**Cs 210 Project Phase 3 – Ege Tan – 30977**

Description:

In this step of the project, I employed two machine learning models: K-Nearest Neighbors (KNN) and Random Forest. My goal was to predict Tesla's stock price using a combination of historical stock prices and Brent crude oil prices. KNN, a simple yet effective algorithm, operates by finding the closest data points (neighbors) in the training set to make predictions about new data points. Its simplicity and interpretability make it a good starting point for regression tasks. On the other hand, the Random Forest model, a more sophisticated ensemble learning method, builds multiple decision trees and combines their predictions to improve accuracy and robustness. This model is particularly effective in handling large datasets and complex relationships within the data. By using both KNN and Random Forest, I aimed to leverage the strengths of each model to achieve a comprehensive analysis and reliable predictions.For regression tasks, I evaluated performance using metrics that measure the difference between the predicted values and the actual values. Common regression metrics include Mean Squared Error (MSE), which is the average of the squared differences between the predicted and actual values, giving more weight to larger errors due to the squaring; Root Mean Squared Error (RMSE), the square root of the MSE, which brings the error metric back to the same units as the target variable, making it more interpretable. Using RMSE or MSE is more appropriate for regression tasks because they provide a quantitative measure of how close the predicted values are to the actual values. Here’s why I used RMSE instead of accuracy score for regression; Accuracy Score measures the fraction of correct predictions in classification tasks. It is not meaningful for continuous outcomes, as there is no concept of "correct" or "incorrect" in regression; instead, I measure how close the predictions are to the actual values. RMSE, MSE provide a measure of the magnitude of prediction errors, which is crucial in regression tasks where the goal is to minimize these errors. The subsequent evaluation and comparison of these models allowed me to determine which algorithm performed better in predicting Tesla's stock price, providing valuable insights into the underlying patterns and trends in the data.

First ML model (K-Nearest Neighbors):

While creating this model I applied this algorithm ;

**1)Load the data:** I began by loading the historical financial data and Brent crude oil prices.

**2)Preprocess the data:** I ensured all relevant features were selected and handled any missing values appropriately.

**3)Initialize the value of k:** I selected an initial value for k, the number of nearest neighbors to consider, and determined the optimal value through cross-validation.

**4)Calculate the distance:** For each test data point, I calculated the distance between the test data and each row of the training data using an appropriate distance metric.

**5)Sort the distances:** I sorted the calculated distances in ascending order based on the distance values.

**6)Select top k neighbors:** I selected the top k rows from the sorted array, representing the nearest neighbors.

**7)Predict the value:** For regression, I took the average of the target values (Tesla's stock prices) of these k nearest neighbors to make the prediction.

**8)Evaluate the model:** I assessed the model's performance using regression metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE).

A line graph with red dots

Description automatically generatedSince I have allready done the Load the data , preprocess the data steps in the previous part of the project , I continued from the third step which is Initialise the value of k. In my first try I initialized the k value to 1 . As a result of this, I got the RMSE,MSE values and the scatter plot that shows the relation between predicted Tesla\_Prices and actual ones;  
RMSE: 113.76 / MSE: 12941.80

A graph with blue dots

Description automatically generated

Then for finding best k value I implemented the interaction graph that shows the change in the relation between predicted Tesla\_Prices and actual ones, dependent to the k values;

A line graph with orange dots

Description automatically generatedA graph with red dots

Description automatically generatedI also implemented k value graph that shows the best k value for my first ml model prediction.The one that have less RMSE value is the k value that we are searching for.

As we see from the graph above the best k value for my KNN model is 8. So in the next step I visualised how my prediction model looks like when k value is 8.

RMSE: 89.78 / MSE: 8061.72

Using these optimal parameters, the model achieved a Mean Squared Error (MSE) of 8048.72 and a Root Mean Squared Error (RMSE) of 89.71 on the test dataset. These metrics indicate the average squared difference and the average error between the predicted and actual Tesla stock prices, respectively. The RMSE, in particular, provides an interpretable measure of error in the same units as the target variable, highlighting the model's accuracy.In contrast, when using a random k value without hyperparameter tuning, we observed a significant decline in model performance. For instance, with a k value of 1, the model's MSE increased to 14003.92, and the RMSE rose to 118.34. This comparison underscores the importance of choosing the optimal k value and other hyperparameters to minimize prediction errors and improve model accuracy.The scatter plot of actual vs. predicted values further illustrates this difference. The optimal model's predictions closely align with the actual values along the diagonal line, indicating better accuracy. Conversely, the random k model shows more significant deviations from the diagonal, reflecting higher prediction errors.In summary, selecting the optimal k value and other parameters significantly reduces prediction errors, providing more accurate and reliable results for Tesla stock price predictions based on Brent crude oil prices.

Second ML model (Random Forest):

While creating this model, I applied the following algorithm for the Random Forest model:

**1)Load the data:** I began by loading the historical financial data and Brent crude oil prices.

**2)Preprocess the data:** I ensured all relevant features were selected and handled any missing values appropriately.

**3)Split the data:** I divided the dataset into training and testing sets to evaluate the model's performance on unseen data.

**4)Initialize the Random Forest model:** I experimented with different values for the number of decision trees, known as n\_estimators, to find the optimal value.

**5)Train the model:** Using the training set, I trained the Random Forest model, starting with the optimal value of n\_estimators and optimal max\_depth.

**6)Make predictions:** I used the trained model to predict Tesla's stock prices on the test set.

**7)Evaluate the model:** I assessed the model's performance using regression metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE).

**8)Visualize the results:** I created scatter plots to compare actual versus predicted Tesla prices and line plots to show the actual and predicted prices over time.

Since I have allready done the Load the data , preprocess the data and split the data steps in the previous part of the project , I continued from the fourth step which is Initialise the random forest model. In my first try I initialized the model with 50 trees(n\_estimator) and with 10 max\_depth . As a result of this, I got the RMSE,MSE values and the scatter plot that shows the relation between predicted Tesla\_Prices and actual ones;

A graph of purple dots

Description automatically generated

RMSE: 94.06 / MSE: 8848.81

In this step, the process of applying and testing hyperparameters for the Random Forest model was carried out to ensure optimal performance. The key hyperparameters tuned include n\_estimators and max\_depth, which significantly influence the model's performance and complexity.

**n\_estimators:** This hyperparameter controls the number of trees in the forest. By iterating through different values, the optimal number was determined based on the balance between model performance and computational efficiency. Higher values generally improve performance up to a certain point but also increase the computational cost.

**max\_depth:** This hyperparameter limits the maximum depth of each tree. By tuning this parameter, the model avoids overfitting by not allowing trees to grow too deep, which would capture noise in the training data. The optimal depth ensures the model generalizes well to unseen data.

In the end I created a graph that shows the change in RMSE value with change in max\_Depth and n\_estimators.For better understanding I visualised them with color since , the process required to compare 3 parameter (Max\_depth,n\_estimator,RMSE value).

A chart with numbers and a number of different levels

Description automatically generated with medium confidence

As we see from the graph above with max\_depth = 5 and n\_estimators = 750 the minimum RMSE value occurred ,which is 83.88, for my ML model with random forest.And lastly I visualised the best performance of my Ml model by showing the predicted and actual values grapha as bellow.

A graph of green dots

Description automatically generated

RMSE: 86.63 / MSE: 7504.92

**Comparison of ML models:**

The ensemble approach of the Random Forest model, which combines multiple decision trees to enhance prediction accuracy, proved more effective in capturing the complex relationships within the dataset. This led to more accurate and reliable predictions compared to the KNN model. The detailed scatter plots and performance metrics underscored the Random Forest model's ability to minimize prediction errors and its robustness in handling the intricacies of the data. This comparison highlights the critical importance of selecting and meticulously tuning the appropriate machine learning algorithms for predicting Tesla's stock prices based on Brent crude oil prices, with the Random Forest model exemplifying its better performance and reliability.

Overall, both models demonstrated substantial improvement with hyperparameter tuning. However, the Random Forest model outperformed the KNN model, with a lower RMSE (86.63 vs. 89.78) and MSE (7504.92 vs. 8061.72). The ensemble approach of the Random Forest model proved more effective in capturing the complex relationships within the data, leading to more accurate and reliable predictions. This comparison highlights the significance of selecting and tuning the appropriate machine learning algorithms for predicting Tesla's stock prices based on Brent crude oil prices, with the Random Forest model showcasing its robustness and better performance.