Task 2: Create a Final Report

### **1.** Final Report: Stock Price Prediction Using Linear Regression

#### Introduction

In this project, I focused on predicting the stock price of Amazon using a dataset that contains various economic indicators and historical stock prices. The primary goal was to build a reliable linear regression model that can accurately predict future prices of Amazon stock based on the given data.

To begin, I carefully examined the dataset to understand its structure and identify any potential issues, such as missing values, outliers, and multicollinearity among the features. Handling these issues is crucial because they can significantly impact the performance and accuracy of the model.

Next, I applied several preprocessing techniques, including data imputation, outlier handling, and scaling, to prepare the data for modeling. I then used Principal Component Analysis (PCA) to reduce the dimensionality of the dataset, which helped in retaining only the most important information while eliminating redundant features.

The core of the project involved building and evaluating a linear regression model. I experimented with different variations, including Ridge and Lasso regression, to address the issue of multi-collinearity and to improve the model's performance. Additionally, I performed cross-validation to ensure that the model generalizes well to unseen data.

After building the model, I analyzed the coefficients of the linear regression model to determine the importance of each feature. This analysis helped in understanding which factors most significantly influence the Amazon stock price.

The findings and model performance are discussed in detail, highlighting the challenges faced during the project and potential improvements that can be made in future work. This report aims to provide a clear and comprehensive overview of the project, making it accessible to both technical and non-technical stakeholders.

**Data Preprocessing and Feature Engineering**

Before diving into model building, I focused on preparing the dataset to ensure it was clean and ready for analysis. This phase, known as data preprocessing and feature engineering, is crucial because the quality of the input data directly affects the performance of the machine learning model.

**1. Handling Missing Data**

The first step was to address any missing values in the dataset. Missing data can lead to inaccurate predictions if not handled properly. I used a technique called imputation to fill in the missing values. Specifically, I applied the median imputation method, which replaces missing values with the median of the respective column. This method is less sensitive to outliers compared to mean imputation and helps in maintaining the integrity of the data.

**2. Removing Commas and Converting Data Types**

In this dataset, some numeric columns contained commas, which needed to be removed for proper numerical analysis. I replaced these commas with empty strings, effectively cleaning the data. After this, I converted all columns to numeric data types to ensure that the dataset was ready for mathematical operations and statistical analysis.

**3. Outlier Detection and Handling**

Outliers are extreme values that deviate significantly from the rest of the data. These can skew the results and lead to inaccurate model predictions. To detect and handle outliers, I used a method called Winsorization. This technique adjusts the extreme values by capping them at a specified percentile. In this case, I applied Winsorization using the Interquartile Range (IQR) method, which helped in reducing the impact of outliers without removing any data points.

**4. Feature Scaling**

After handling outliers, the next step was to scale the features. Feature scaling is essential because it ensures that all features contribute equally to the model. I used the RobustScaler, a scaling technique that is particularly effective in dealing with outliers. This scaler transforms the data by removing the median and scaling it according to the interquartile range, making the dataset more robust to outliers.

**5. Dimensionality Reduction using PCA**

With the data cleaned, scaled, and imputed, I performed Principal Component Analysis (PCA) to reduce the dimensionality of the dataset. High-dimensional data can lead to overfitting, where the model performs well on training data but poorly on unseen data. PCA helps in selecting the most important features that capture the majority of the variance in the data while discarding the less important ones. By retaining 99% of the variance, I was able to reduce the number of features while still keeping most of the important information intact.

**6. Addressing Multicollinearity**

Multicollinearity occurs when two or more features are highly correlated, leading to redundancy in the model. This can cause issues in linear regression models, as it makes it difficult to determine the independent effect of each feature on the target variable. I identified multicollinear features and addressed them by using techniques like Ridge and Lasso regression, which add penalties to the model coefficients, helping to reduce the impact of multicollinearity.

**7. Preparing Data for Modeling**

Finally, I split the data into training and testing sets, ensuring that the model would be evaluated on unseen data to test its generalization ability. The training set was used to build the model, while the testing set was reserved for evaluating the model's performance.

**Model Building and Evaluation**

After preparing the data, I moved on to the crucial steps of building and evaluating the model. The goal was to create a model that could accurately predict Amazon stock prices based on historical commodity prices and volumes.

**1. Building the Linear Regression Model**

I started by choosing a Linear Regression model for this project. Linear Regression is a statistical method that models the relationship between a dependent variable (in this case, the Amazon stock price) and one or more independent variables (the features we processed earlier).

To build the model, I used the training data, which is a portion of the dataset reserved for teaching the model how to make predictions. The model learns by finding the best-fit line that minimizes the difference between the actual and predicted values. This line is determined by calculating coefficients for each feature, which represent their impact on the target variable. For instance, a higher coefficient means the feature has a more significant effect on the stock price.

**2. Model Evaluation on Training Data**

Once the model was trained, I evaluated its performance on the training data. I used several key metrics to assess how well the model was learning:

* **R² Score**: This metric indicates how well the independent variables explain the variance in the dependent variable. An R² score close to 1 means the model fits the data well. On the training data, the R² score was 0.92, indicating that the model could explain 92% of the variance in the stock prices.
* **Mean Absolute Error (MAE)**: MAE measures the average magnitude of errors between predicted and actual values. For the training data, the MAE was 7.05, meaning that, on average, the model’s predictions were off by 7.05 units of the stock price.
* **Mean Squared Error (MSE)**: MSE is the average of the squared differences between predicted and actual values. It gives more weight to larger errors. For the training data, the MSE was 74.75.
* **Root Mean Squared Error (RMSE)**: RMSE is the square root of MSE and provides an error measure in the same units as the target variable. For the training data, the RMSE was 8.65.

These metrics showed that the model was performing well on the training data, but it was important to ensure that it also performed well on unseen data.

**3. Cross-Validation for Model Robustness**

To make the model more robust and avoid overfitting (where the model performs well on training data but poorly on new data), I applied a technique called cross-validation. Specifically, I used K-Fold Cross-Validation, where the data is split into multiple folds (in this case, 5). The model is trained on some folds and tested on the others, rotating through all the folds. This method provides a more accurate estimate of the model’s performance.

* **Cross-Validation R² Scores**: The average R² score across the folds was 0.93, with a standard deviation of 0.01, indicating that the model was consistently performing well across different subsets of the data.
* **Cross-Validation MAE, MSE, RMSE**: The MAE was 7.05, MSE was 74.75, and RMSE was 8.65, similar to the metrics from the training data, reinforcing that the model was not overfitting.

**4. Evaluating the Model on Test Data**

Next, I tested the model on the testing data, which was not used during training. This step was crucial to see how well the model generalizes to new data:

* **R² Score on Test Data**: The R² score on the test data was 0.95, higher than on the training data, indicating that the model was making very accurate predictions on unseen data.
* **MAE, MSE, RMSE on Test Data**: The MAE was 5.54, MSE was 47.31, and RMSE was 6.88. These lower error values compared to the training data showed that the model was not only accurate but also reliable.

**5. Analyzing Feature Importance**

To understand which features were most influential in predicting the stock prices, I analyzed the coefficients of the linear regression model. Features with larger absolute coefficients were considered more important. This analysis provided insights into which factors were driving the stock price predictions, helping in making more informed decisions based on the model.

**6. Addressing Challenges**

During the model-building process, I encountered some challenges, such as multicollinearity, where certain features were highly correlated, leading to redundancy. To address this, I explored regularization techniques like Ridge and Lasso regression, which help by adding penalties to the model and reducing the impact of less important features. Additionally, I applied Principal Component Analysis (PCA) to reduce the dimensionality of the data, focusing on the most significant features.

**7. Final Model Selection**

After testing different models and techniques, I found that incorporating PCA with the Linear Regression model gave the best results. The model was able to achieve a test R² score of 0.95 with relatively low errors (MAE: 5.54, RMSE: 6.88), making it a robust and reliable choice for predicting Amazon stock prices.

**Feature Importance Analysis**

To determine the importance of different features, I analyzed the coefficients of the Linear Regression model. The analysis showed that several features had a significant impact on the prediction of Amazon's stock price. However, due to the presence of multicollinearity, interpreting these coefficients was challenging.

**Challenges Faced**

1. Multicollinearity : One of the major challenges was dealing with multicollinearity among the features. This issue led to inflated standard errors and unreliable coefficient estimates, making it difficult to assess the true impact of each feature on the stock price.

2. Overfitting : The initial models, especially the Decision Tree model, exhibited overfitting. While the model performed exceptionally well on the training data, it did not generalize as well to the test data.

3. Feature Selection : Selecting the right set of features was crucial. I found that including too many features led to multicollinearity and overfitting, while reducing the feature set helped improve model performance but required careful consideration.

**Potential Improvements**

1. Regularization : Further fine-tuning of regularization parameters could help mitigate multicollinearity and improve model generalization.

2. Ensemble Techniques : Implementing ensemble techniques, such as Random Forest or Gradient Boosting, could enhance model performance by reducing overfitting and capturing complex relationships in the data.

3. Feature Engineering : Addsitional feature engineering, such as creating interaction terms or polynomial features, could potentially improve the model's ability to capture non-linear relationships.

**Conclusion**

This project provided valuable insights into predicting stock prices using Linear Regression. While the model performed well overall, there were challenges related to multicollinearity and overfitting that required careful handling. By applying techniques such as PCA and regularization, I was able to improve model performance. For future work, exploring advanced modeling techniques and further feature engineering could lead to even better results.

Note : Please refer to the model evaluation file, "***HUBBLEMIND\_model\_eval\_metrics.xlsx***," for detailed metrics and analysis.

THANK YOU