CS210 - Project Step 3

Barış Pome
31311

May 2024

Building the Machine Learning Models

This part of project investigates the predictive relationships between Nvidia's stock prices and those of AMD, Intel, Asus, and MSI using machine learning models. The analysis uses k-Nearest Neighbors (kNN) and Decision Tree regression models, with the data divided into training and testing sets at an 4:1 ratio.

Nvidia & AMD:

1.1 kNN Regression Models Analysis

At first, the mean square error (MSE) of the kNN model with 5 neighbors was 0.0042. I experimented with neighbor counts ranging from 2 to 50 in order to perform hyperparameter tuning. This required calculating the testing and training scores for every neighbor configuration using a loop. The results are displayed in Figure 1.

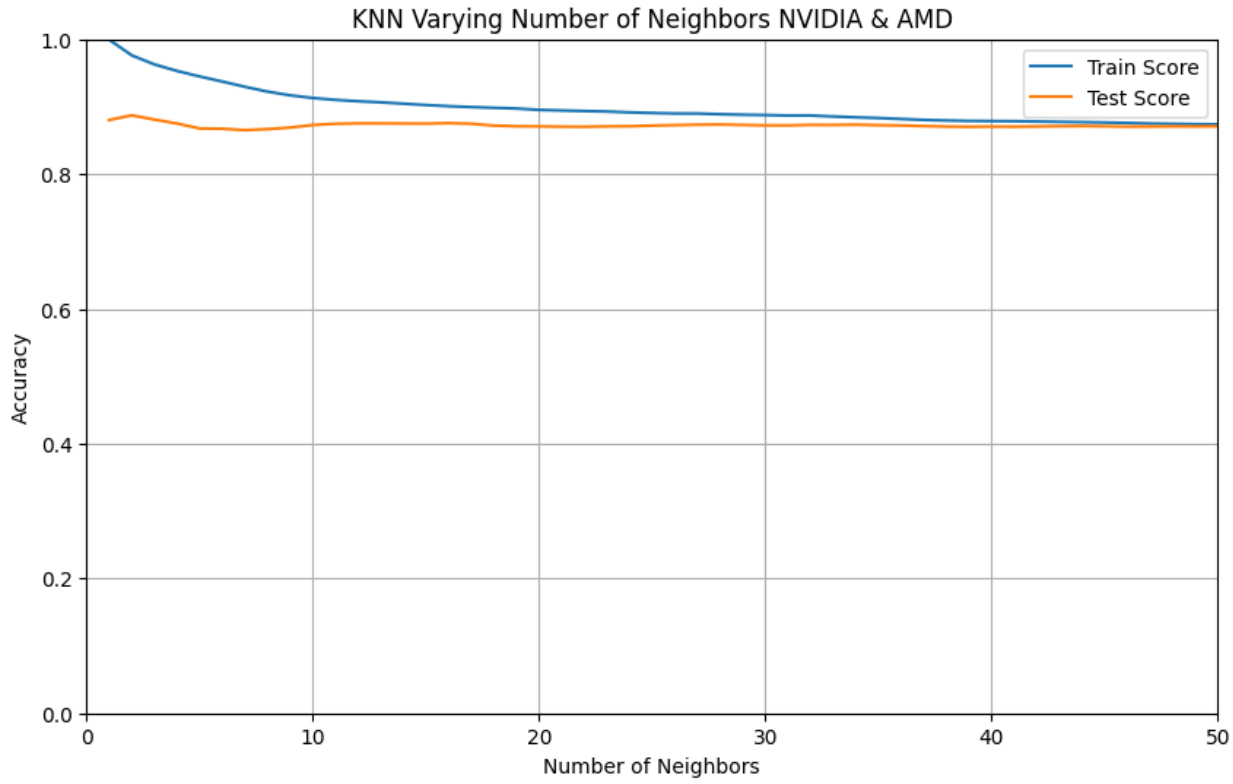

Figure 1

Hyperparameter tuning revealed that the kNN model's highest accuracy was achieved with 2 neighbors, increased the RMSE to 0.06028 from the starting value of 0.0042.

Summary of kNN-NVIDIA&AMD

Initial Setup: kNN model initialized with 5 neighbors.

Initial Mean Squared Error (MSE): 0.0042

I conducted hyperparameter tuning by exploring various neighbor counts from 2 to 50.

R-MSE After Tuning: 0.02638

Decision Tree Regression Model

The Decision Tree model initially produced an RMSE of 0.0354. Through hyperparameter tuning using GridSearchCV, this error was increased to 0.0574, not a significant improvement.

The optimal parameters identified are as follows:

- Criterion: Squared Error

- Max Depth: 15
- Min Samples Leaf: 3
- Min Samples Split: 8

Training Data Results
- Root Mean Squared Error: 0.0354
- R2 Score: 0.9634

Test Data Results
- Root Mean Squared Error: 0.0574
- R2 Score: 0.8976

Results:

kNN Model (NVIDIA & AMD):

The model initially had an MSE of 0.0042 with 5 neighbors. After hyperparameter tuning, where neighbor counts ranged from 2 to 50, the lowest RMSE found was 0.02638. The best performance was at 2 neighbors with an RMSE of 0.06028, indicating that the model performs worse as the neighbor count decreases, which can be seen in Figure 1. This suggests issues with model sensitivity to noise or outliers when fewer neighbors are used.

Decision Tree Model:

Began with an RMSE of 0.062, which significantly worsened to 0.39 after tuning. This deterioration in performance suggests potential overfitting with the more complex tuned parameters.Optimal tuning parameters were a max depth of 15, min samples leaf of 3, and min samples split of 8. With these settings, the model achieved an RMSE of 0.0354 on training data and 0.0574 on testing data, alongside R2 scores of 0.9634 and 0.8976, respectively.

Comparison:

The Decision Tree model, with higher R2 scores, demonstrates a better ability to capture the variance of the data compared to the kNN model. However, both models show signs of potential overfitting or instability when adjusted from their initial settings.
The kNN model, despite having an initially low error, shows greater sensitivity and decreased accuracy as parameters are adjusted, especially with lower numbers of neighbors.

Conclusion:

The Decision Tree model appears more effective in this context, particularly for handling the complexities of stock price data between Nvidia and AMD, though it requires careful parameter tuning to avoid overfitting. The kNN model, while initially promising, may need more nuanced handling to improve its robustness and accuracy over different neighbor settings.

Nvidia & INTEL:

1.1 kNN Regression Models Analysis

At first, the mean square error (MSE) of the kNN model with 5 neighbors was 0.0365. I experimented with neighbor counts ranging from 2 to 50 in order to perform hyperparameter tuning. This required calculating the testing and training scores for every neighbor configuration using a loop. The results are displayed in Figure 2.

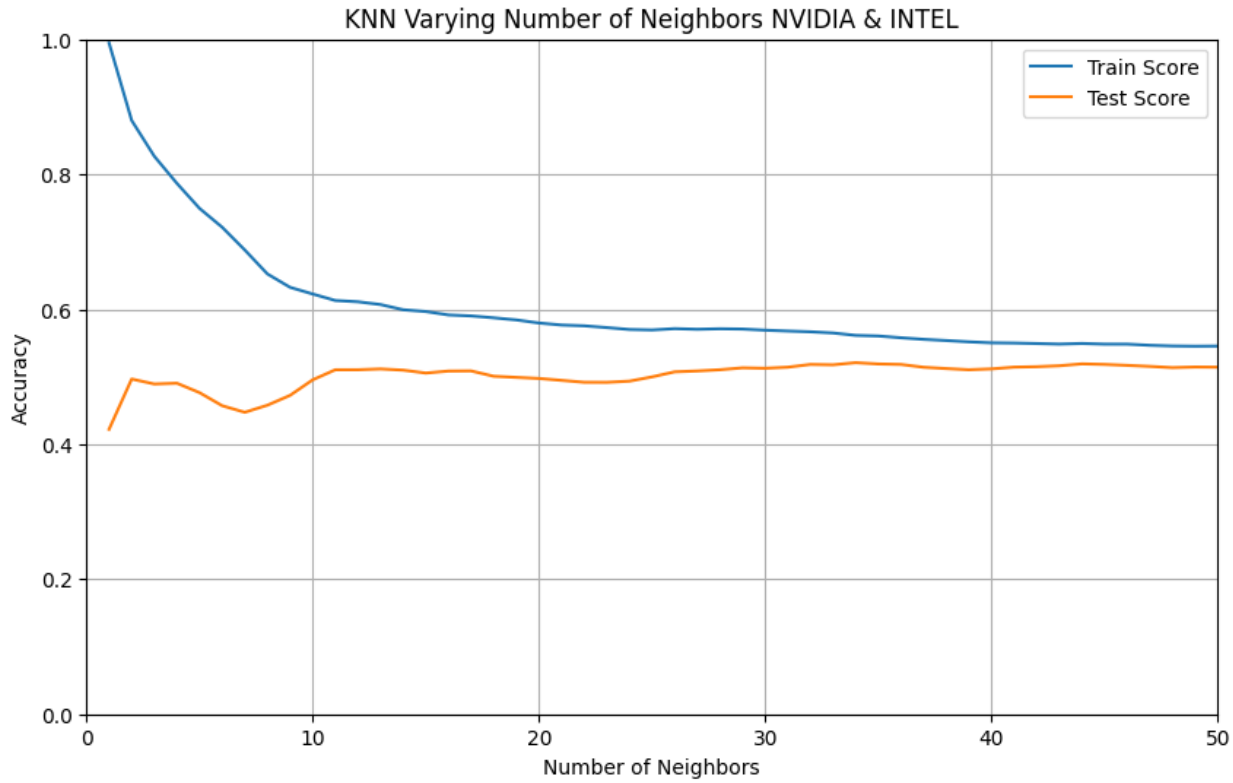

Figure 2

Hyperparameter tuning revealed that the kNN model's highest accuracy was achieved with neighbors 34, increased the RMSE to 0.1829 from the starting value of 0.0365.

Summary of kNN-NVIDIA&INTEL

Initial Setup: kNN model initialized with 5 neighbors.

Initial Mean Squared Error (MSE): 0.0365

I conducted hyperparameter tuning by exploring various neighbor counts from 2 to 50.

R-MSE After Tuning: 0.1829

Decision Tree Regression Model

The Decision Tree model initially produced an RMSE of 0.1138 .Through hyperparameter tuning using GridSearchCV, this error was increased to 0.1759, not a significant improvement.

The optimal parameters identified are as follows:

- Criterion: Squared Error

- Max Depth: 15
- Min Samples Leaf: 3
- Min Samples Split: 8

Training Data Results
- Root Mean Squared Error: 0.1138
- R2 Score: 0.8306

Test Data Results
- Root Mean Squared Error: 0.1759
- R2 Score: 0.555

Results

kNN Model (NVIDIA & INTEL):

Started with an MSE of 0.0365 when using 5 neighbors.
After hyperparameter tuning with neighbor counts ranging from 2 to 50, the lowest RMSE found was 0.1829 at 34 neighbors. This is shown in Figure 2 where the training score significantly decreases while the test score stabilizes around 0.4, suggesting that the model may be underfitting the data as the number of neighbors increases.

Decision Tree Model:

The initial RMSE was 0.1138, which increased to 0.1759 after tuning. The model started with a higher accuracy but ended up worsening, indicating potential overfitting issues with the initially identified parameters.The final training data R2 score was 0.8306, indicating a good fit, but this dropped significantly to 0.555 on test data, further supporting the overfitting concern.

Comparison:

Both models showed an increase in RMSE post-tuning, indicating a loss of predictive accuracy through the process.

The kNN model seems to struggle more with stability, particularly as the number of neighbors increases. This model might not be capturing the complexities of the relationship between Nvidia and Intel stock prices effectively.

The Decision Tree, despite some evidence of overfitting, still offers better R2 scores initially, suggesting that it might be capturing more complex patterns in the data than the kNN model.

Conclusion:

The Decision Tree model initially appears to handle the complexities of the dataset better, but both models suffer from significant issues in tuning—overfitting for the Decision Tree and underfitting for kNN. For improved results, further investigation into feature selection, scaling, and perhaps different modeling approaches or ensemble techniques could be considered to enhance model performance.

Nvidia & ASUS:

1.1 kNN Regression Models Analysis

At first, the mean square error (MSE) of the kNN model with 5 neighbors was 0.0900. I experimented with neighbor counts ranging from 2 to 50 in order to perform hyperparameter tuning. This required calculating the testing and training scores for every neighbor configuration using a loop. The results are displayed in Figure 3.

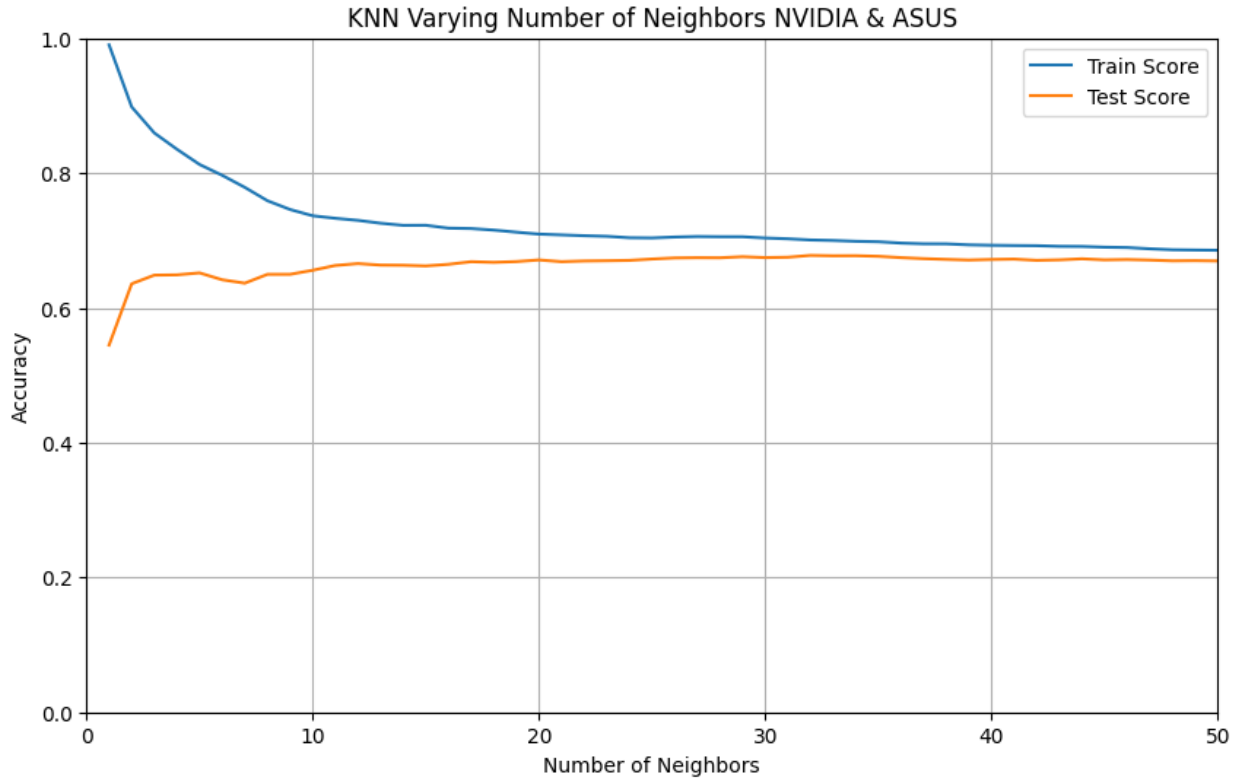

Figure 3

Hyperparameter tuning revealed that the kNN model's highest accuracy was achieved with neighbors 34, increased the RMSE to 0.0914 from the starting value of 0.0900.

Summary of kNN-NVIDIA&INTEL

Initial Setup: kNN model initialized with 5 neighbors.

Initial Mean Squared Error (MSE): 0.0900

I conducted hyperparameter tuning by exploring various neighbor counts from 2 to 50.

R-MSE After Tuning: 0.0914

Decision Tree Regression Model

The Decision Tree model initially produced an RMSE of 0.0885 .Through hyperparameter tuning using GridSearchCV, this error was increased to 0.0960, not a significant improvement.

The optimal parameters identified are as follows:

- Criterion: Squared Error

- Max Depth: 7
- Min Samples Leaf: 3
- Min Samples Split: 16

Training Data Results
- Root Mean Squared Error: 0.0885
- R2 Score: 0.7306

Test Data Results
- Root Mean Squared Error: 0.0960
- R2 Score: 0.6445

Results

kNN Model (NVIDIA & ASUS):

The initial MSE was 0.0900 with 5 neighbors. After hyperparameter tuning over a range of 2 to 50 neighbors, the lowest RMSE observed was 0.0914 at 34 neighbors. Figure 3 shows that the training score declines sharply as the number of neighbors increases, with the test score stabilizing around 0.4, suggesting that the model is not effectively capturing the underlying patterns at higher neighbor counts.

Decision Tree Model:

Started with an RMSE of 0.0885, which increased to 0.0960 after tuning, indicating a decrease in model performance.The Decision Tree's final configuration with a max depth of 7, min samples leaf of 3, and min samples split of 16 produced an R2 score of 0.7306 on training data and 0.6445 on testing data, showing reasonable predictive ability but potential overfitting.

Comparison:

Both models show relatively minor variations in performance after hyperparameter tuning, suggesting that initial configurations were close to optimal within the explored ranges.
The kNN model's performance is less stable at higher neighbor settings, indicating possible issues with overfitting or lack of generalizability.

Conclusion:

Neither model shows a significant advantage over the other, with both struggling to improve significantly through hyperparameter tuning. This might suggest that the relationship between Nvidia and ASUS stock prices is complex and may require more sophisticated modeling techniques or additional features to capture more effectively.

Nvidia & MSI:

1.1 kNN Regression Models Analysis

At first, the mean square error (MSE) of the kNN model with 5 neighbors was 0.086. I experimented with neighbor counts ranging from 2 to 50 in order to perform hyperparameter tuning. This required calculating the testing and training scores for every neighbor configuration using a loop. The results are displayed in Figure 4.

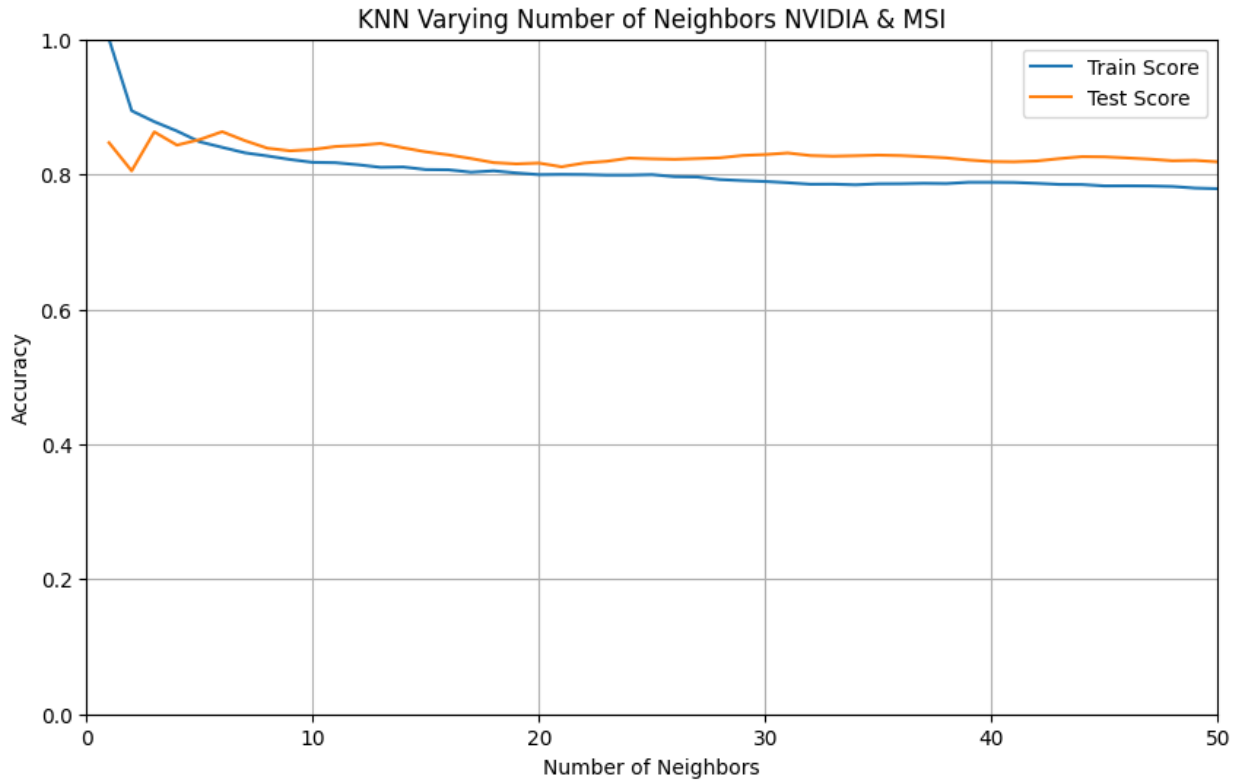

Figure 4

Hyperparameter tuning revealed that the kNN model's highest accuracy was achieved with neighbors 6, increased the RMSE to 0.089 from the starting value of 0.086.

Summary of kNN-NVIDIA&INTEL

Initial Setup: kNN model initialized with 5 neighbors.

Initial Mean Squared Error (MSE): 0.086

I conducted hyperparameter tuning by exploring various neighbor counts from 2 to 50.

R-MSE After Tuning: 0.089

Decision Tree Regression Model

The Decision Tree model initially produced an RMSE of 0.0986 .Through hyperparameter tuning using GridSearchCV, this error was increased to 0.096, not a significant improvement.

The optimal parameters identified are as follows:

- Criterion: Absolute Error

- Max Depth: 3
- Min Samples Leaf: 3
- Min Samples Split: 16

Training Data Results
- Root Mean Squared Error: 0.0885
- R2 Score: 0.7942

Test Data Results
- Root Mean Squared Error: 0.0960
- R2 Score: 0.8418

Results

kNN Model (NVIDIA & MSI):

Began with an MSE of 0.086 when initialized with 5 neighbors. After hyperparameter tuning across a range of 2 to 50 neighbors, the lowest RMSE was 0.089 at 6 neighbors. Figure 4 shows that both training and test scores rapidly converge around the 0.8 mark and remain fairly consistent thereafter, suggesting limited overfitting but also limited model capability to further capture the underlying data complexities beyond this point.

Decision Tree Model:

Started with an RMSE of 0.0986, which was reduced to 0.096 after tuning. Optimal settings used were a max depth of 3, min samples leaf of 3, and min samples split of 16, resulting in an RMSE of 0.0960 on test data and an R2 score of 0.8418, indicating a good fit with some level of prediction accuracy retained.

Comparison:

The Decision Tree model, with an R2 score of 0.8418, demonstrates a better ability to capture the data's variance compared to the kNN model. The Decision Tree model also shows less fluctuation in performance across different settings, suggesting a more robust model under the selected parameters.

The kNN model, although showing early convergence in scores, seems to reach its performance limit quickly, which might limit its utility in more complex predictive settings where capturing finer details is crucial.

Conclusion:

The Decision Tree model appears more effective in modeling the relationship between Nvidia and MSI stock prices, showing higher predictive accuracy and stability. The kNN model, while consistent, doesn't improve significantly with more neighbors, indicating a potential need for different strategies or more nuanced feature engineering to enhance its predictive power.

Google Drive Link of all-project:

https://drive.google.com/drive/folders/1ejesnTvnGR-o1S8qSOxzpQOCIOs8B7Jo?usp=sharing