)]$

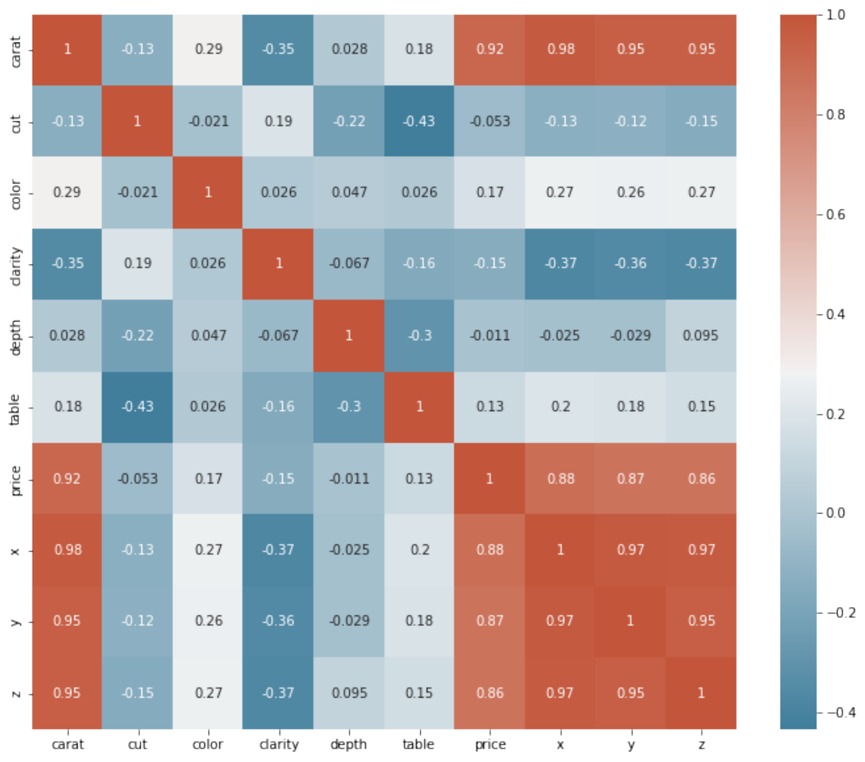


Figure 1: The matrix of correlation coefficients for each variable for diamond price

We see that for the diamond dataset, carat, x, y, and z have the biggest correlation with the target variable, which is price. This makes sense, as bigger diamonds have higher prices.

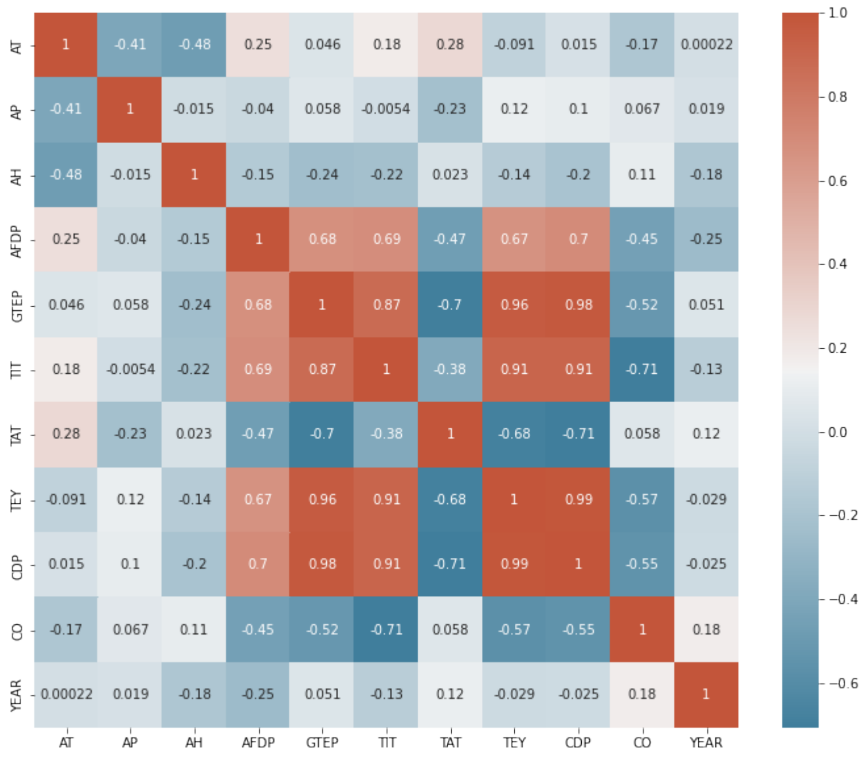
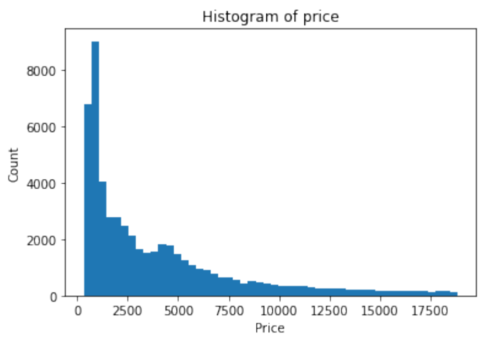


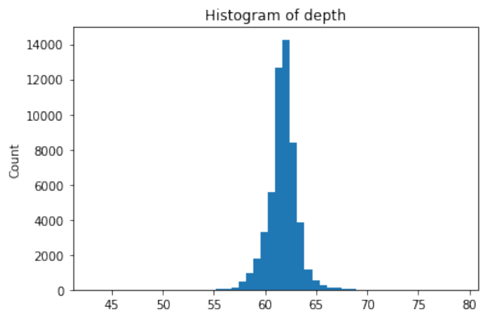
Figure 2: The matrix of correlation coefficients for each variable for CO output from the gas turbine

For the gas dataset, we see that the feature turbine inlet temperature(TIT) has the highest negative correlation with the target variable. This means that when the turbine is running, it causes heat and so higher temperatures, and the CO emission decreases. This suggests that the turbine is being used for some form of filtration or air cleaning system. Also, we see that CO emission is positively correlated with year, suggesting that the CO emissions is increasing every year.

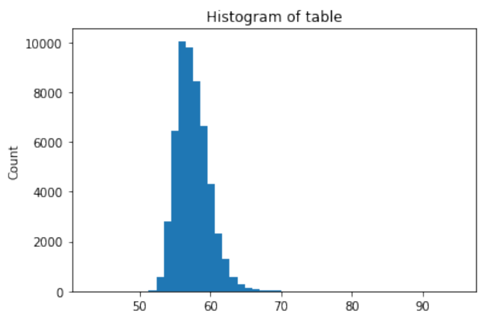
Question 3



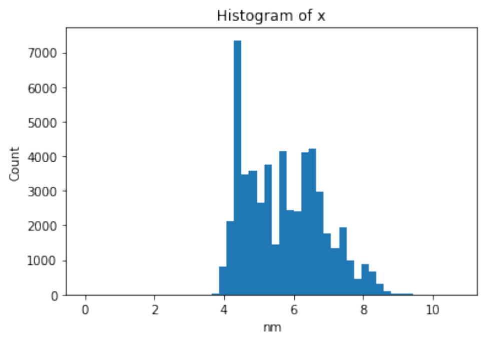
(a) The histogram for diamond price



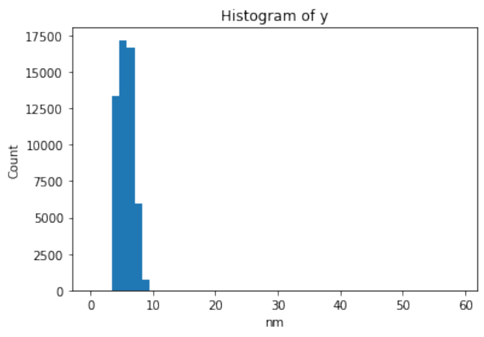
(b) The histogram for diamond depth



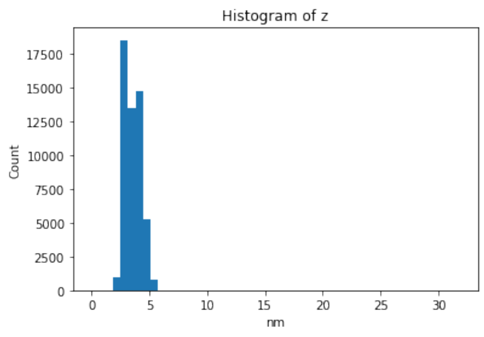
(c) The histogram for diamond table



(d) The histogram for diamond X



(e) The histogram for diamond Y



(f) The histogram for diamond Z

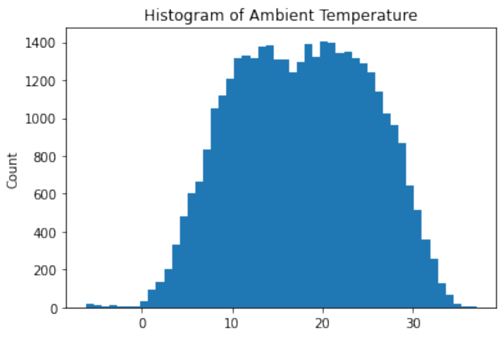
Figure 3: Histograms for diamond dataset

The numerical features of the diamond dataset and their histograms are plotted above along with the histogram for the target variable price.

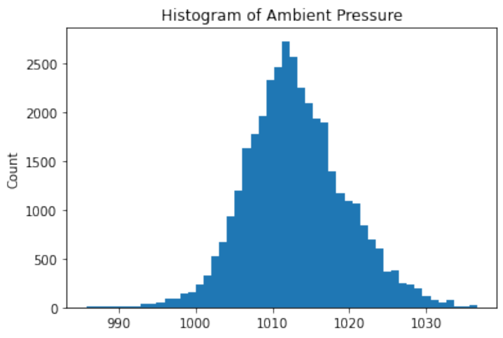
Skewness measures the amount of asymmetry in a histogram. A histogram that has a normal distribution has 0 skewness. From the figures, we see that the target variable price is skew right. Among the features, table, Y, and Z are also skew right, but X and depth are not skewed, and X seems to be bimodal.

The numerical features of the gas dataset and their histograms are plotted below along with the histogram for the target variable CO. We see that AFDP and the target vairable CO are skew right, while TIT and TAT are skew left. Also, CDP and TEY appear to be multimodal.

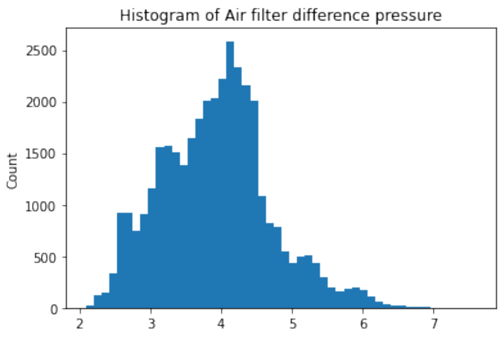
Skewness in undesirable and has to be fixed as it can tamper with learning. To this end, **pre-processing such as outlier removal, scaling, and standardization** among other methods are used.



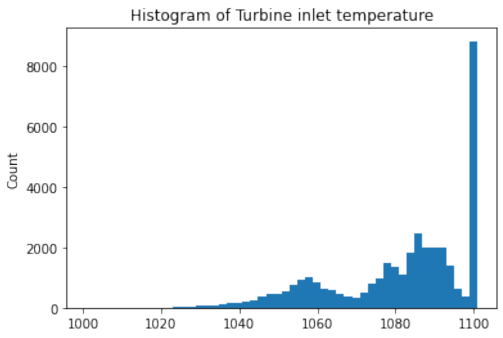
(a) The histogram for ambient air temperature for the gas turbine



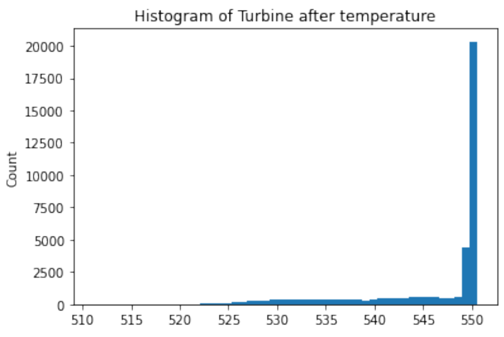
(b) The histogram for ambient air pressure for the gas turbine



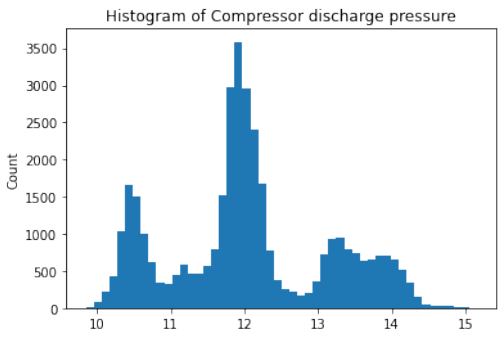
(c) The histogram for air filter difference pressure for the gas turbine



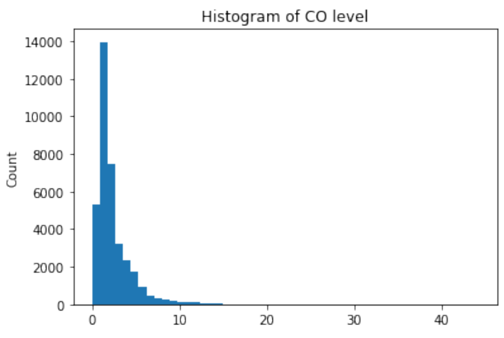
(d) The histogram for turbine inlet temperature for the gas turbine



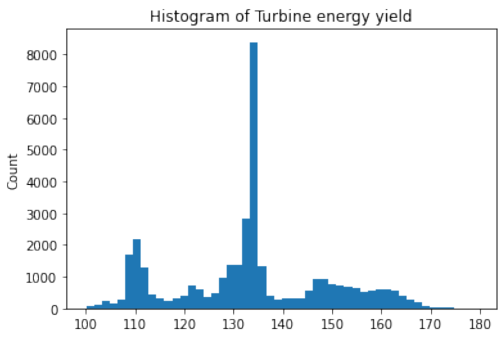
(e) The histogram for Turbine after temperature for the gas turbine



(f) The histogram for Compressor discharge pressure for the gas turbine



(g) The histogram for carbon monoxide emissions for the gas turbine



(h) The histogram for Turbine energy yield for the gas turbine

Figure 4: Histograms for Gas emissions dataset

Question 4

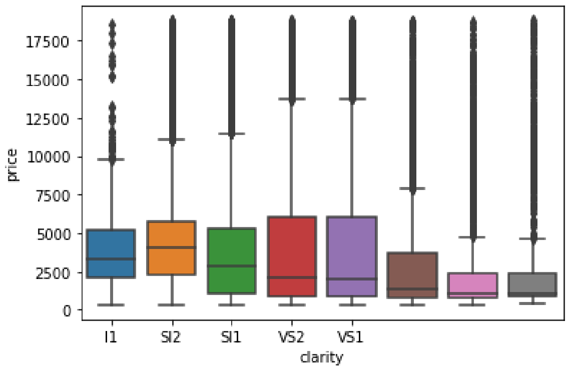


Figure 5: The boxplot for price and diamond clarity

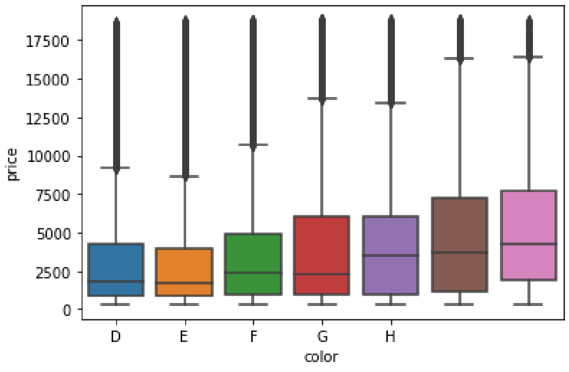


Figure 6: The boxplot for price and diamond color

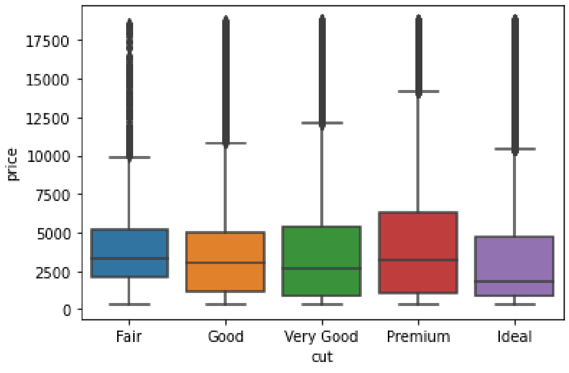
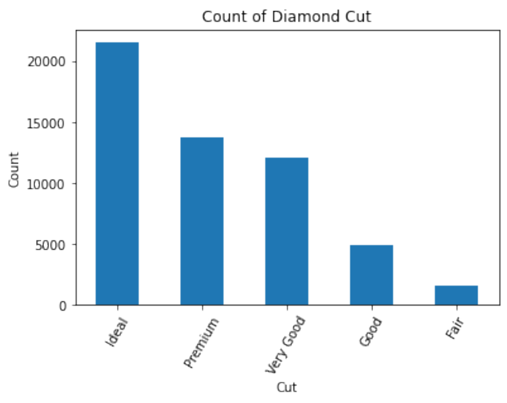


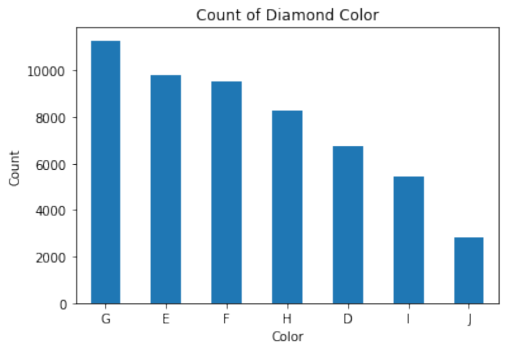
Figure 7: The boxplot for price and diamond cut

We find that there is a clear trend among clarity and color - they both have a clear impact on the distribution for the price of the diamond as seen by their respective means. There is not much trend among cut, implying that it does not matter as much for the diamond price. However, there are big distributions among diamond prices for all of these variables with plenty of outliers, implying that there are other factors that matter more for the diamond price such as human emotions, market price, number of bidders, and so on.

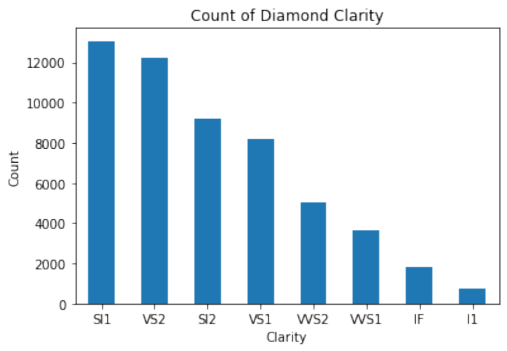
Question 5



(a) The counts plot for diamond clarity



(b) The counts plot for diamond color



(c) The counts plot for diamond cut

Figure 8: Plot of Counts of diamonds by cut, color, and clarity

All three of the categorical features have very straightforward plots, where the number of diamonds decreases as the quality increases. This is expected and also seen in reality, as the price of the diamond is influenced by the quality: better the quality, lesser the number of diamonds, rarer the diamond, and hence higher the cost.

Question 6

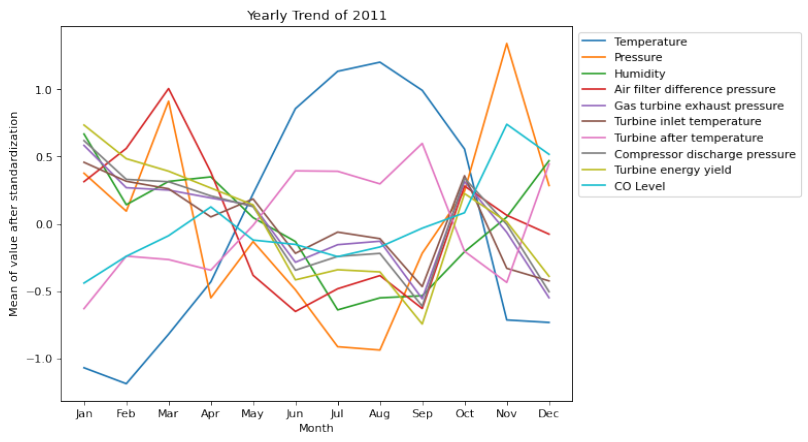


Figure 9: The Yearly trends in 2011

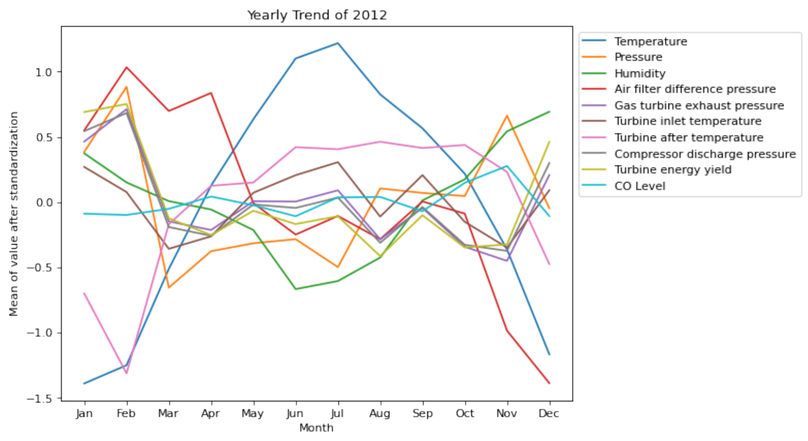


Figure 10: The Yearly trends in 2012

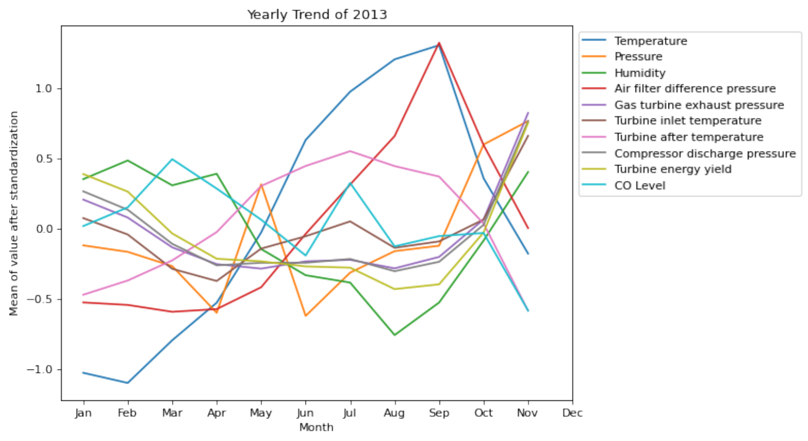


Figure 11: The Yearly trends in 2013

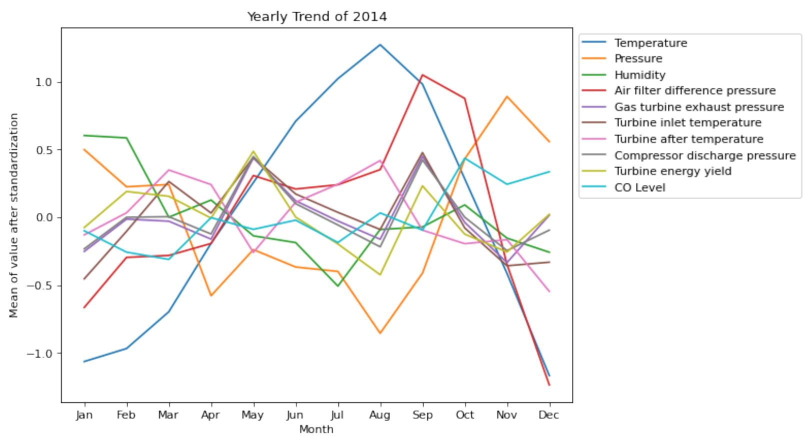


Figure 12: The Yearly trends in 2014

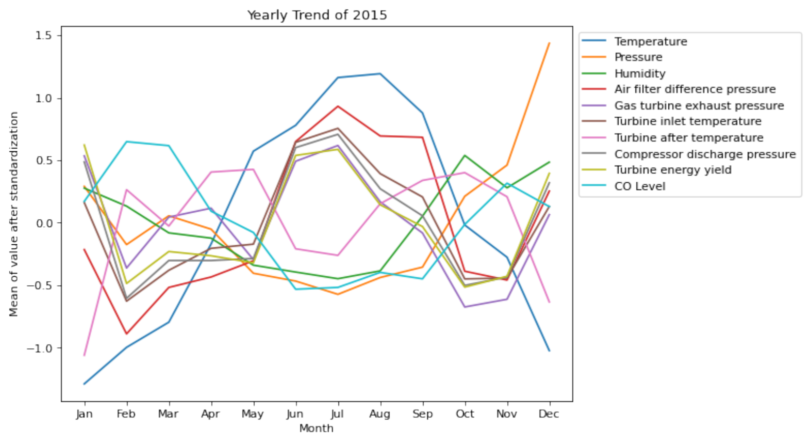


Figure 13: The Yearly trends in 2015

The yearly trends of each feature in the gas dataset is shown above. We see the following patterns in all years:

1. Temperature always increases in the center and falls off on wither sides, indicating the seasons.
2. CO level always decreases in the summer.
3. TIT, GTEP, TEY, and CDP are always similar, imdicating that they are highly correlated. This can confirmed by looking at the pearson correlation coefficient heatmap.
4. Pressure and Temperature are inversely proportional, and this can also be confirmed by looking at the pearson correlation coefficient heatmap.

Feature Selection

Question 7

Feature selection is the process of reducing the number of input variables when developing a machine learning model. It is preferable reduce the number of input variables as it leads to reduction in both time and space complexity and also sometimes improves the performance of a model.

Mutual information regression returns the estimaed mutual information between a feature and the label. Mutual information is a non-negative value that measures the dependency between two variables. If the variables are completely independent, the value of mutual information is 0, while higher dependency leads to higher values. The function relies on non-parametric measures based on entropy estimation from nearest neighbors.

F regression return the F-score, which is used for comparing the change in the results of a model before and after the usage of new additional features.

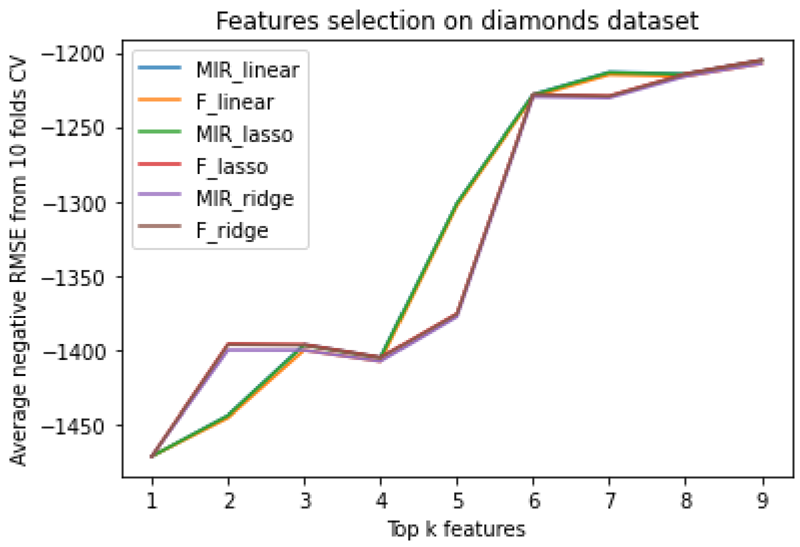


Figure 14: Feature selection on Diamonds dataset

The effect of feature selection on the diamonds dataset is shown above. We clearly see that as the number of features increases, the test RMSE becomes better. This is expected, as the model will perform better with more features in general. Also, both F and MI feature selection converge to similar values of test RMSE. A good value of the number of features for all of the 6 methods in the graph is 6.

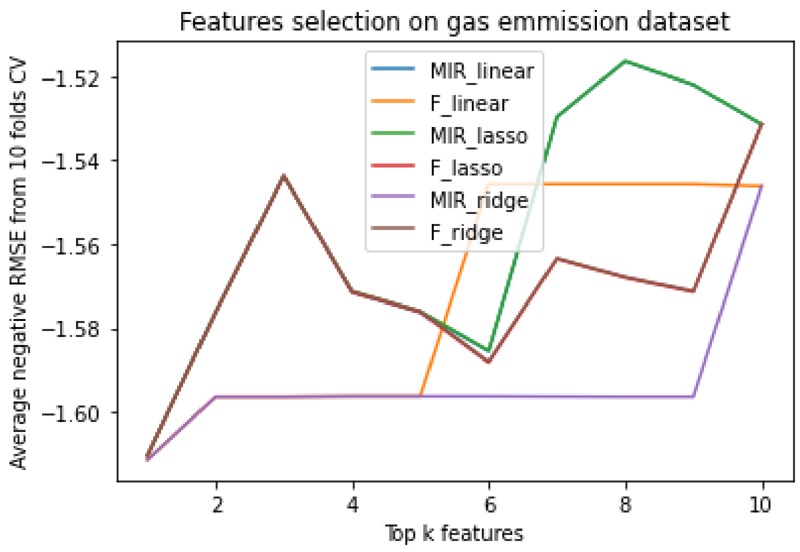


Figure 15: Feature selection on Gas emission dataset

The effect of feature selection on the gas emissions dataset is shown above. Here, we do see a difference between MI and F feature selection. F feature selection converges much faster when paired with ridge or linear regression as the test RMSE becomes better very quickly. This indicates that the F score converges faster than MI. However, the opposite holds true for lasso regression.

We recorded the best k values for each of these methods in the graph as follows:

	k
Linear with F	3
Linear with MI	8
Lasso with F	10
Lasso with MI	6
Ridge with F	3
Ridge with MI	8

Table 1: Optimal number of features for gas emissions dataset

Linear Regression

Introduction

Linear regression is a tool used to predict one variable, called the target variable, based on another or multiple other variables called features. The objective function is to find a relationship between one or more features (independent variables) and a continuous target variable (dependent variable).

Ridge regression, also known as linear regression with L2 regularization, and Lasso regression, also known as linear regression with L1 regularization are popular methods used to prevent models from overfitting the dataset. This helps the models to become more generalizable by keeping the weights smaller, and so constraining how much a curve can "bend".

Question 8

Ridge regularization is a process under which an optimization algorithm is modified by adding a penalty equivalent to the square of the magnitude of the coefficients. This means that if two coefficients are correlated with one another, and there could be a range of possible coefficients, the ones with the smallest absolute values are chosen. This reduces the magnitude of the coefficients, and reduces the model size and multi-collinearity. It is an effective strategy to deal with multi-collinearity if the variables are correlated with each other.

Lasso regularization or L1 regularization is similar, but does not square the penalty. The optimization algorithm is regularized by adding a penalty equivalent to the absolute value of the coefficients. This means that the incentive is to minimize coefficients, and thus can also reduce overfitting. However, this encourages weights to shrink to 0, and so the weights matrix tends to be sparse. This, L1 regularization tends to zero out a lot more weights than L2 regularization.

Question 9

Dataset	**Model Type**	**Feature Selection**	**Optimal Penalty Parameter**	**Test RMSE**
Diamonds	**Linear**	**F**	N/A	-1228.22030417
		MI	N/A	-1228.22030417
	Lasso	**F**	0.0001	-1228.22044014
		MI	0.0001	-1228.22044014
	Ridge	**F**	0.0001	-1228.22033265
		MI	0.0001	-1228.22033265
Gas Emissions	**Linear**	**F**	N/A	-1.54369348657
		MI	N/A	-1.5163336268
	Lasso	**F**	0.1	-1.49500899026
		MI	1	-1.54568715557
	Ridge	**F**	100	-1.5436256945
		MI	1000	-1.50079798246

Table 2: Performance of different models on un-scaled datasets.

The table above lists results obtained for the 3 types of models paired with each feature selection scheme on unscaled datasets. We ran all experiments using the optimal k values that we obtained in the feature selection section described previously. To obtain the optimal regularization penalty, we ran a 10-fold cross-validation Grid Search over each model.

For the diamonds dataset, we see from the table that the type of feature selection does not play a role in the final test RMSE value, indicating that both methods yield the same features as important. Also, we notice that linear regression without regularization is the best possible choice here with a test RMSE of **-1228.22030417**.

For the gas emissions dataset, we see that the type of feature selection does play an important role, as seen by the different k values we obtained earlier. Here, the best Test RMSE is obtained for the lasso regression model with F feature selection, the test RMSE being **-1.49500899026** for the optimal penalty parameter of **0.1**.

Question 10

Dataset	**Model Type**	**Feature Selection**	**Optimal Penalty Parameter**	**Test RMSE**
Diamonds	**Linear**	**F**	N/A	-1228.22030417
		MI	N/A	-1228.22030417
	Lasso	**F**	0.0001	-1228.22038230
		MI	0.0001	-1228.22038230
	Ridge	**F**	0.0001	-1228.22031004
		MI	0.0001	-1228.22031004
Gas Emissions	**Linear**	**F**	N/A	-1.54369348657
		MI	N/A	-1.5163336268
	Lasso	**F**	0.01	-1.50687528107
		MI	1	-1.54568715557
	Ridge	**F**	100	-1.54282783629
		MI	1000	-1.50700128978

Table 3: Performance of different models on scaled datasets.

The table above lists results obtained for the 3 types of models paired with each feature selection scheme on scaled datasets. We see that feature scaling by and large does not affect the test RMSE values by a marge margin, especially when regularization is not used. Normally, we would expect feature scaling to improve test RMSE as discussed previously when using regularization but here, it seems like the datasets are good enough for regression without performing feature scaling. This is also reflected in the fact that the test RMSE values more or less remain constant regardless of the regularization method used.

Question 11

The meaning of p-values is the probability that an observed difference would appear by random chance alone. In a regression, the p-value is the chance that we would see this result for a given coefficient, assuming there is no real relationship. The program calculates it based on a normal distribution, using the standard error of its estimate for the coefficient. Results with very low p-values are the most significant, because those are the least likely to be generated by random chance. We can thus infer the most significant features by finding the regression coefficients with the lowest p-values.

Here are the p-values obtained for the diamonds dataset:
carat: 0.000000e+00
color: 0.000000e+00
clarity: 0.000000e+00
cut: 4.517350e-212
x: 8.216234e-106
depth: 2.554568e-35
z: 3.321856e-06
y: 1.049686e-03
table: 6.025587e-01

Polynomial regression

Introduction

Polynomial regression is similar to linear regression, but it uses a polynomial function, of the nth degree, where n is determined in advance. This enables the independent variables to be squared, cubed, or raised to other exponential powers. One might think of it as "creating" new features from existing features that make more sense. Fore example, using the area of a house rather than its length and breath to calculate its price makes more intuitive sense.

Question 12

Note: This question utilizes the results obtained on unscaled datasets using ridge regression without any feature selection methods. The best estimator values obtained using the 10-fold cross validation grid search employed in question 13 were used.

The 5 most salient features for each dataset are as follows:

Diamonds: carat, x, clarity, color, cut

Gas Emission: TAT, TIT * TAT, TIT, TAT^2,TIT^2, YEAR

For the Diamonds dataset, the most salient feature is carat, indicating that larger diamonds sell for a larger price. Also, we see that all 3 categorical features are in the most list, indicating that the beauty of a diamond is in its quality.

For the gas emissions dataset, the most salient features are the temperatures of the turbine and their combinations, indicating that CO emissions vary depending on whether the turbine is switched on.

Question 13

To find out what degree of polynomial is best, er performed a 10-fold cross validation grid search over degrees ranging from 1 to 4 on the scaled diamonds dataset, along with the regularization penalty parameter. The graph for this is shown below:

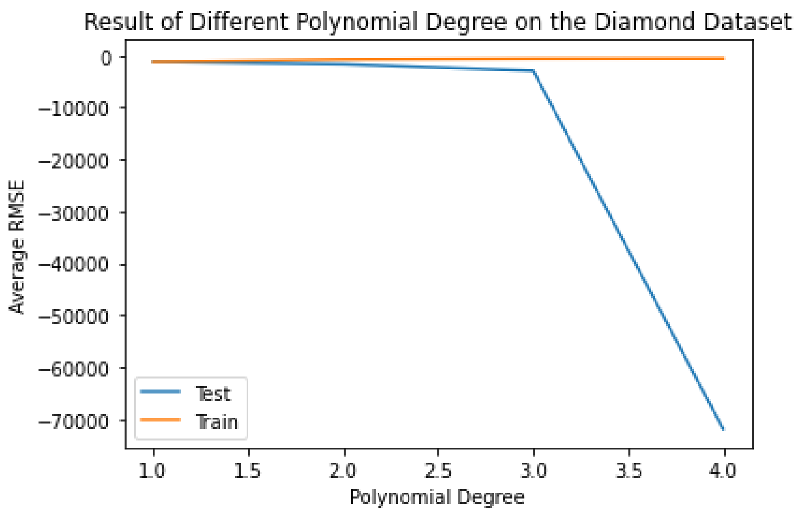


Figure 16: Polynomial degree selection on Diamonds dataset

We see that the test RMSE decrease for any higher order polynomial, indicating that first order polynomials are the best for this particular dataset. Any larger models have higher capacity and are capable of fitting more complex curves, leading to overfitting. We see that the model completely overfit the training data when the degree was 4, as seen by the sharp rize in test RMSE. The best test RMSE achieved is **-1205.045136**

To find out what degree of polynomial is best, er performed a 10-fold cross validation grid search over degrees ranging from 1 to 5 on the scaled gas emissions dataset, along with the regularization penalty parameter. The graph for this is shown below:

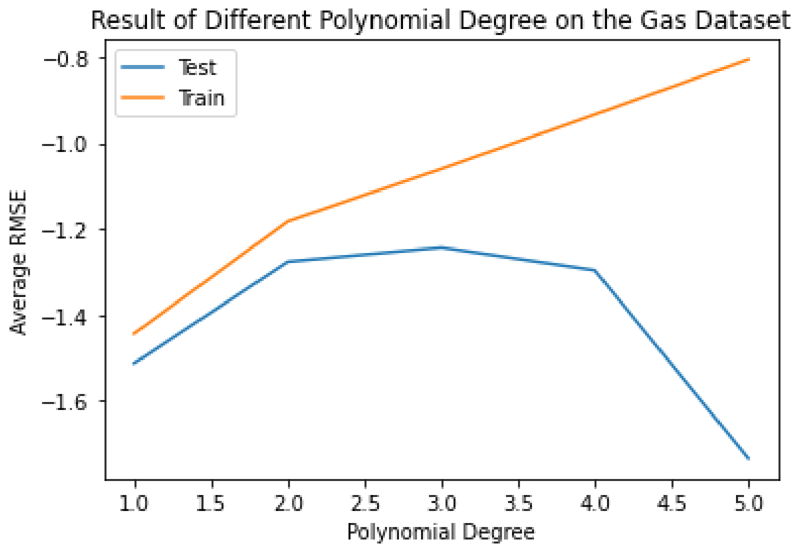


Figure 17: Polynomial degree selection on Gas Emissions dataset

Here, we see that polynomials up to the 3rd order help in decreasing test accuracy, as seen previously by the salient features. However, we see that overfitting does occur for 5th order polynomials. Other problems caused due to higher order polynomials are that they are memory intensive, and hard to interpret compared to linear models. The test RMSE achieved is **-1.243378**.

Question 14

The most obvious feature that is easily created from the existing ones is the volume of the diamond, given by X * Y * Z. Here are the results that we obtained after adding the volume feature and removing X, Y, and Z from our dataset:

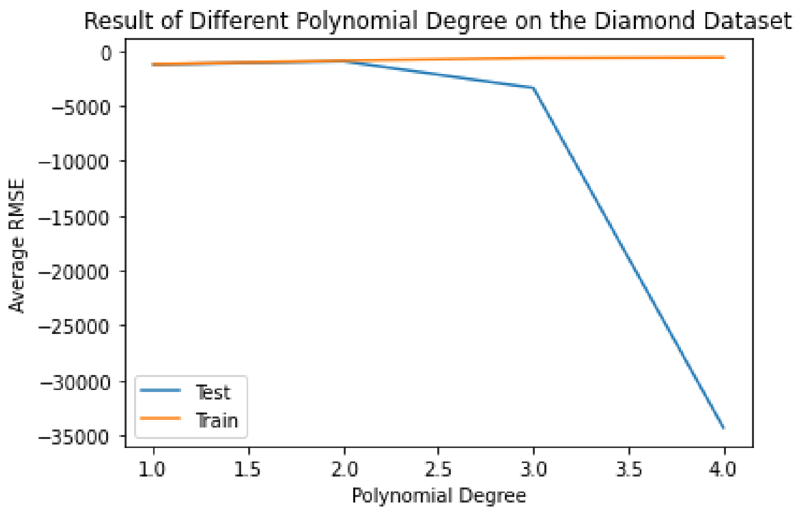


Figure 18: Polynomial degree selection Diamonds dataset with Volume

From the graph, we see that 2nd order polynomials perform well on the dataset now, meaning that there is no overfitting and the model performs better. This is also seen in the test RMSE values that we obtain. Without the volume feature, **the test RMSE was -1205.045136** but after including volume, we get the test RMSE value of **-947.00439**. Another observation is that the **test RMSE using polynomial features is much lesser than when using standard regression**, meaning that we would want to use polynomial features as long as we don't overfit on the dataset.

Neural networks

Introduction

Neural networks are a subset of machine learning models that form the basis of deep learning algorithms. These models can enable the independent variables to interact with each other in a nonlinear fashion using activation functions. They were inspired by the structure and function of the human brain. Multilayer perceptrons or MLPs are fully connected networks that have at least 3 layers of nodes. Learning in an MLP occurs by the idea of backpropagation, each connection between is given a weight, and these weights are tweaked constantly to achieve the minimum possible loss.

Question 15

Neural networks perform better than linear regression because they can take into account non-linear effects, and interactions between independent variables. For example, a linear regression for the density of a fluid based on temperature will not perform well because freezing or melting is a nonlinear effect.

Furthermore, sometimes independent variables interact with each other. For example, suppose we are modeling the stopping distance of a car as a function of temperature and precipitation. Temperature has no effect on stopping distance if there is no precipitation and the road surface is dry. But, if the road surface is wet, then temperature matters in a nonlinear way. If there is precipitation and temperature below freezing, then that forms ice which increases stopping distance dramatically. This means that the independent variables interact with each other nonlinearly. The hidden layer of the multilayer perceptron is what enables the model to consider the interactions of the independent variables with each other.

Question 16

To find a good set of hyperparameters systematically, we performed grid-search and used 10-fold cross validation over the following parameters:

Hidden Layers: [(10), (20), (10, 10), (10, 20), (20, 10), (20, 20), (10, 10, 10), (10, 10, 20), (10, 20, 10), (10, 20, 20), (20, 10, 10), (20, 10, 20), (20, 20, 10), (20, 20, 20)]
L2-Regularization: [10.0 ** x for x in np.arange(-3,2)]
Activation: ['logistic', 'tanh', 'relu']

Here are the results that we obtained for each dataset:

Dataset	Hidden Layers	L2 Reg	Activation	Train RMSE	Test RMSE
Diamonds	(20, 20)	0.1	ReLU	-723.53826	-624.74864
Gas Emissions	(20, 10)	0.01	ReLU	-0.3931406	-0.559295

Table 4: Best Hyper-parameters for MLP Regressors on each dataset.

We see that Test RMSE for MLP is much lower when compared to standard regression models.

Question 17

The output activation has to be either Linear / Identity as regression problems can have outputs on the real scale. Output activations such as ReLU and TanH for example should not be used as they constrain the values to [0, 1] and [-1, 1] respectively and this will not work as regression outputs are meant to be any value on the real number scale.

Question 18

The depth of a network refers to the number of hidden layers. If we increase the depth of a network too far, then there are the following risks:

1. **Overfitting**: If we have too many layers, then the model can overfit, in which case it may not be generalizable outside of the dataset which it used.

2. **Training Complexity**: As network depth increases, so does increase the time, energy, and computational power required to develop and train the model.

3. **Testing constraints**: As network depth increases, so does the time, energy, and computational power required to run the model after it is developed i.e., testing takes time. This is very undesirable because real-time data is found everywhere and performing real-time data analysis requires networks to be quick and efficient during test time.

4. **Vanishing and Exploding gradients**: Harder to train models due to these problems.

5. **Interpretability**: Deep neural networks are hard to interpret and it is difficult explain the reasons behind a particular phenomenon.

Random forest

Introduction

A random forest model is one where consists of a large number of decision trees that operate as an ensemble. Each individual decision point can give an output based on the probabilities it is programmed to use. The large number of decision trees, due to the law of large numbers, gives you a probability distribution that likely outperforms any of the individual models.

Question 19

In this part, we implemented grid search for the random forest model and using the RMSE as an indicator to determine the performance of the random forest model. We used the model on both the diamond dataset and the gas dataset. We used the following hyperparameter set for the model evaluation.

- Maximum number of features: $1-10$
- Number of trees: $10-50$ in steps of 10.
- Depth of each tree: $1-10$

In the grid search section, we utilize the Pipeline() function to implement the model, and we set the oob_score argument to True so that we can obtain the OOB score after the model is trained and evaluated. In the training section, we generate 10 folds of data for cross validation. The result of the grid search is presented in Table 5.

Fig. 19 and 20 present the result of different max features on the diamond dataset and gas dataset, respectively. We can see that as the max features grow, the RMSE is not guaranteed to increase, it decreases after reaching a certain max feature point. However, we do see the training RMSE converges to a plateau while the testing RMSE is diminishing, and thus it is useless for us to blindly increase the max feature number. A large max feature value may also introduce overfitting problem. The reason behind this is that as the max feature increases, the correlation and connection between trees increases, weakeing the independence of trees and therefore induces overfitting. We should choose the max feature wisely to avoid overfitting and the wast of computational time and resources.

Data	Max feature	No. of trees	Max Depth	Train RMSE	Test RMSE	OOB
Diamond	7	40	9	-525.12	-817.57	0.9783
Gas	3	30	9	-0.3513 (scaled)	-0.5684 (scaled)	0.7456

Table 5: Grid Search result of Random Forest

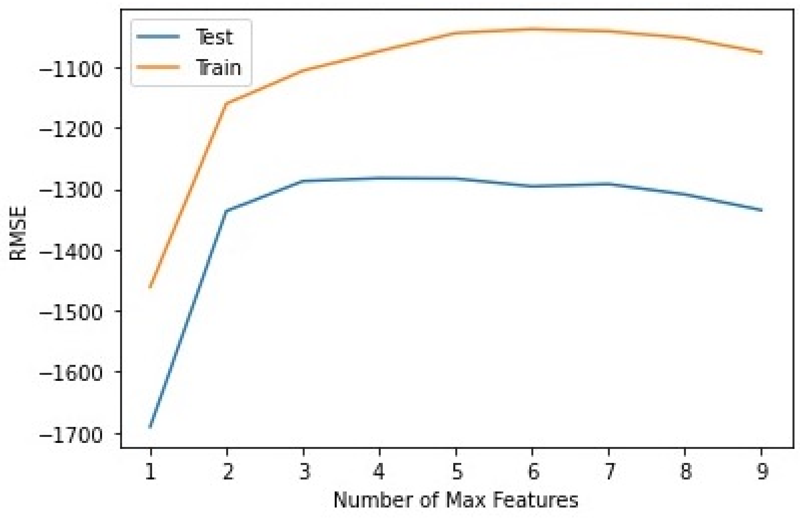


Figure 19: Result of Different Max Features on Diamonds dataset

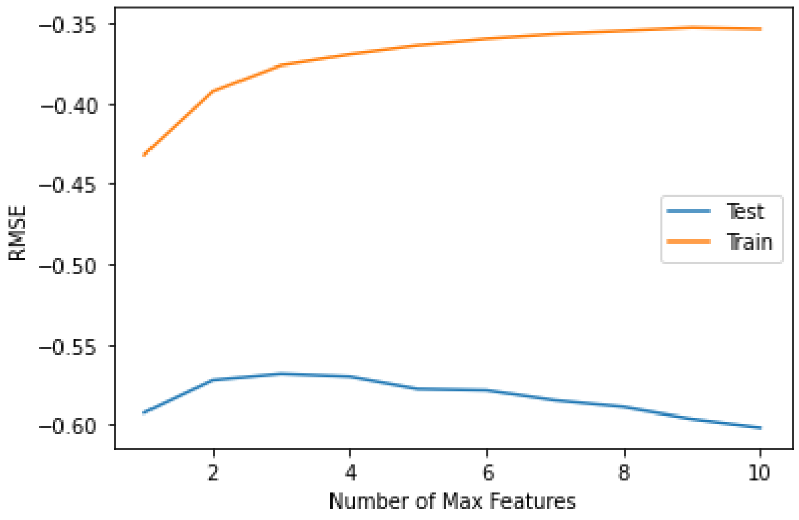


Figure 20: Result of Different Max Features on Gas dataset

Fig. 21 and 22 show the trend of RMSE as more trees are added to the random forest model. We can observe that as more trees are added to the model, the RMSE increases first but then reach a plateau stage where no obvious changes are seen as the number of trees rises. The trees in the forest are drawn independently, and thus do not induce overfitting problem.

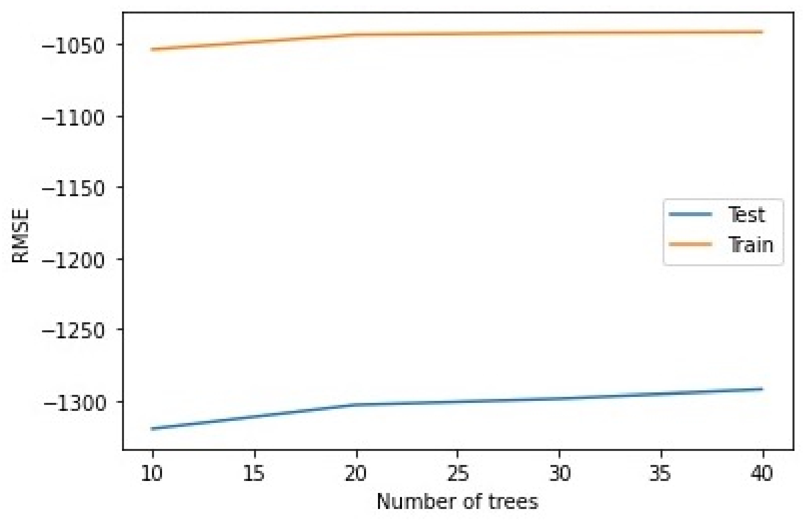


Figure 21: Result of Different Number of Trees on Diamonds dataset

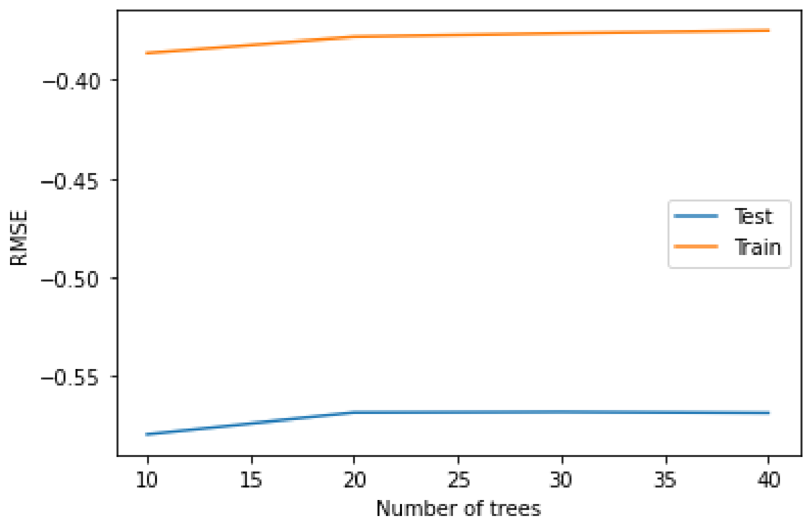


Figure 22: Result of Different Number of Trees on Gas dataset

Fig. 23 and 24 display the monotonically increasing rate of RMSE as the depth of the trees grows. The growth, however, slows down as the depth reaching a certain value, stabilizing the RMSE value. We can conclude that the depth of tree acts as a **regularizer** in that within a certain range, when the depth of tree increases, the fitting performance improves, indicating by the imprved RMSE value. Nevertheless, caution should be taken when choosing the depth value because choosing too large a value will result in overfitting. A large depth allows a tree to fit highly complicated dataset and make decision and that is the reason why we see an improvement as we increase the depth.

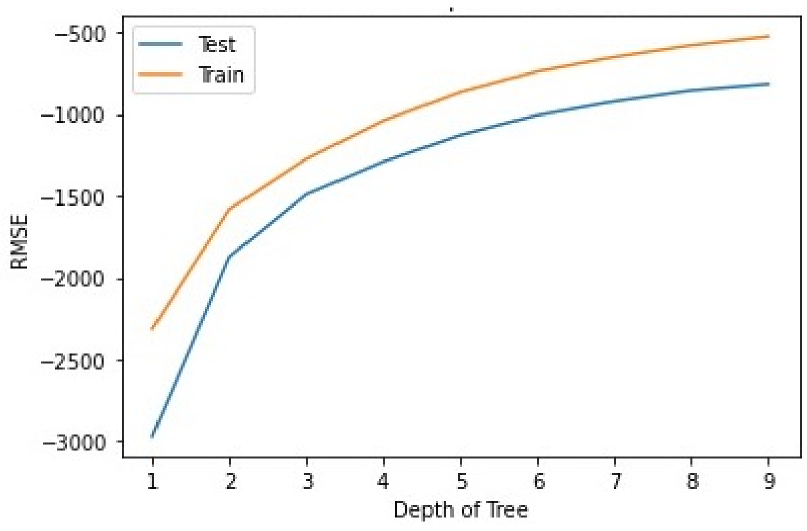


Figure 23: Result of Different Tree Depth on Diamonds dataset

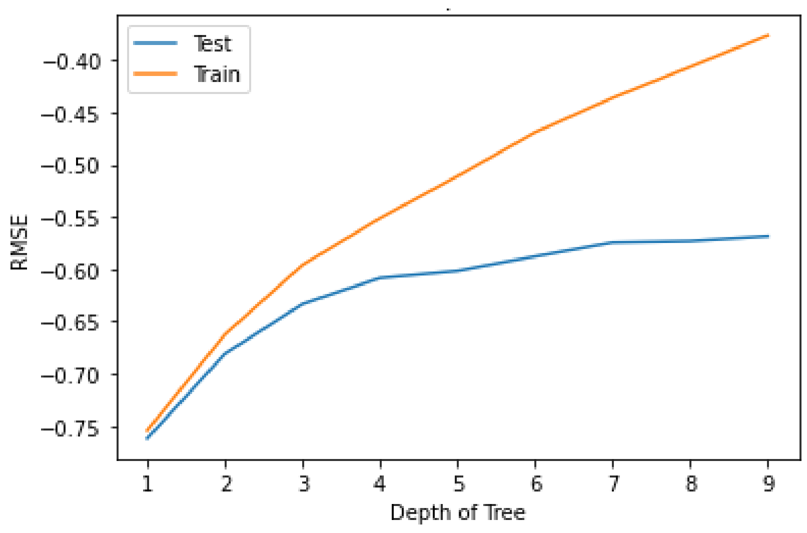


Figure 24: Result of Different Tree Depth on Gas dataset

Question 20

A Random Forest regression model performs well because it incorporates multiple miniature models within it, and takes into account all of their results. The results converge to the expected value with a much higher probability, due to the law of large numbers. The expected values for the coefficient for each variables are much more robust, and much more likely to reflect their true values, and the error bars will be lower, because the mean value is actually a mean of means.

Source: Tony Yiu. Understanding Random Forest: How the algorithm works and why it is so effective. Towards Data Science.

Question 21
We visualized the tree using max depth 4, max feature 7, number of trees 40, k feature 9 (all the features), and the result is presented in Fig. 25.

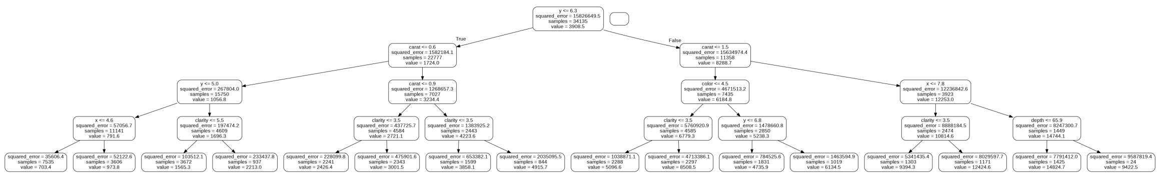


Figure 25: Tree Visualization of Diamond dataset

The feature $y \leq 6.3$ is chosen as the root node, indicating that this feature is the most important feature for the tree to start the branching process and make decisions. The following features are observed in the visualization graph.

1. $carat \leq 0.6$, $carat \leq 1.5$
2. $y \leq 0.5$, $carat \leq 0.9$, $color \leq 4.5$, $x \leq 7.8$
3. $x \leq 4.6$, $clarity \leq 5.5$, $clarity \leq 3.5$, $y \leq 6.8$, $depth \leq 65.9$

According to the five salient features (carat, x, clarity, color, cut) we extracted from the polynomial part, we do see carat, x, and clarity exist in the tree, which means the most important features are similar using different models.

LightGBM and CatBoost

LightGBM is a machine learning algorithm developed by Microsoft. The acronym stands for Light Gradient Boosting Machine. Gradient boosting is a machine learning technique used in regression and classification tasks. It is based on decision tree algorithms and used for ranking, classification, and other related tasks.
Catboost is an open-source library for gradient boosting decision trees that solves for categorical features permutation driven algorithms.**For this part, we worked on the diamonds dataset**. We performed all experiments on the scaled diamonds features without using any feature selection method.

Question 22
For **LightGBM**, we worked with the following hyper-parameters:

1. **boosting_type**: ['gbdt','rf']
2. **num_leaves**: np.arange(20,500,10)
3. **max_depth**: np.arange(1,100,10)
4. **n_estimators**: np.arange(10,1000,100)
5. **reg_alpha**: [10.0**x for x in np.arange(-3,4)]
6. **reg_lambda**: [10.0**x for x in np.arange(-3,4)]
7. **subsample**: np.arange(0.1,1,0.1)
8. **subsample_freq**: np.arange(0,50,5)
9. **min_split_gain**: [10.0**x for x in np.arange(-4,0)]

We set the **bagging frequency** to 1 and **bagging fraction** to 0.1.
For **Catboost**, we worked with the following hyper-parameters:

1. **colsample_bylevel**: np.arange(0.1,1,0.1)
2. **num_trees**: np.arange(10,2000,100)
3. **l2_leaf_reg**: [10.0**x for x in np.arange(-3,4)]
4. **num_leaves**: np.arange(20,500,10)
5. **max_depth**: np.arange(1,100,10)
6. **bagging_temperature**: np.arange(0.1,10,1)

We set the **grow policy** to 'Lossguide'.

Question 23

Bayesian optimization provides an elegant framework to find the global minimum in the least number of steps. It performs tasks using the method of surrogate optimization to estimate an objective function by sampling.

In this question, we used Bayesian optimization to find the optimal hyperparameters for the two boosted tree algorithms. We trained for 20 iterations with 10-fold cross validation. Here are the results that we observed for LightGBM:

Algo	**Booster**	**Leaves**	**Depth**	**Estimators**	**L1**	**L2**	**Sub-sample**	**Sub-sample Freq**	**Min Split Gain**
LGBM	GBDT	390	51	510	0.1	100	0.4	15	0.001

Table 6: LightGBM Optimal hyper-parameters

We obtained a Train RMSE of **-2231.34099** and a test RMSE of **-582.3995** for LightGBM, indicating that the model is capable of performing better over more iterations and has not at all overfit the dataset.

Here are the results that we observed for CatBoost:

Algo	**Bagging Temp.**	**Feature Fraction**	**Num Leaves**	**Num Trees**	**Max Depth**	**L2 Reg**
CatBoost	**5.1**	**0.4**	**130**	**1110**	**11**	

Table 7: CatBoost Optimal Hyper-parameters

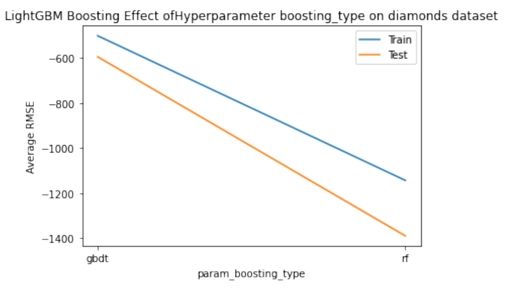
Similar to LightGBM, we obtained a Train RMSE of **-1256.909** and a test RMSE of **-597.22486** for Catboost, indicating that the model is capable of performing better over more iterations and has not at all overfit the dataset.

Question24

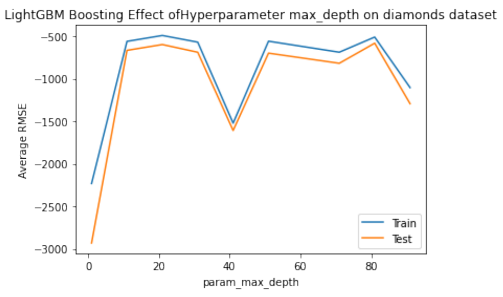
Our observations of the hyper-parameters of LightGBM are as follows:

1. **Boosting Type**: We see that GBDT outperforms RF by a significant margin. Clearly, GBDT helps with performance.
2. **Max Depth**: From the graph, it looks like increasing tree depth will improve performace and also helps in generalization up to a point after which overfitting occurs and hence RMSE becomes worse.
3. **L1 Regularization**: In general, increasing alpha more that a certain value decreases model performance.
4. **L2 regularization**: Opposite to alpha, we see that high regularization value of lambda helps with model generalization and prevents overfitting.
5. **Num Leaves**: This parameter does not seem to be interpretable and zigzags too much.
6. **Num Estimators**: This parameter does not seem to be interpretable and zigzags too much. However, we expected the RMSE value to become better as the number of trees increased up to a point.
7. **Subsampling and Subsampling Frequency**: If subsampling is performed more times, performance of the model in general increases and then decreases indicating overfitting in the later parts.

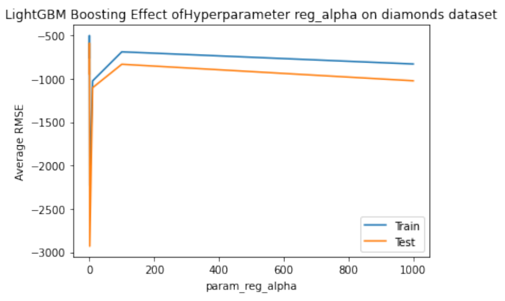
The plots are displayed in the following pages.



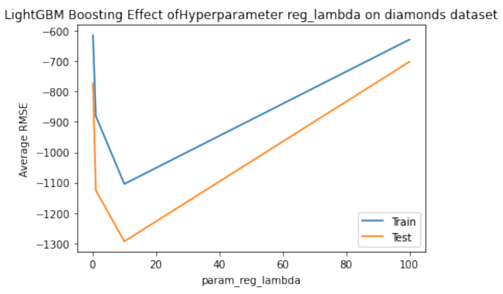
(a) Boosting Type



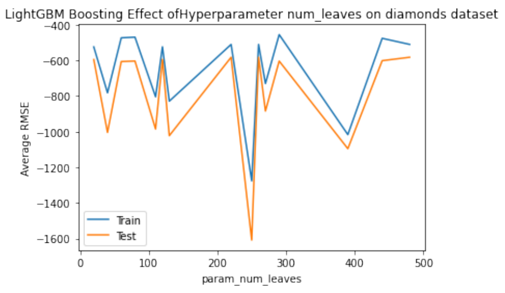
(b) Max Depth



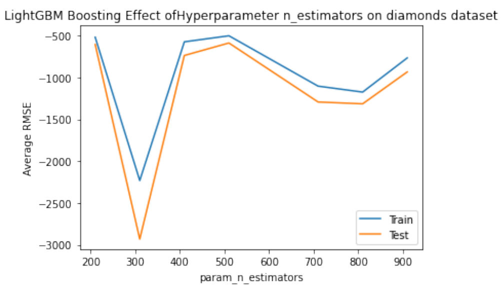
(c) L1 Regularization



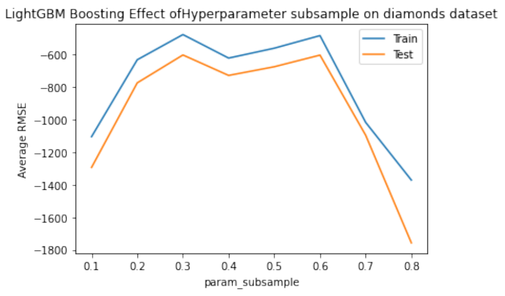
(d) L2 Regularization



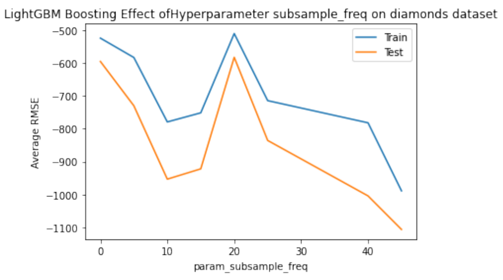
(e) Number of Leaves



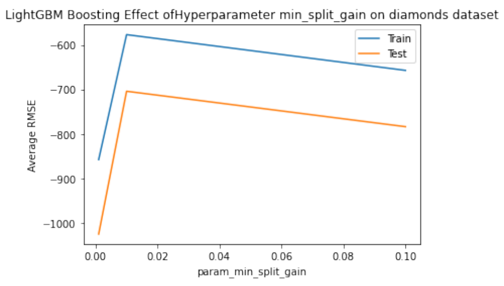
(f) Num Estimators



(g) Subsample

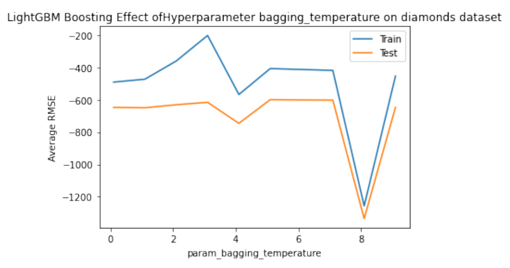


(h) Subsamples Frequency

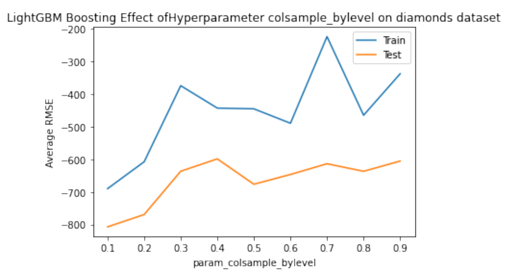


(i) Minimum Split Gain

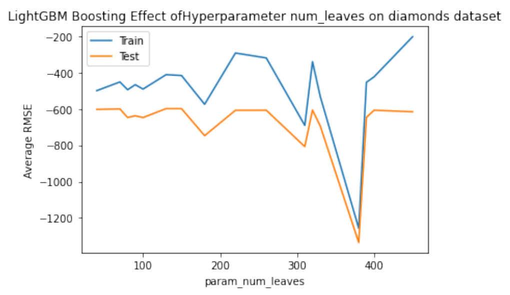
Figure 26: Effect of hyperparameters of LightGBM



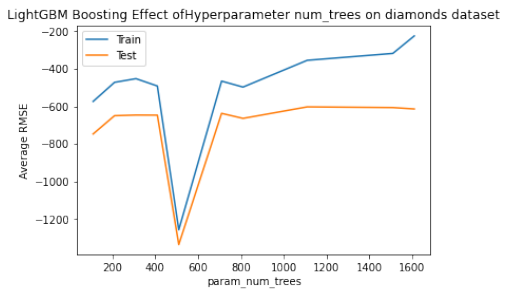
(a) Bagging Temperature



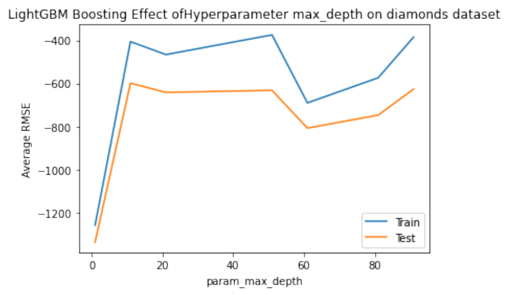
(b) Feature Fraction



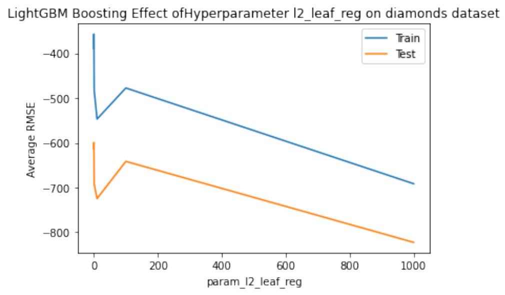
(c) Number of Leaves



(d) Number of Trees



(e) Maximum depth



(f) L2 Regularization

Figure 27: Effect of hyperparameters of CatBoost

Our observations of the hyper-parameters of CatBoost are as follows:

1. **Bagging temperature**: We see that as bagging temperature increases, the RMSE values get worse in general and think that the model is overfitting.

2. **Feature Fraction**: As this increase, performance improves, and so this helps with dealing with overfitting and generalizing the models.

3. **Num Leaves**: The plot does not show any general trend in improvement. However, we expected this to act as a regularization method to prevent overfitting.

4. **Num Trees**: As expected, as the number of trees increases, we see an improvent in model performance as more trees means more generalization .

5. **Max depth**: We see that increasing max depth helps with performance up to a point after which the model overfits and becomes worse.

6. **L2 Regularization**: We see that L2 regularization helps initially, but a lot of regularization impedes model performance and the model starts to underfit.

Evaluation

Question 25

Here are the best results that we obtained over all the models we tested on:

Model	**Dataset**	**Test RMSE**
Linear	Diamonds	-1228.2203
	Gas Emission	-1.516336
Lasso	Diamonds	-1228.22038
	Gas Emission	-1.516336
Ridge	Diamonds	-1228.22031
	Gas Emission	-1.49500
Polynomial	Diamonds	-1205.0451
	Gas Emission	-1.50079
MLP	Diamonds	-624.748
	Gas Emission	-1.243378
Random Forest	Diamonds	-817.57
	Gas Emission	-0.5684
LightGBM	Diamonds	-582.3995
CatBoost	Diamonds	-597.2248

Table 8: Test RMSE over all regression models implemented for both datasets.

We see that in general, as the complexity of the model increases, the models also perform better. However, we need to take care so that models do not overfit by performing regularization. Also, we see that in general, more complex models tend to have a much lower training RMSE than test/validation RMSE. This happens because complex models are capable of fitting to characteristics of the training data that might not be there in the validation data, leading to poor generalization.

Question 26

The out of bag error is not a type of error that can occur. Out of bag error (OOB) is a method of measuring prediction error of random forests, boosted decision trees, and other machine learning models using bootstrap aggregating. It measures whether a model can accurately make predictions that are "out of bag" - meaning outside the realm of the training data it has been given. If a model is designed to predict a dependent variable based on several independent variables, then it will have several specific combinations of values of independent variables as training data. If it is asked to predict the value of the dependent variable based on a different combination of values of independent variables, which it has not seen before, then that is an out of bag prediction. The out of bag prediction error is a measure of the error rate when the model is asked to make out of bag predictions.

R^2 is the coefficient of determination. It measures the proportion of variation in one variable is predicted by variation in another. If the R^2 between the predicted and actual values for a variable is high (that is close to 1), then the model performs very well.

Here are the OOB errors that we obtained for both datasets:

Dataset	**OOB Error**
Diamonds	0.97832286
Gas Emissions	0.74563121

Table 9: OOB Errors for Random Forest Models for both Datsets

Part 2 - TWITTER DATA ANALYSIS

Question 27

In this question, we are asked to report a few statistics. They are as follows:

```
File name: gohawks
Average number of tweets per hour: 292.48785
Average number of followers of users posting the tweets: 2217.92373
Average number of retweets: 2.01320
.............................................................
File name: gopatriots
Average number of tweets per hour: 333.14892
Average number of followers of users posting the tweets: 2121.42173
Average number of retweets: 1.93936
.............................................................
File name: nfl
Average number of tweets per hour: 725.22826
Average number of followers of users posting the tweets: 3512.44995
Average number of retweets: 1.71770
.............................................................
File name: patriots
Average number of tweets per hour: 1475.95549
Average number of followers of users posting the tweets: 3394.45283
Average number of retweets: 1.75207
.............................................................
File name: sb49
Average number of tweets per hour: 2742.97989
Average number of followers of users posting the tweets: 6618.48654
Average number of retweets: 2.11008
.............................................................
File name: superbowl
Average number of tweets per hour: 4811.06670
Average number of followers of users posting the tweets: 7562.66682
Average number of retweets: 2.23092
.............................................................
```

We see that the superbowl is by far the most popular topic, with all 3 statistics being the highest for tweets that have the hashtag superbowl.

Question 28

Here are the plots of number of tweets in hour for the hashtags ”superbowl” and ”nfl”.

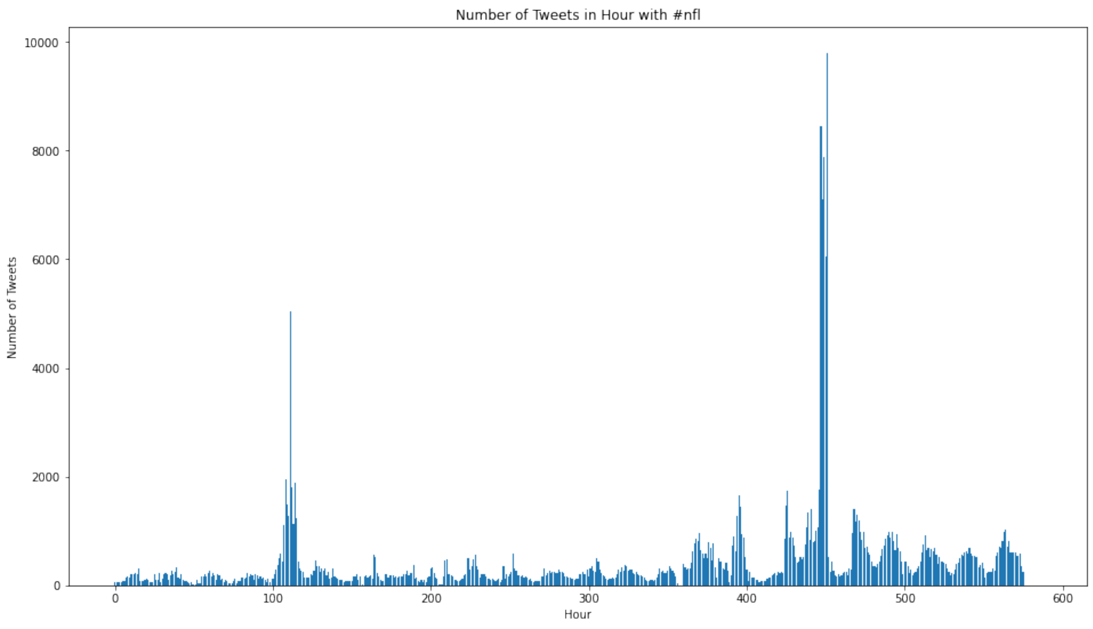


Figure 28: Number of Tweets in hour for #nfl

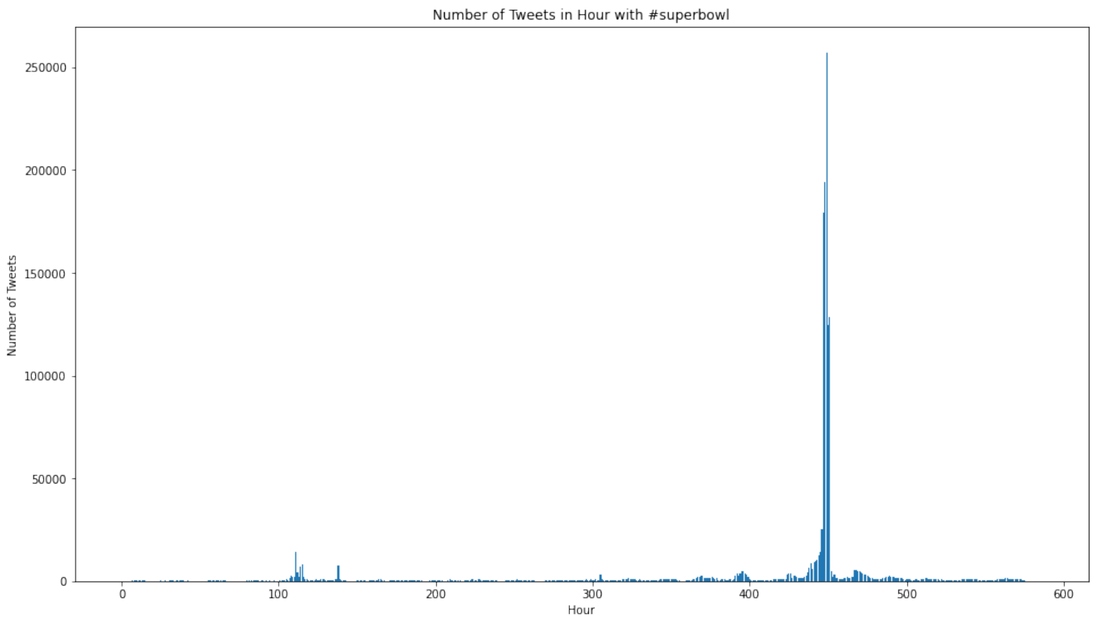


Figure 29: Number of Tweets in hour for #superbowl

We observe a pretty high correlation between the hashtags. We see two spikes, one of which is very large in both hashtags. We conclude that as the superbowl is so popular, people tend to talk a lot more about football in general during superbowl season and so, thehashtag nfl aslo gets used more in that time.

Question 29

Introduction

For this question, we implemented regression models to tackle the task of retweets prediction and classification models to tackle the problem of fan location prediction.

Regression analysis enables us to find the relationships between variables in a real-life system. For example, in a thermal power plant, regression can help us find the strength of the relationship between ambient temperatures, power output, and pollutant output. Using regression analysis, we can find the strength and the signs of the relationships between different variables, to see how strongly they affect each other. Regression analysis has two main components:

1. Dependent Variable: what we're trying to understand or predict. It can be the price of a house, the price of a stock at a certain point in the future, and so on.
2. Independent Variables / Features are what we believe have an impact on the dependent variable and influence its fluctuation.

Regression models primarily find the correlation between the dependent variable and the features and fit a best-fit curve to the data for predictions.
A classification problem is when the output variable is a category and we need to predict which category a test sample belongs to. Classification analysis is a data analysis method used to identify and assign categories to a collection of data to allow for accurate analysis.

Contrary to its name, logistic regression is a popular method that is used to handle classification problems rather than regression problems. Logistic regression is a statistical model that uses a sigmoid function to obtain optimal parameters to maximize the likelihood of correct binary classification. A regularization term is used to penalize the model for using large weights and to prevent overfitting and facilitate generalization.

Data Exploration

In addition to question 27 and 28, we perform more data exploration on the tweets dataset. To get a brief overview of the way the dataset was structured, we printed out the hierarchy of the dataset and the datatypes which is as follows:

'firstpost_data': int

'title': string

'url': string

'tweet': dict_keys(['contributors', 'truncated', 'text', 'in_reply_to_status_id', 'id', 'favorite_count', 'source', 'retweeted', 'coordinates', 'entities', 'in_reply_to_screen_name', 'in_reply_to_user_id', 'retweet_count', 'id_str', 'favorited', 'user', 'geo', 'in_reply_to_user_id_str', 'possibly_sensitive', 'lang', 'created_at', 'filter_level', 'in_reply_to_status_id_str', 'place', 'extended_entities'])

'author': dict_keys(['author_img', 'name', 'url', 'nick', 'followers', 'image_url', 'type', 'description'])

'original_author': dict_keys(['author_img', 'description', 'url', 'nick', 'followers', 'image_url', 'type', 'influence_level', 'name'])

'citation_date': int

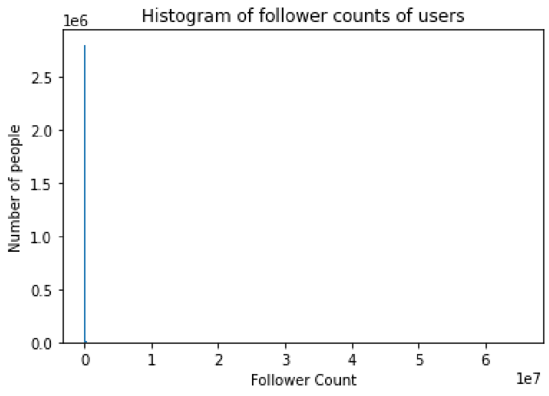
'metrics': dict_keys(['acceleration', 'ranking_score', 'citations', 'peak', 'impressions', 'momentum'])

'highlight': string

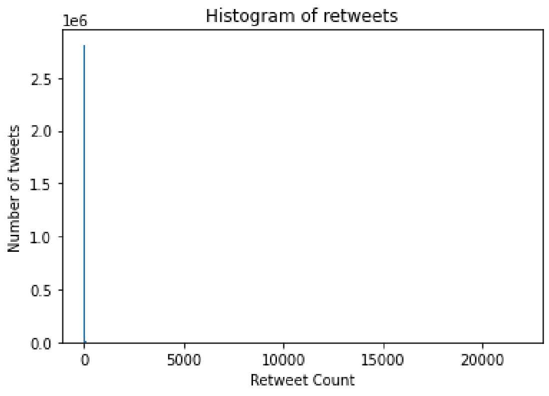
'type': string

'citation_url': string

To get an idea of the distribution of followers for users and the distribution of retweets for tweets, we plotted the following histograms with number of bins = 500.



(a) Histogram of follower counts



(b) Histogram of retweet counts

Figure 30: Histograms

We see that both histograms have very similar distributions. For the follower distribution, we see that most users have very few followers. This is expected as it is realistic that the overwhelming majority of the population do not have many followers. The only users that do have a lot of followers are celebrities that are a minority by a large margin. Similar reasoning holds for the retweets counts histogram. Most tweets just go un-noticed, and very rarely, tweets go viral, mostly because it belonged to a famous celebrity or a player taking part in the upcoming sports event.

Fan Location Prediction

Introduction

Rapid advancement in social media has led to unprecedented amounts of textual data being generated every second. Tweets are short text messages and are often to the point. Utilizing these tweets for location prediction is beneficial as chances are that if a fan lives at a particular location/state/county, it is overwhelmingly the case that the fan also supports the local teams. Knowing the location of a fan can be very useful for local businesses as they can use this information to immediately advertise their merchandise cost-effectively and to also suggest team-related souvenirs. For this part, we perform team prediction using only the text in tweets as our feature with the aim of predicting whether a fan supports Washington or Massachusetts.

Data Representation and Feature Extraction

An important procedure in classifying a corpus of texts is to extract the representing features of each data point. A good representation should remove redundancy of the raw features, and retain defining information for proper classification and to avoid overfitting.

Before performing the feature extraction process, we need to sanitize the text so that the useless information such as HTML artifacts, filler words and numeric terms are excluded. This helps us to downsize the textual entries and remove undesired redundancy, and it improves the feature extraction outcome. Firstly, we read every tweet in the .txt files and append them to a list while simultaneously finding out which team the fan belonged to using keywords in the data. Also, while extracting location information, we create labels for each tweet with 0 being Washington and 1 being Massachusetts. We split the data into training and testing datasets using train_test_split().We then perform lemmatization on the data using nltk's WordNetLemmatizer().

After the lemmatization process, we vectorize our results using CountVectorizer() and convert it into a matrix on which we can apply our classifiers. The parameter stopwords are set to 'English' so that words that are commonly used but contribute trivially to the textual features are removed. In addition, we also perform part of speech (POS) tagging and also use min_df = 3 to ignore words below a frequency of 3. These trivial but important procedures help the classifiers in their task by providing information of better quality.

The resulting matrices have the following shape:

- **Train dataset**: (151955, 18609)
- **Test dataset**: (37989, 18609)

Clearly, the matrix sizes are too big to apply any classifiers because it will be computationally expensive and algorithmically inefficient. Therefore, we need a dimensionality reduction step to retain the information of the matrices while removing redundancy.

Dimensionality Reduction

To get a good value for k, we first use explained variance ratio. The EVR plot is monotonically decreasing and is shown below:

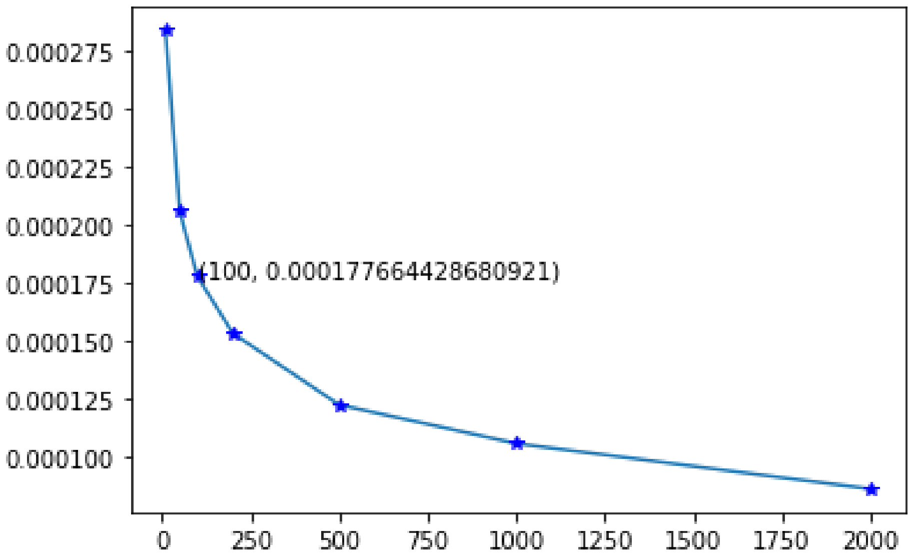


Figure 31: Explained Variance Ratio

From the figure, we see that for a k value of 100, the EVR is high enough and most of the information is contained in 100 components and so, we cohse k = 100 for the following steps.

For the dimensionality reduction step, we used truncated SVD rather than NMF. This is because NMF does not allow negative values in its decomposition, due to which it is very inflexible in higher dimensions. In fact, NMF is computationally expensive in higher dimensions and also does not converge well. After we performed dimensionality reduction on the training dataset, the shape of the dataset is now (151955, 100).

Classification Algorithms

For classification, we experimented with four different models. In addition to the standard Logistic regression model with L1 and L2 regularization, we also used a naive bayse model as well as a new model, the MLPClassifier that is a neural network-based model in sklearn.

For the two logistic regression models, we had to perform grid search to find the optimal value of the regularization penalty. The function find_optimal_lambda() implements this. The results that we got are as follows:

- **L2 regularization**: Lambda = 1 with a validation accuracy of 0.855243819509948.
- **L1 regularization**: Lambda = 1000 with a validation accuracy of 0.8548972294733147.

For all for models, we used sklearn's Pipeline() class to stack all the modules required as follows:

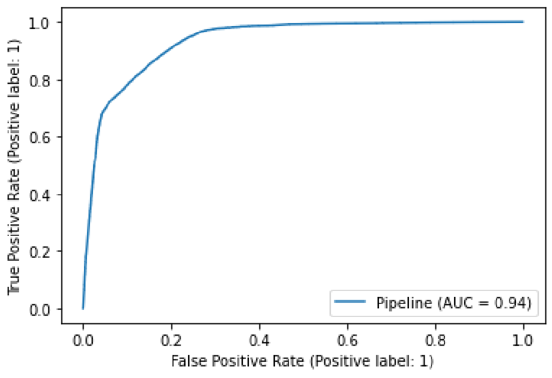
```
pipeline = Pipeline([
('vect', vectorizer), ('tfidf', TfidfTransformer()),
('reduce_dim', TruncatedSVD(n_components=100, random_state=42)), ('clf', classifier),
])
```

Here are the results that we obtained:

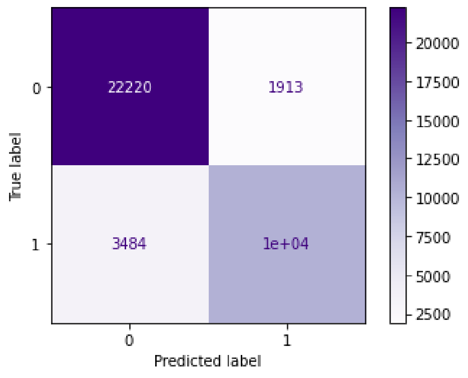
Model	**Optimal Lambda**	**Precision**	**Recall**	**F1-Score**	**Accuracy**	**AUC**
Logistic L2	1	0.84027	0.75173	0.79354	0.85732	0.94
Logistic L1	1000	0.83745	0.75331	0.79316	0.85669	0.94
Naive Bayes	-	0.62734	0.65473	0.64074	0.73221	0.79
MLP Classifier	-	0.81032	0.81986	0.81506	0.86430	0.94

Table 10: Results obtained for our different classifier models

We observe that the MLP classifier outperforms every other model both in terms of F1-Score as well as accuracy. This is expected as MLPs are capable of fitting complex curves and are regularized partially by the amount of data input. As the dataset is pretty large in size, the MLP performs very well. We believe that the MLP will not perform up to its full capacity and has the problem of overfitting if used on small datasets. Here are the confusion matrices and ROC plots for each model:

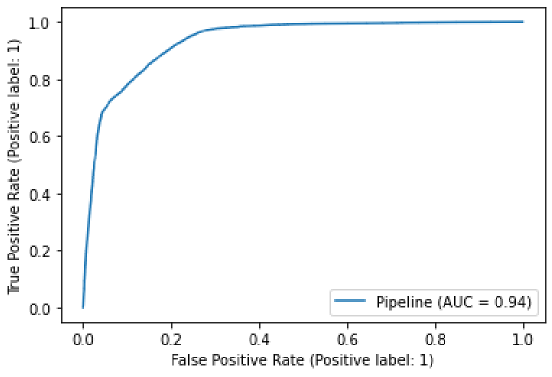


(a) ROC

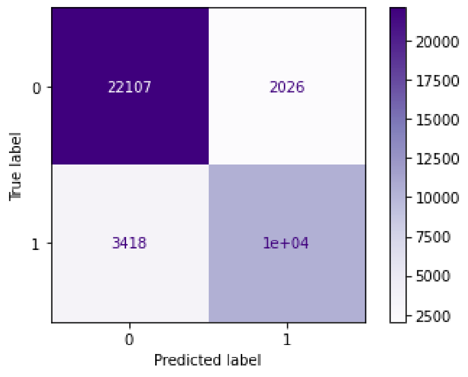


(b) Confusion matrix

Figure 32: ROC and Confusion matrix for Logistic Regression with L2 Regularization

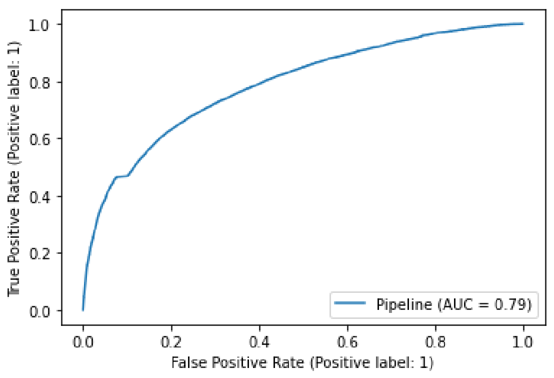


(a) ROC

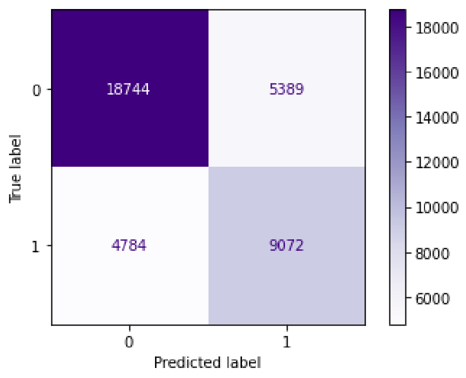


(b) Confusion matrix

Figure 33: ROC and Confusion matrix for Logistic Regression with L1 Regularization

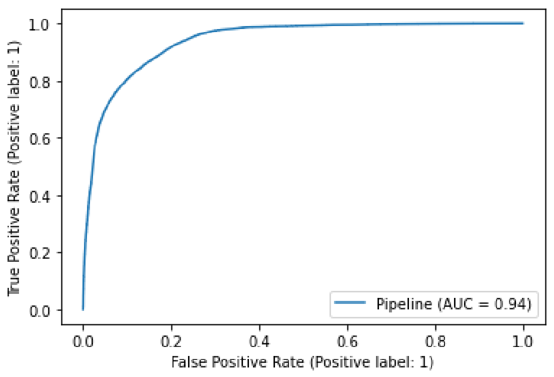


(a) ROC

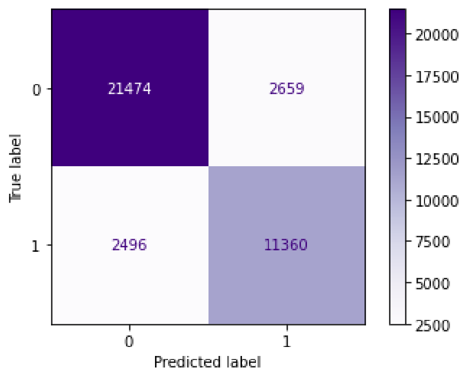


(b) Confusion matrix

Figure 34: ROC and Confusion matrix for Naive Bayes



(a) ROC



(b) Confusion matrix

Figure 35: ROC and Confusion matrix for MLP Classifier

Retweet Count Prediction

Introduction

Understanding and prediction information on social media is a growing communication field as it helps to understand information propagation as well as to control it. An important point to note here is that retweets count will grow exponentially as users retweet an original tweet. This property can be used for marketing, giving companies the power to control their future through free advertisement by word of mouth. Thus, retweet count prediction is a very important tool to have for companies to market their products effectively and so, for this part, we perform retweets count prediction with the aim of finding the number of retweets a tweet will garner given its details.

Data Representation and Feature Extraction

For this part, we initially tried with using multiple features but the time and computation constraint led us to pick three different features that we thought were most useful for regression. They are the following:

1. **Number of followers**: The number of followers that a user who tweeted has is a very important feature. For example, tweets made by celebrities who have millions of followers have a much higher chance of going viral than tweets made by ordinary people.
2. **Time of tweet**: We noticed that tweets were made between 0 hours and 586 hours from the plots made in Question 28. These plots have a very distinct spike and so, it can serve as an important feature. Tweets made closer to the spike that represents the sports event are much more likely to go viral.
3. **Hour of the day**: The final feature we used was hour of the day. Obviously, a tweet made in the middle of the day has a much higher chance of going viral than a tweet made in the middle of the night.

We performed this pre-processing task by first parsing through and gathering the required data from each tweet in all 6 .txt files. This is done in the function parse(). Then, the function localize_time() performs all pre-processing to obtain features 2 and 3. Finally, all of this is converted into a pandas dataframe. The labels are of course the retweet counts for each tweet.

To complete the preprocessing steps, we perform feature scaling, a standardization step to change the values of each column such that the columns have a distribution with mean 0 and variance 1. This is a very important step in training machine learning models as un-standardized datasets can have columns that have values that range from a very small value to very large values which disrupts learning.

For all experiments, we use 5-fols cross validation. We split the dataset into train and test using train_test_split(). Also, as we are using just 3 features due to time and computational power constraints, we do not perform feature selection, but it can easily accommodated into the pipeline.

Regression algorithms

For regression, we experimented with 5 different models: linear regression, lasso regression, ridge regression, random forest, and light GBM. We did try to train MLP as well as polynomial regression models but they took too long to run.

For the random forest and lightGBM models, we performed grid search using GridSearchCV() with 10-fold cross validation to obtain optimal hyper-parameters. Here the parameters used for grid search on **Random Forest** along with the best hyper-parameter combination in parentheses:

- max_features = ['auto', 'sqrt'], **('sqrt')**
- n_estimators = [5,10,50,100], **(100)**
- max_depth = [10, 20, 50], **(10)**

Here the parameters used for grid search on **LightGBM** with Bayesian optimization along with the best hyper-parameter combination in parentheses:

- boosting_type: ['gbdt','rf'], **(rf)**
- num_leaves: np.arange(20,500,10), **(440)**
- max_depth: np.arange(1,100,10), **(71)**
- n_estimators: np.arange(10,1000,100), **(810)**
- reg_alpha: [10.0**x for x in np.arange(-3,4)], **(10.0)**
- reg_lambda: [10.0**x for x in np.arange(-3,4)], **(100.0)**
- subsample: np.arange(0.1,1,0.1), **(0.4)**
- subsample_freq: np.arange(0,50,5), **(10)**
- min_split_gain: [10.0**x for x in np.arange(-4,0)], **(0.0001)**

Here are the results that we obtained for each model:

Model	**Train RMSE**	**Test RMSE**
Linear	-33.138985	-36.19584
Lasso	-33.16894	-36.21048
Ridge	-33.13895	-36.19584
Random Forest	-22.28975	-33.39883
LightGBM	-37.64109	-34.55841

Table 11: Results obtained for our 5 different regression models.

As expected, we see random forest and lightGBM outperforming the standard models.