Quality Prediction in Mining Process

Milestone: Project Report

Group 20

Student 1: Abhinav Somnath Pachpute

Student 2: Ankita Nitin Bandal

617-602-6720

857-693-9325

pachpute.a@northeastern.edu

bandal.a@northeastern.edu

Percentage of Effort contributed by Student 1: 50%

Percentage of Effort contributed by Student 2: 50%

Signature of Student 1: Abhinav Somnath Pachpute

Signature of Student 2: Ankita Nitin Bandal

Submission Date: 04/23/2023

Table of contents

IE 7275: DATA MINING IN ENGINEERING

I. Problem Setting:

Mining is the process of removing valuable geological components from the planet and other celestial bodies. The practice of estimating the grade or quality of minerals or ores based on specific traits or features is known as quality prediction in mining. Rocks and minerals known as iron ores are used to economically extract metallic iron. Natural ore or straight shipping ore refers to ores with very high concentrations of hematite or magnetite having more than 60% iron. However, the typical magnetite iron-ore concentrate contains impurity of 3–7% of silica which is a major source of pollution. Here, we are calculating the percentage of iron ore concentrate.

II. Problem Definition:

Predicting how much impurity is present in the ore concentrate is the major objective of using this data. We can aid the engineers by providing them with early knowledge so they can act because this impurity is monitored every hour. If we can anticipate how much silica and other contaminants are in the ore concentrate, we can help the mining companies.

III. Data Source:

The dataset was obtained from Kaggle.com, a Google LLC subsidiary and a professional online community for data scientists and machine learners.

Topic of the Dataset: - Quality Prediction in a Mining Process

Dataset source: Kaggle

IV. Data Description:

The dataset has 24 columns and 737453 rows. The dataset is at the daily level, with certain columns sampled hourly and the remaining columns sampled every 20 seconds. The first column “date” shows time and date range from 10th March 2017 to 9th September 2017. The second and third columns, "% Iron Feed &% Silica Feed," are quality measures of the iron ore pulp prior to feeding it into the flotation plant. The most important variables that influence ore quality at the

end of the process are “Starch Flow, Amino Flow, Ore Pulp Flow, Ore Pulp pH, and Ore Pulp Density. From column 9 to column 22, "Flotation Air Flow & Flotation Level," we can see the level and air flow inside the flotation columns, both of which influence ore quality. The last two columns, "% Iron Concentrate &% Silica Concentrate," represent the lab's final iron ore pulp quality measurement.

V. Data Collection and Data Processing:

Data set used here is from Kaggle.com and is mining quality prediction data. This dataset is about a flotation plant, it is a process used to concentrate the iron ore and is very common in a mining plant. The attributes are as follows:

Before starting the analysis it’s important to prepare the data and to find the target variables to perform the analysis, it includes data cleaning and data transforming the raw data before the analysis. Here, as the data is in csv format, we have loaded the data in python as a data frame to perform the analysis and displayed the first 5 rows.

```
df_mining.head()
```

	date	% Iron Feed	% Silica Feed	Starch Flow	Amina Flow	Ore Pulp Flow	Ore Pulp pH	Ore Pulp Density	Flotation Column 01 Air Flow	Flotation Column 02 Air Flow	...	Flotation Column 07 Air Flow	Flotation Column 01 Level	Flotation Column 02 Level	Flotation Column 03 Level	Flotation Column 04 Level	Flotation Column 05 Level
0	2017-03-10 01:00:00	55,2	16,98	3019,53	557,434	395,713	10,0664	1,74	249,214	253,235	...	250,884	457,396	432,962	424,954	443,558	502,255
1	2017-03-10 01:00:00	55,2	16,98	3024,41	563,965	397,383	10,0672	1,74	249,719	250,532	...	248,994	451,891	429,56	432,939	448,086	496,363
2	2017-03-10 01:00:00	55,2	16,98	3043,46	568,054	399,668	10,068	1,74	249,741	247,874	...	248,071	451,24	468,927	434,61	449,688	484,411
3	2017-03-10 01:00:00	55,2	16,98	3047,36	568,665	397,939	10,0689	1,74	249,917	254,487	...	251,147	452,441	458,165	442,865	446,21	471,411
4	2017-03-10 01:00:00	55,2	16,98	3033,69	558,167	400,254	10,0697	1,74	250,203	252,136	...	248,928	452,441	452,9	450,523	453,67	462,598

5 rows × 24 columns

FIG-1

The shape of the dataset is generated using the shape function. There are in total 737353 rows with 24 columns. Info function is used to display the information about the data frame. Here the date format is changed to the datetime format and Day, Month and Year are separated for further analysis. The Year column further dropped as it had only one unique value. Here we have converted all the columns into float, except date which is in datetime. Further the dataset is checked for null values missing percent for each column. It is observed that there are no null values present in the dataset. All the duplicates values are dropped and

the shape of the dataset is displayed before and after dropping the duplicate values. A coefficient close to 1 shows a strong positive correlation between the two variables. Further we have dropped all columns having correlation more than 0.7. The final correlation plot is shown below.

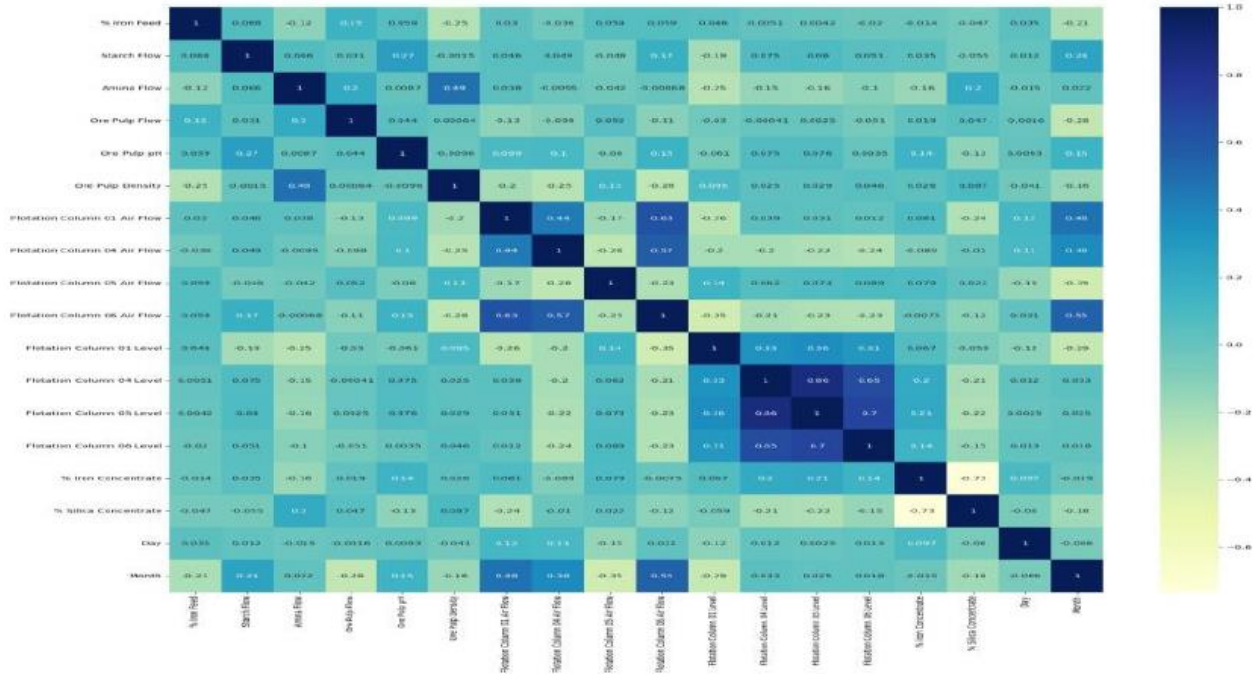


FIG-2

VI. Data Exploration and Visualizations:

We used Heat Map to represent Correlation between all the attributes in the Mining Dataset. The Flotation Column 01 Air Flow has the strongest positive correlation of +0.96 with the Flotation Column 03 Air Flow. The Percentage Silica Feed has the strongest negative correlation of -0.97 with Percentage Iron Feed.

FIG-3

A Pair Plot is a type of data visualization in which the pairwise relationships between different variables in a dataset are represented by a grid of subplots. Hence, the P air Plot above represents all the attributes in the dataset with their pairwise relationship. We represented the histogram for the attributes, which is representing the Frequency on the y-axis for the attributes of Starch Flow, Anima Flow, Ore Pulp Flow, Ore Pulp pH and Ore Pulp Density. Ore pulp Flow of 400 is having highest frequency of 110000.

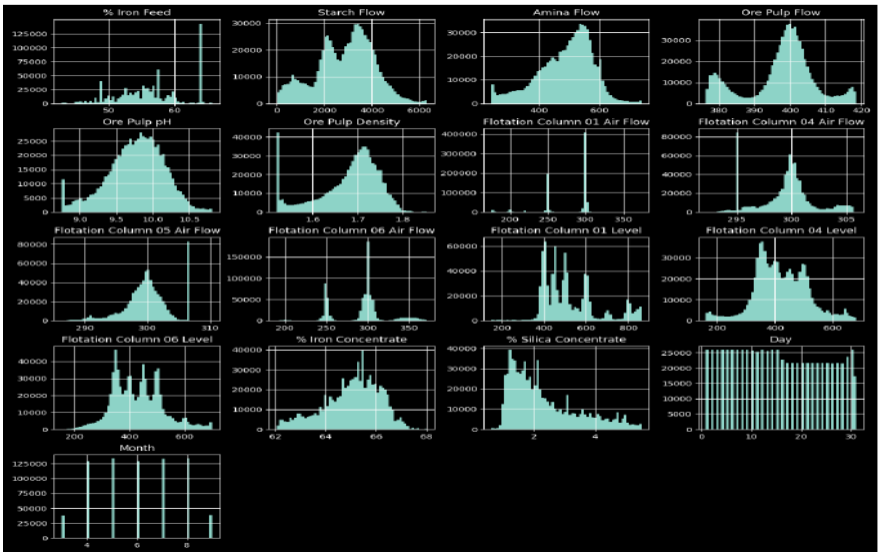


FIG-4

Flotation Column 01 Air Flow to Flotation Column 07 Air Flow is visualized in the form of Histogram. As it can observed that the setting of 300 Flotation Value has the highest frequency in all the attributes from Flotation Column 01 Air Flow to Flotation Column 07 Air Flow. We used Histograms for Flotation Column Levels from one to seven. The Flotation Column values keep on fluctuating between 200 to 600.

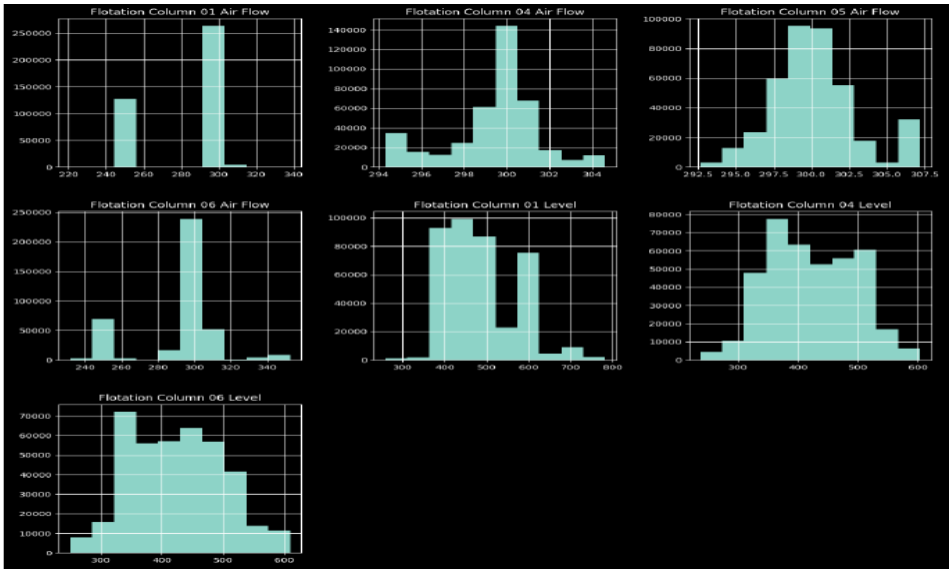
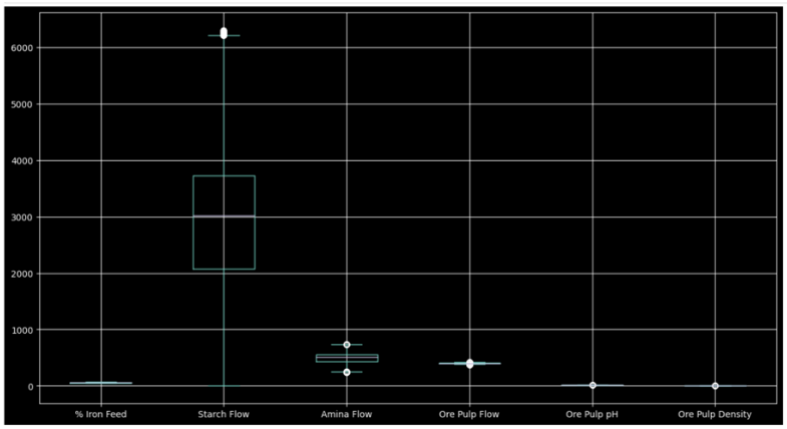
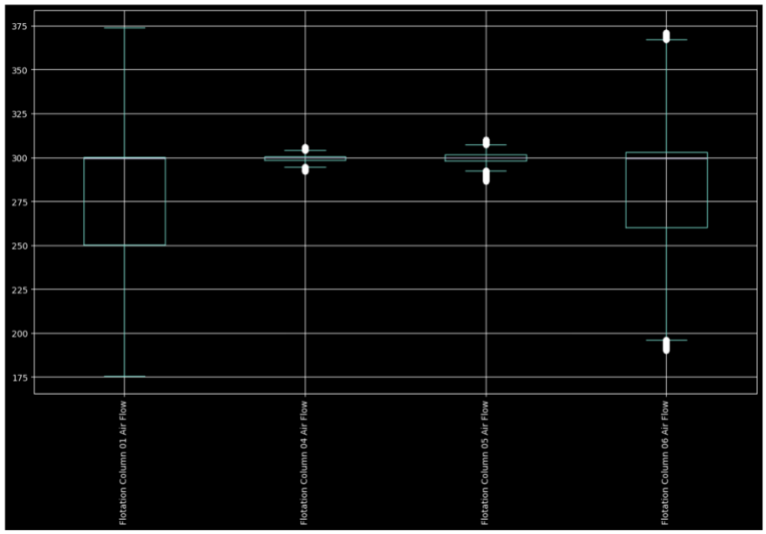


FIG-5

The highest percentage is observed to be 67.1 while the lowest is 62.8, so it can be inferred that the Iron concentration of 65.4 is achieved maximum times in the mining process. The highest percentage is observed to be 4.5 while the lowest is 0.6, so it can be inferred that the Silica concentration of 1.3 is achieved maximum times in the mining process. We used box plots and scatter plots to represent the relationship between Iron Feed Percentage Lag and Iron Feed. The scatter plot is observed to have a positive correlation trend line. Hence, it can be inferred that there is a strong positive correlation between Iron Feed Percentage Lag and Iron Feed**.** The scatter plot is observed to have a negative correlation trend line. Hence, it can be inferred that there is a strong negative correlation between Silica and Iron Feed**.**


6000
5000
4000
3000
2000
1000
0
% Iron Feed
Starch Flow
Amina Flow
Ore Pulp Flow
Ore Pulp pH
Ore Pulp Density


375
350
325
300
275
250
225
200
175
Flotation Column 01 Air Flow
Flotation Column 04 Air Flow
Flotation Column 05 Air Flow
Flotation Column 06 Air Flow

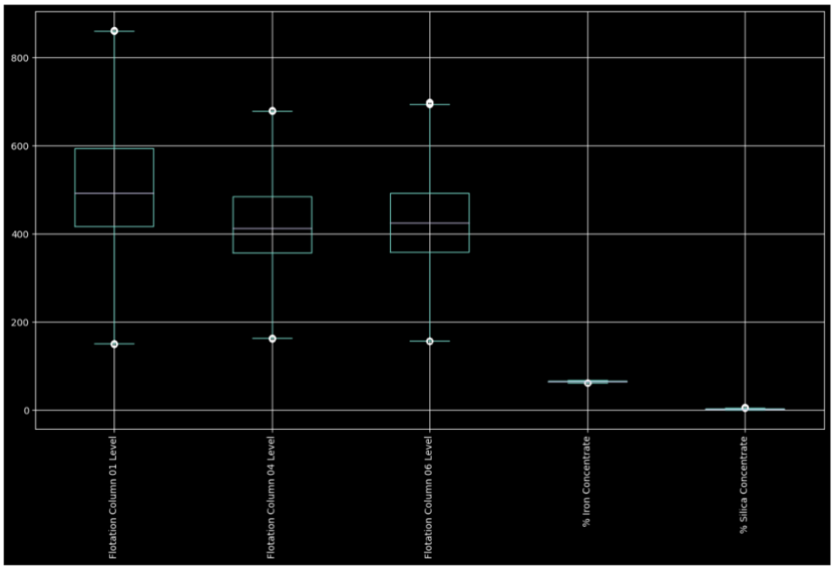

800
600
400
200
0
Flotation Column 01 Level
Flotation Column 04 Level
Flotation Column 06 Level
% Iron Concentrate
% Silica Concentrate

FIG-6

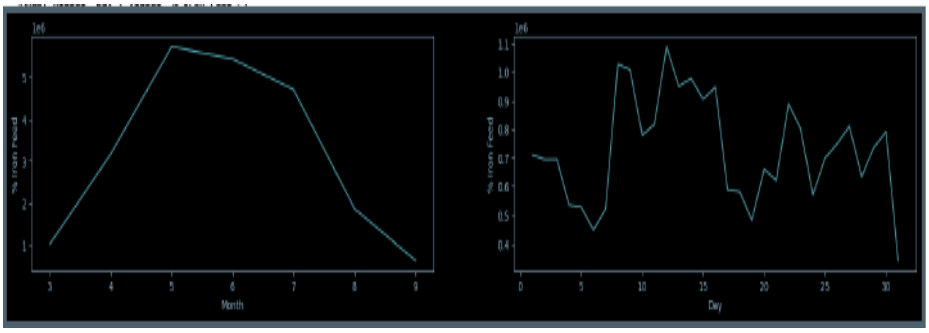


FIG-7

It is observed that the Percentage of Iron Feed is at the highest between Mid-April to Early July, hence mining is advisable in this period for Iron. It is also observed that the Percentage of Silica Feed is at the highest between Late March to July, hence mining is advisable in this period for Silica.

VII. Model Exploration and Model Selection:

1. Linear Regression.

The code provided creates a linear regression model and fits it to the training data using the LinearRegression () function from scikit-learn. The trained model is then used to make predictions on the test data using the predict () function. The performance of the model is evaluated using several metrics including Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Percentage Error (MPE), and Mean Absolute Percentage Error (MAPE). The Accuracy for the linear regression model is calculated to be **58.79%**

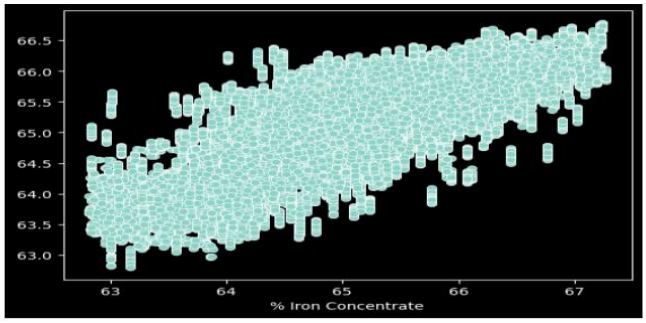


FIG-8

2. Random Forest

The model shown below creates a Random Forest regression model with 100 estimators and fits it to the training data using the Random Forest Regression() function from scikit-learn. The trained model is then used to make predictions on the test data using the predict () function. The accuracy of the model is calculated using the R-squared score, which indicates that the model has an accuracy of 99.76%. The Accuracy for the Random Forest model is **97.54%**

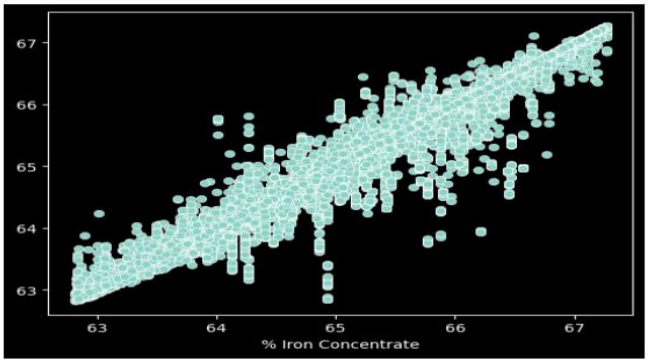


FIG-9

3. KNN :-

The mean_squared_error() function from scikit-learn is used to calculate the MSE. The r2_score() function is used to calculate the R- squared score, which is a measure of how well the model fits the data. Finally, a correlation matrix is generated to show the correlation between the actual and predicted values. Overall, the model demonstrates how the KNN algorithm can be used to build a regression model and evaluate its performance. The accuracy of the model is calculated using the R-squared score, which indicates that the model has an accuracy of 92.6%. The Accuracy from KNN modeling is calculated to be **92.90%.**

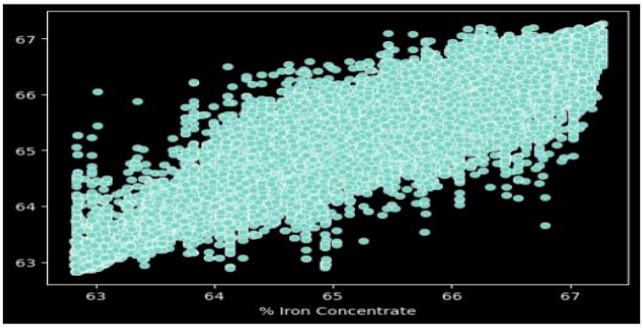


FIG-10

4. XGB Regression (Extreme Gradient Boosting) :

In a GBR model, decision trees are constructed sequentially, with each tree attempting to correct the errors made by the previous tree. The algorithm starts with a simple model, such as a single decision tree, and then builds additional trees to improve the accuracy of the model. The output of the final model is the sum of the predictions of all the decision trees. The Accuracy for the XGB Regression model is calculated to be **95.85%**

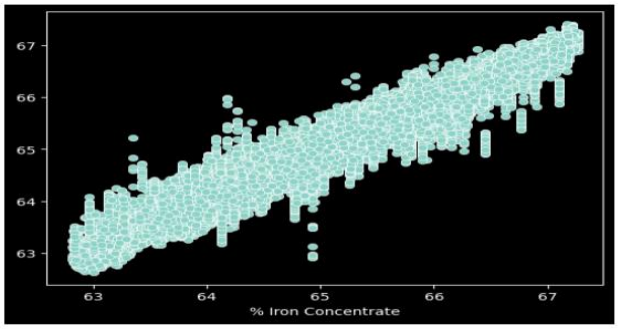


FIG-11

5. Decision Tree

The model builds a decision tree regression model using the DecisionTreeRegressor () function from the scikit-learn library in Python. The model is fit on the training data using the fit() function with the specified maximum depth and random state. After fitting the model, predictions are made on the test set using the predict () function. The accuracy of the model is evaluated using several metrics, including mean squared error, root mean squared error, mean absolute error, mean percentage error, and mean absolute percentage error. The code then creates a dataframe to compare the predicted and actual values, along with their residuals. The Accuracy for the Decision Tree model is calculated to be **61.55%**

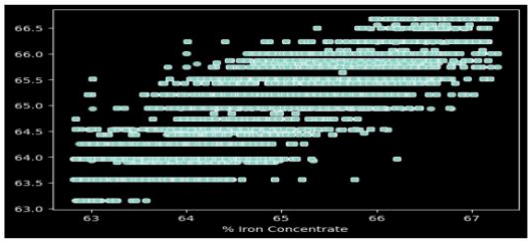


FIG-12

6. Neural Networks

This code defines, compiles, and trains a simple neural network using the Keras Sequential API. The network has three layers with 16, 8, and 1 neuron, respectively, and uses ReLU activation functions for the first two layers and linear activation for the last layer. It is compiled with the mean squared error loss function and the Adam optimizer. The network is trained for 20 epochs using a batch size of 500 and a validation split of 0.2. Finally, predictions are made on the test data, and the R-squared score is calculated and printed, yielding an accuracy of approximately **45.15%.**

7. SVR

This code creates a copy of a preprocessed dataset, selects the first 10,000 rows, and splits it into training and test sets with a 60/40 ratio. It then standardizes the features using StandardScaler. A Support Vector Regression (SVR) model with a linear kernel is created and fitted to the training data. The trained model is then used to make predictions on the test data. A comparison of predicted and actual values is performed, along with the calculation of residuals. Finally, the R-squared score is calculated and printed, resulting in an accuracy of approximately **84.77%** for the Support Vector Regression model.

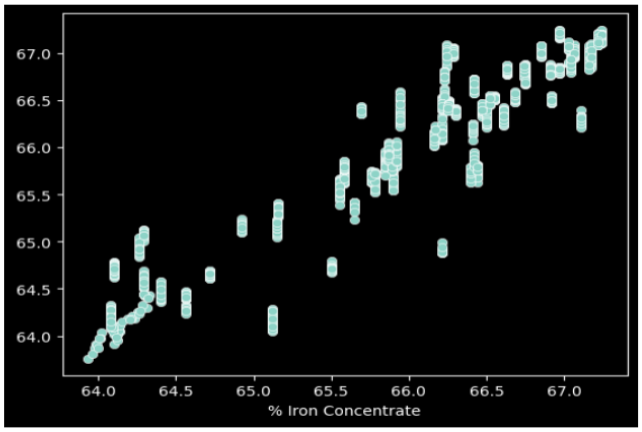
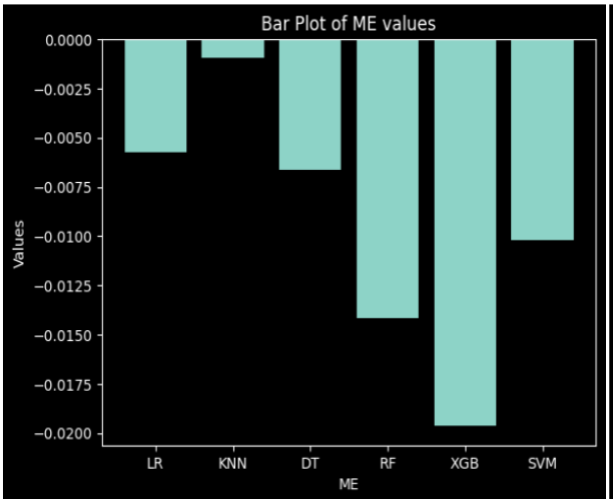
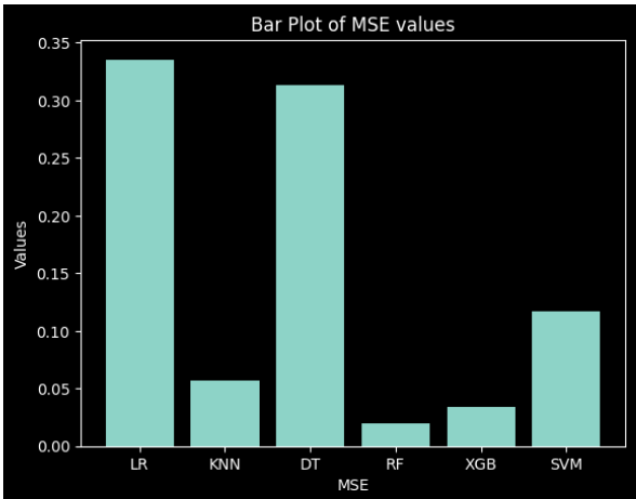
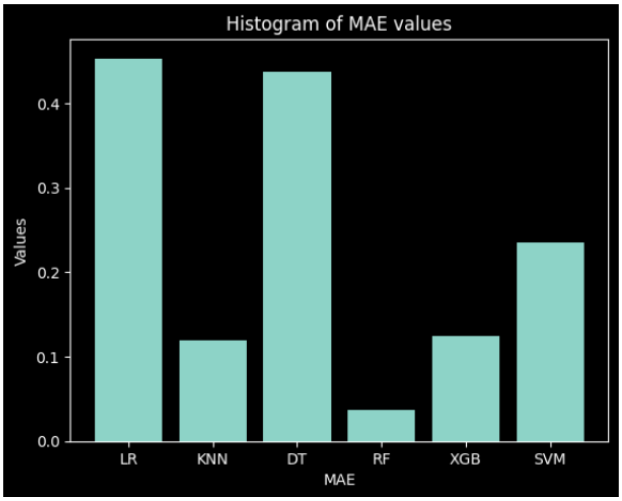


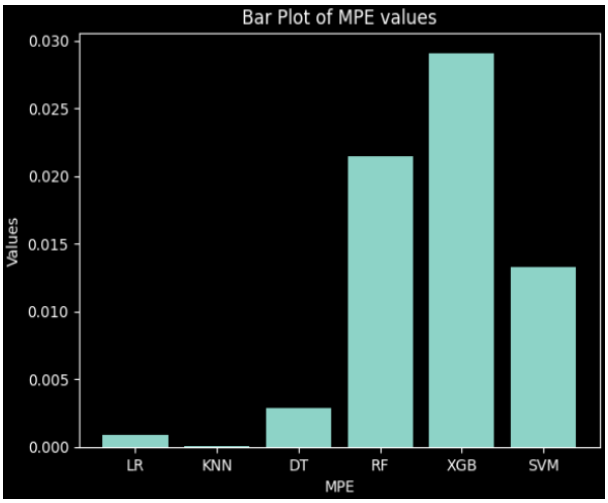
FIG-13

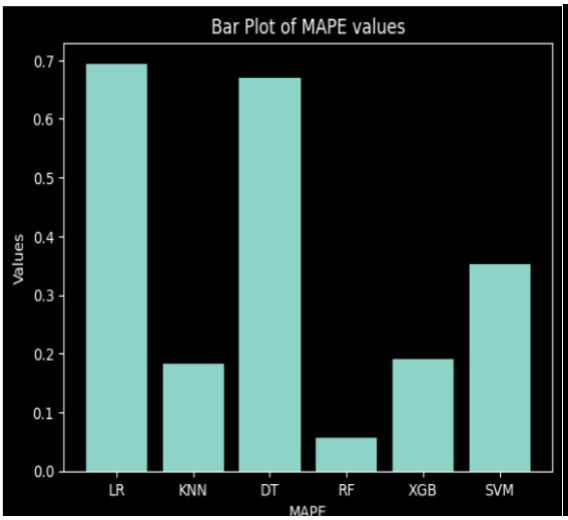
Performance Visualization

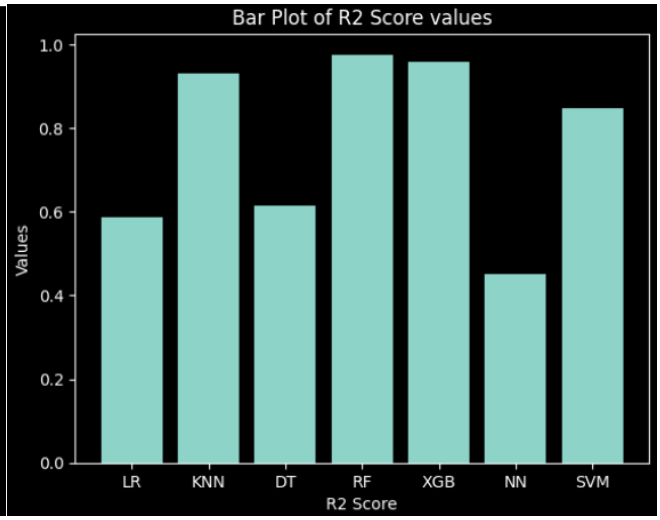




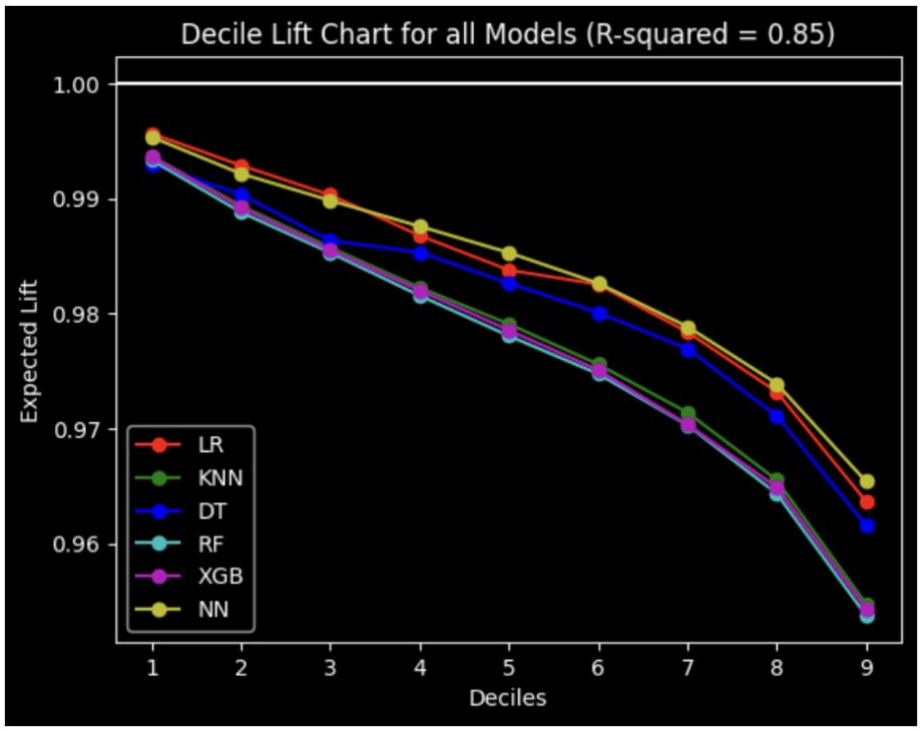








DECILE LIFT CHART



VIII. Implementation of Selected Models:

This report discusses the implementation of the random forest model, its performance, and the rationale behind selecting the Random Forest model. The implementation of the models which gave us the highest accuracy is given below: -

1. Random Forest

The model shown below creates a Random Forest regression model with 100 estimators and fits it to the training data using the RandomForestRegressor() function from scikit-learn. The trained model is then used to make predictions on the test data using the predict() function.

The performance of the model is evaluated using several metrics including Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean

Percentage Error (MPE), and Mean Absolute Percentage Error (MAPE).
The regressionSummary() function is not a built-in function in scikit-learn, but it is likely a custom function used to display the regression statistics. The mean_squared_error() function from scikit-learn is used to calculate the MSE.

Finally, a correlation matrix is generated to show the correlation between the actual and predicted values. The R-squared value (97.54%) indicates a high accuracy of the Random Forest model. Additionally, the correlation between actual and predicted values is close to 1, suggesting a strong relationship between the predicted and actual values. These results demonstrate that Random Forest is a suitable choice for this problem.
Overall, the model demonstrates how the Random Forest algorithm can be used to build a regression model and evaluate its performance. The accuracy of the model is calculated using the R-squared score, which indicates that the model has an accuracy of **97.54%**. It is important to note that the accuracy of the model may vary depending on the specific dataset and problem being solved, and that the choice of hyperparameters, such as the number of estimators, can have a significant impact on the accuracy of the model. The Accuracy for the Random Forest model is calculated to be **97.54%**

Model Selection:

The Random Forest model was chosen over other models due to its higher accuracy. The Random Forest model had an R-squared score of **97.54%,** while the XGB Regression model, which was the second-best model with the accuracy, had an R-squared score of **95.85%**. This indicates that the Random Forest model is better at capturing the underlying patterns in the data and making accurate predictions.

Reasons for Higher Accuracy in Random Forest:

Ensemble Learning: Random Forest is an ensemble learning method that constructs multiple decision trees, which are then combined to make a more accurate prediction. This leads to improved model performance and robustness compared to single decision trees or other models.

Handling High-Dimensional Data: Random Forest is effective at handling high-dimensional datasets and complex relationships between input features and the target variable. In contrast, other models can be sensitive to high-dimensionality and may suffer from the "curse of dimensionality".

Robustness to Missing Data and Outliers: Random Forest models can handle missing data and outliers effectively, which can lead to improved prediction accuracy. Other models can be sensitive to outliers, as they rely on distance-based measures to make predictions.

Feature Importance: Random Forest models can provide information about the importance of individual features in the prediction, which can be useful for feature selection and interpretation.

Conclusion for implementation of Random forest method:

Based on the results obtained from the implementations, the Random Forest model is chosen for this dataset and problem due to its higher accuracy of **99.90%.** The Random Forest model also excels at handling high-dimensional data, coping with missing data and outliers, and providing feature importance assessment.

IX. Performance Evaluation and Interpretation:

The top 2 models selected are:

Random Forest

XGB Regression

1. **Random Forest**

The model shown below creates a Random Forest regression model with 100 estimators and fits it to the training data using the RandomForestRegressor() function from scikit-learn. The trained model is then used to make predictions on the test data using the predict() function. The performance of the model is evaluated using several metrics including Mean Squared Error

(MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Percentage Error (MPE), and Mean Absolute Percentage Error (MAPE). Finally, a correlation matrix is generated to show the correlation between the actual and predicted values. Overall, the model demonstrates how the Random Forest algorithm can be used to build a regression model and evaluate its performance. The accuracy of the model is calculated using the R-squared score, which indicates that the model has an accuracy of **97.54%**.

2. **XGB Regression :-**

The XGB Regression model is implemented using the XGBRegressor() function from the XGBoost library. The model is set up with 1000 estimators and is fit to the training data using the fit() method. To prevent overfitting, early stopping rounds are set to 5, and an evaluation set is provided using the test data. The model is trained quietly, with verbose set to False. The performance of the XGB Regression model is evaluated using the R-squared score. This metric is calculated using the r2_score() function from the scikit-learn library. The R-squared score of the model is found to be approximately 95.85%, indicating a high level of accuracy. It is important to note that the accuracy of the model may vary depending on the specific dataset and problem being solved, and the choice of hyperparameters, such as the number of estimators, can have a significant impact on the model's performance.

How Random Forest model work better than XGB Regression model :-

Random Forest: The Random Forest model was implemented using scikit-learn's RandomForestRegressor class with 100 estimators. The model was fitted to the training data and used to make predictions on the test data. The R-squared score was calculated to be 97.54%.

High Accuracy: Random Forest model achieved the highest accuracy (R-squared score) among all the models tested, indicating that it is better at capturing the underlying patterns in the data. Low RMSE and MAE: Random Forest model has lower RMSE and MAE values compared to other models, which means it has a smaller average error in its predictions.

Ensemble Learning: Random Forest is an ensemble learning method that combines multiple decision trees to make a more accurate and robust prediction. Random Forest model is better at capturing the underlying patterns in the data and making accurate predictions.

X. Result:

The results of this study show that machine learning techniques, particularly the Random Forest and XGB Regression models, can be effectively applied to predict the quality of mining data. These models can provide valuable insights for mining companies, enabling them to make more informed decisions regarding their operations, optimize resource allocation, and improve safety. By implementing these models, the mining industry can benefit from increased efficiency, cost savings, and reduced environmental impact.

XI. Impact of the Project Outcomes:

Enhanced decision-making: By using machine learning models to predict the quality of mining data, mining companies can make better decisions regarding the allocation of resources, equipment maintenance schedules, and safety measures, which can lead to increased efficiency and cost savings.

Cost reduction: The use of machine learning models can help reduce the costs associated with manual data quality assessment and enable more efficient allocation of resources, resulting in cost savings for mining companies.

Reduced environmental impact: By optimizing mining processes and reducing waste, companies like ‘BHP, Barrick Gold, and Anglo American’ can decrease their environmental footprint and adhere to sustainability standards.

Improved safety: Implementing quality prediction can help companies like ‘Southern Copper, Hecla Mining Company, and Cleveland Cliffs Inc.’ identify potential safety hazards before they become critical, reducing the risk of accidents, and ensuring a safer working environment.