****

**Machine Learning Project Documentation**

Real-Time Air Quality Prediction and Classification System

**Group – 1 Members**

1. **Mohammad Wasil Jalali**
2. **Bahir Saidi**
3. **Mohammad Aref Rezvan Panah**
4. **Habibullah Farahmand**

**Machine Learning Project Documentation**

**Model Refinement**

**1. Overview**

The Model Refinement phase is integral to the success of any machine learning project. This phase focuses on improving the initial model's performance by addressing its weaknesses and optimizing its parameters. The goal is to ensure that the model generalizes well to unseen data, minimizes overfitting, and improves accuracy, precision, and other key performance metrics. The refinement techniques implemented in this project aimed to enhance the model's predictive capabilities for real-time air quality prediction.

**2. Model Evaluation**

In the initial evaluation phase, the following models were tested: CNN, LSTM, TSMixer, and RandomForestClassifier for classification tasks. Despite achieving approximately 95% accuracy on the training dataset, the models performed poorly on the test dataset, indicating overfitting. Key metrics, including precision, recall, F1-score for classification, and MSE, RMSE, MAE for regression, were used to assess the models. Key visualizations, such as confusion matrices and scatter plots, were instrumental in identifying the overfitting issue.

**Areas for Improvement**:

* Overfitting due to high training accuracy and lower testing accuracy.
* Inconsistent performance across different models.
* The need for better generalization on unseen data.

**3. Refinement Techniques**

To improve model performance, several techniques were employed:

* **Hyperparameter Tuning**: Adjusting parameters such as the number of neurons in LSTM layers, the learning rate, and the number of trees in the RandomForestClassifier helped improve accuracy.
* **Model Re-Selection**: Given the performance challenges, the final model selection was refined to prioritize RandomForestClassifier for classification due to its strong generalization, and TSMixer for regression due to its slightly better performance over LSTM.
* **Ensemble Methods**: Ensemble methods were considered to combine the strengths of individual models and mitigate their weaknesses, though a final ensemble model was not implemented due to the computational cost and complexity.

**4. Hyperparameter Tuning**

Hyperparameter tuning was performed using GridSearchCV and RandomizedSearchCV, focusing on critical parameters such as:

* **RandomForestClassifier**: Number of estimators (trees) and maximum depth of trees.
* **LSTM and TSMixer**: Number of layers, neurons per layer, and learning rate.
* Insights gained from tuning showed that while RandomForestClassifier reached a stable point with a higher number of estimators, LSTM models required dropout layers and early stopping to prevent overfitting. TSMixer benefited from slight adjustments in layer sizes, which improved its performance slightly over LSTM.

**5. Cross-Validation**

During the model refinement phase, K-fold cross-validation was implemented to assess the models’ generalization capability across different subsets of the data. Initially, a basic validation strategy was employed, but after observing signs of overfitting, a more robust 10-fold cross-validation was used. This approach improved the model's ability to generalize by testing on multiple data splits, which helped ensure that the final model was not biased by any particular training subset.

**6. Feature Selection**

Feature selection was revisited in this phase:

* **Dimensionality Reduction**: While PCA was initially tested, it was discarded due to the loss of 9% variance. Instead, features like NH3 and NO were dropped because of their high correlation with NO2 and O3, which sufficiently represented air quality indicators.
* **Mutual Information and Correlation Analysis**: These techniques were used to refine the feature set further, ensuring that only the most relevant features were used, thus improving the model’s efficiency and predictive power.

**Test Submission**

**1. Overview**

The Test Submission phase is where the final model, after refinement, is applied to a separate test dataset to evaluate its performance in a real-world scenario. This phase prepares the model for deployment or further evaluation by ensuring it performs well on data it has never seen before. This step is crucial to validate the model’s generalization capability.

**2. Data Preparation for Testing**

The test dataset, which was separate from the training and validation data, was preprocessed in a similar manner:

* Missing values were imputed using KNN imputation, following the same strategy applied during data preparation.
* The test dataset was transformed using the same feature engineering techniques, ensuring that features like temporal and geospatial variables matched the format used during training.
* Any inconsistencies or duplicates were handled, ensuring the model received a clean and well-prepared dataset.

**3. Model Application**

Once the data was prepared, the trained model was applied to the test dataset. The application of the model involved feeding the test data through the final RandomForestClassifier for classification tasks and the TSMixer for regression predictions.

**4. Test Metrics**

The performance on the test dataset was evaluated using the same metrics as during the training phase:

* **Classification**: Precision, recall, F1-score, and confusion matrix were computed to assess the classification quality.
* **Regression**: MSE, RMSE, and MAE were used to evaluate the regression performance.

The results showed that the model performed significantly better on the test data than the initial models, achieving a balance between training accuracy and test accuracy, thus demonstrating improved generalization.

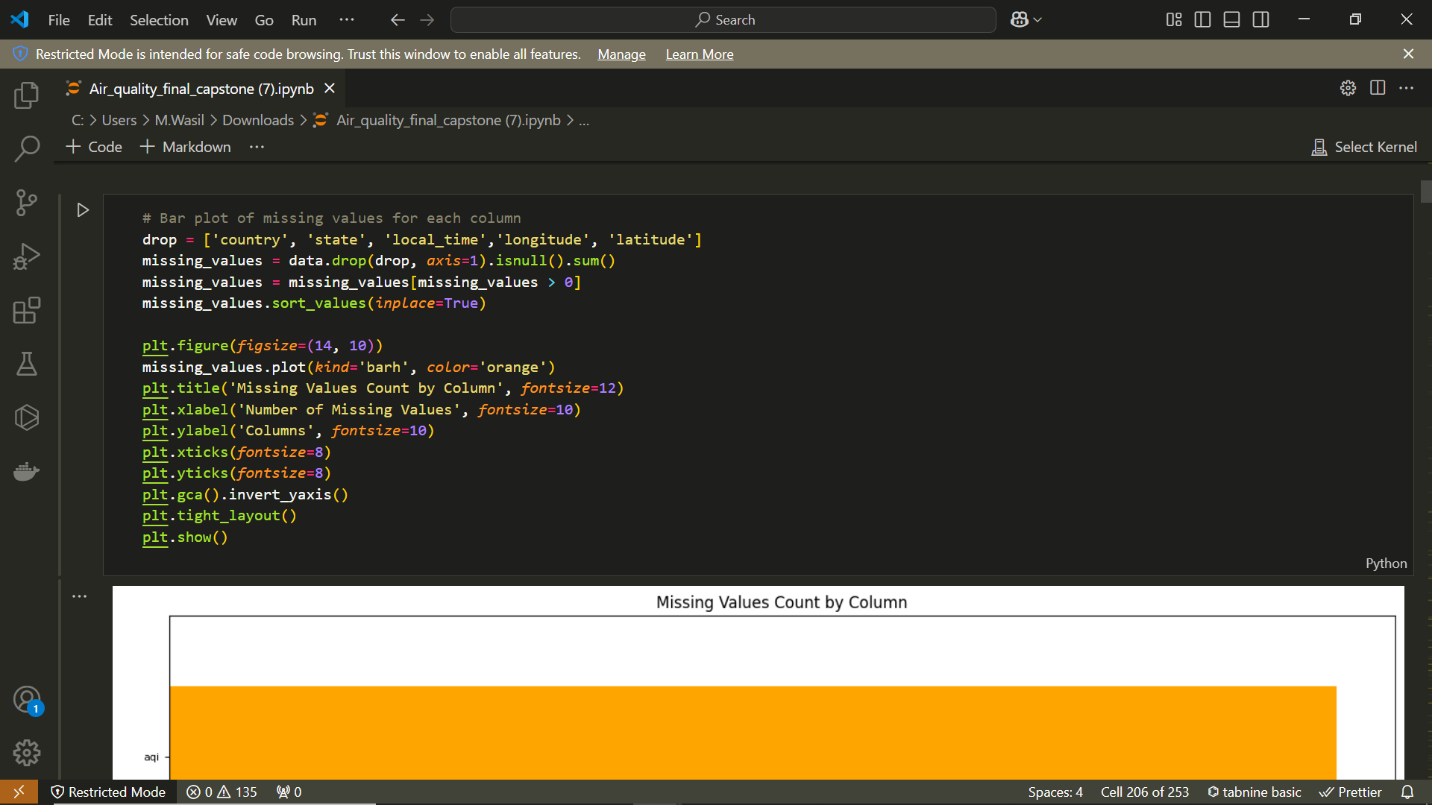
**5. Model Deployment**

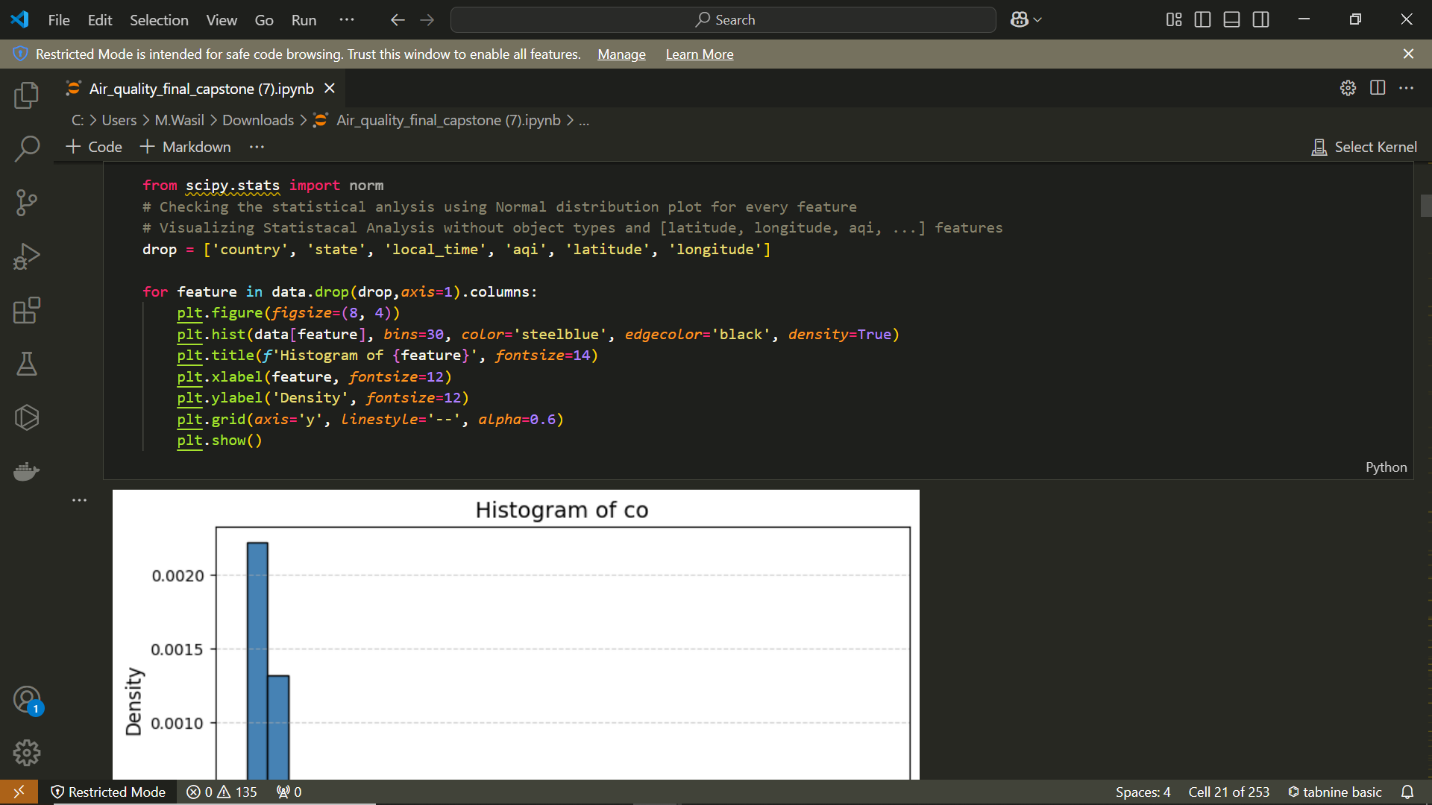
Once the final model was evaluated and its performance was satisfactory, steps were taken to deploy it into a real-world setting. This involved:

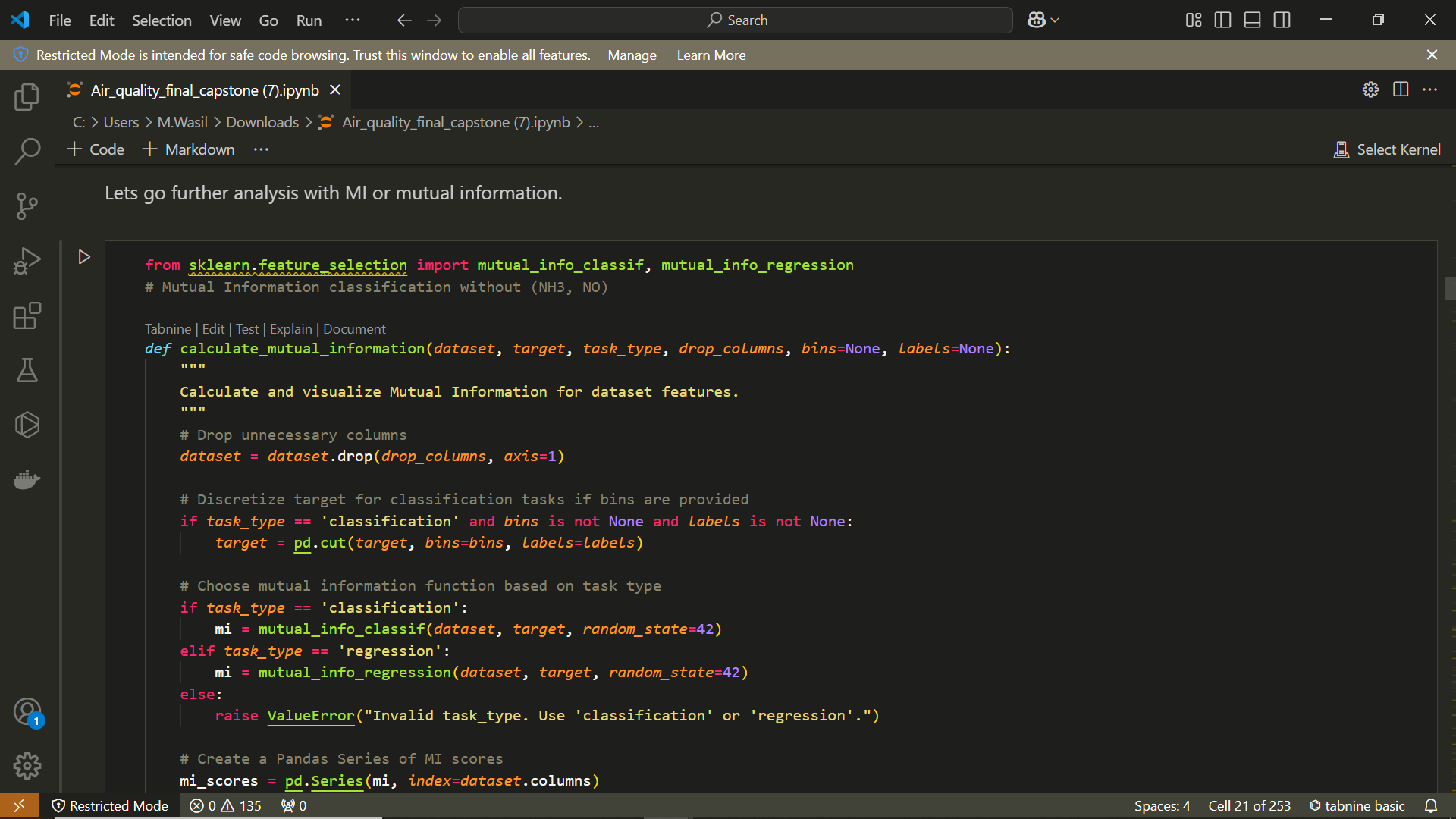
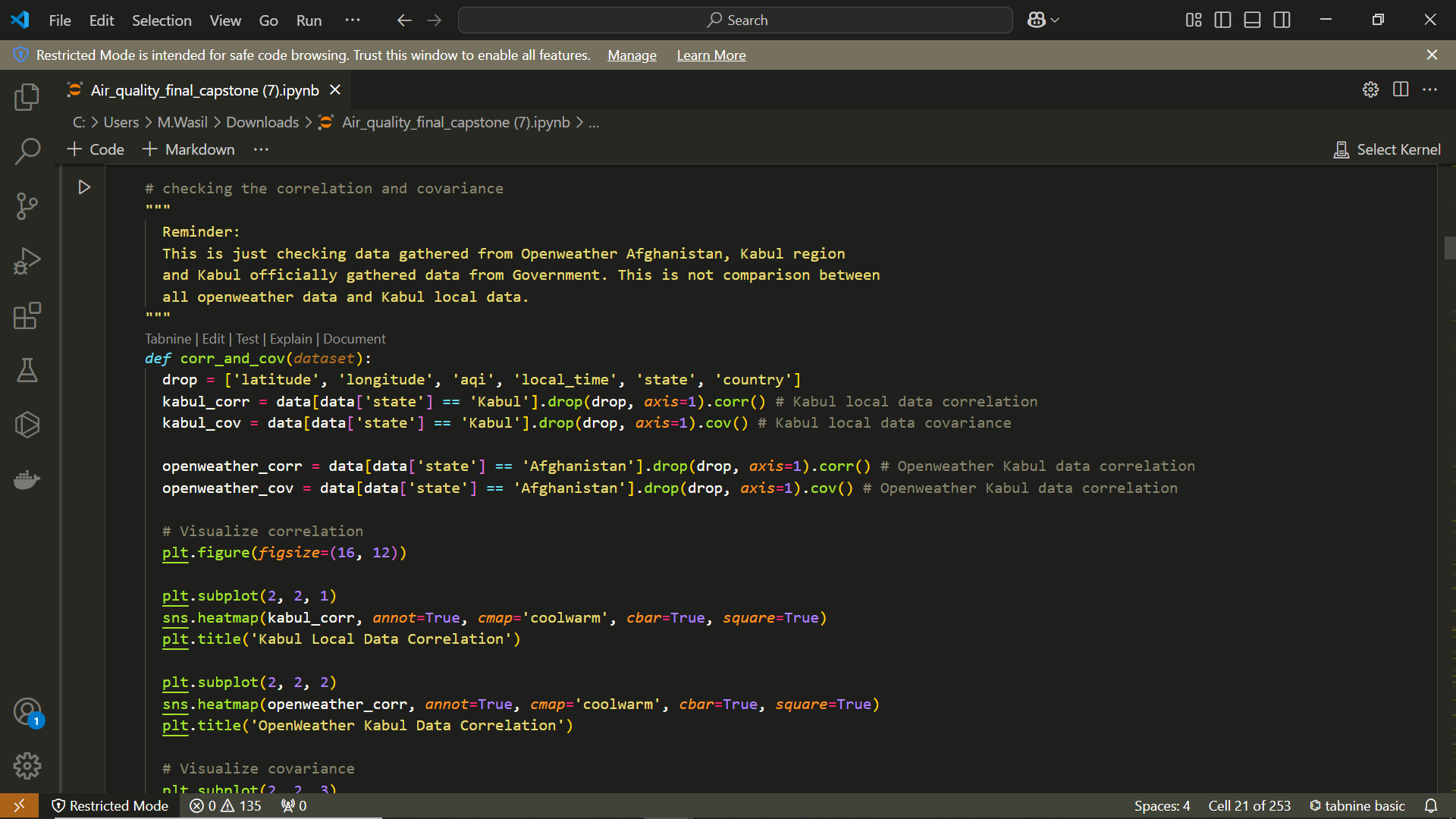
* Integrating the model with a real-time air quality monitoring system, using the OpenWeather API and NEPA dataset for continuous updates.
* Exposing the model through an API to allow external systems to query the predictions.
* Ensuring that the system could handle new data in real-time and make predictions based on the most recent information available.

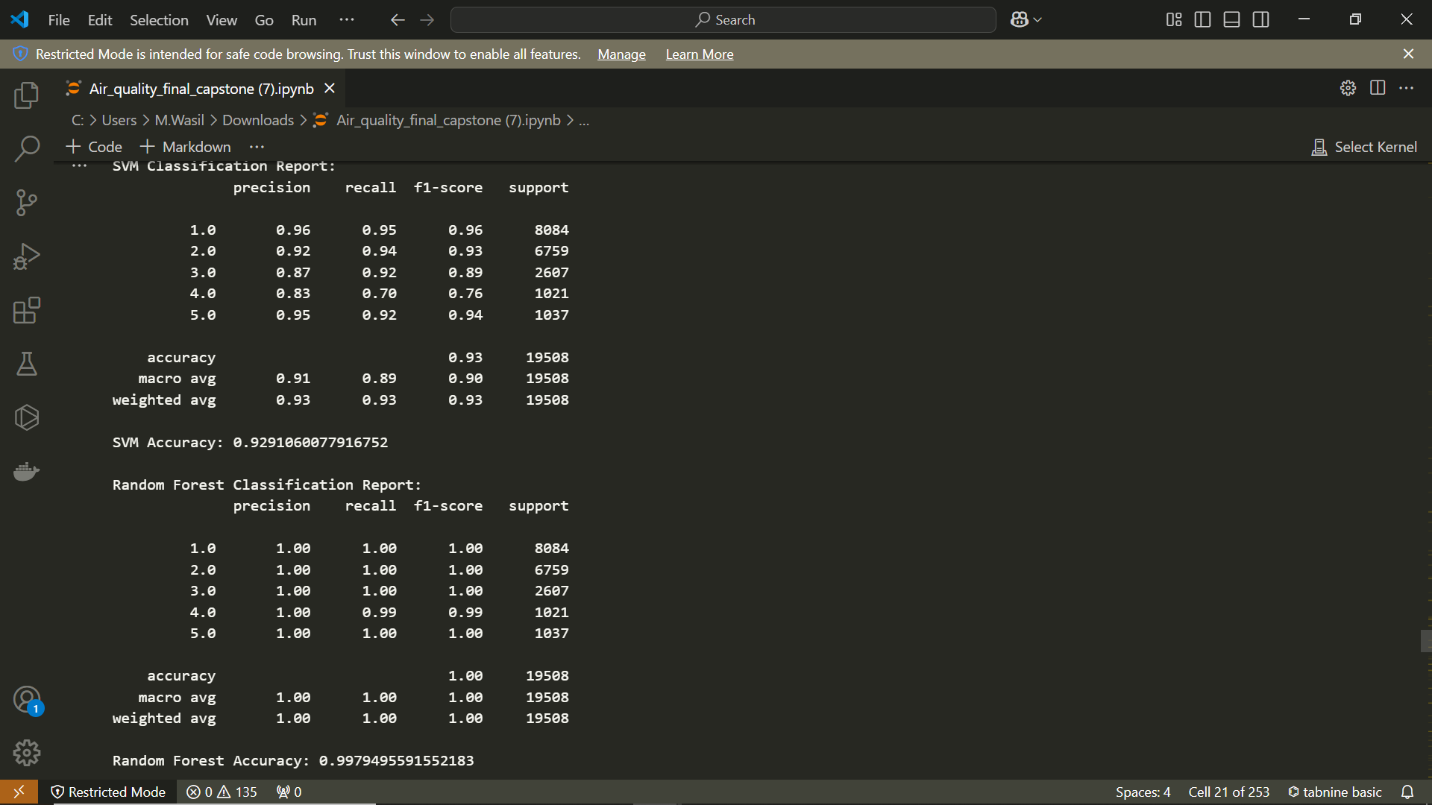
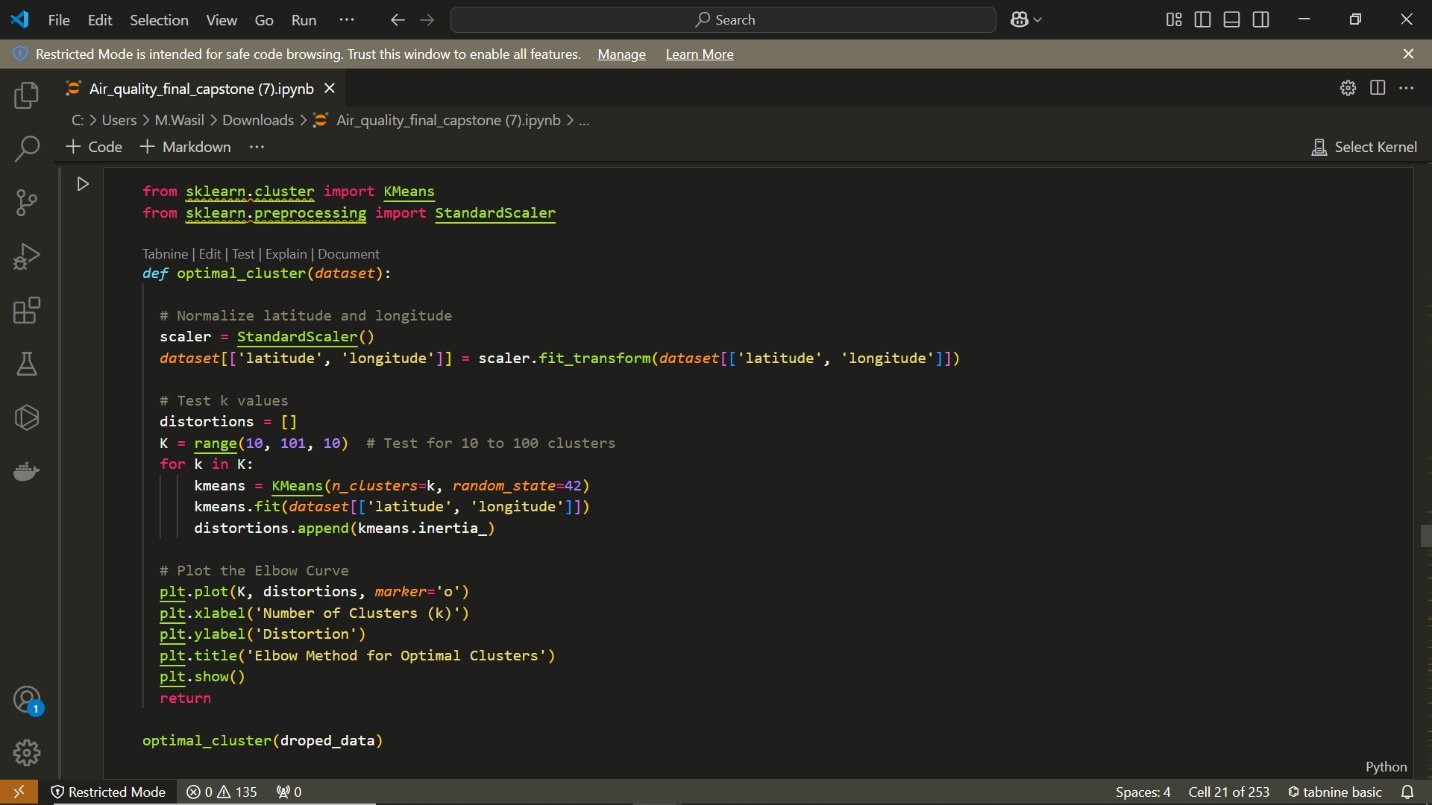
**6. Code Implementation**

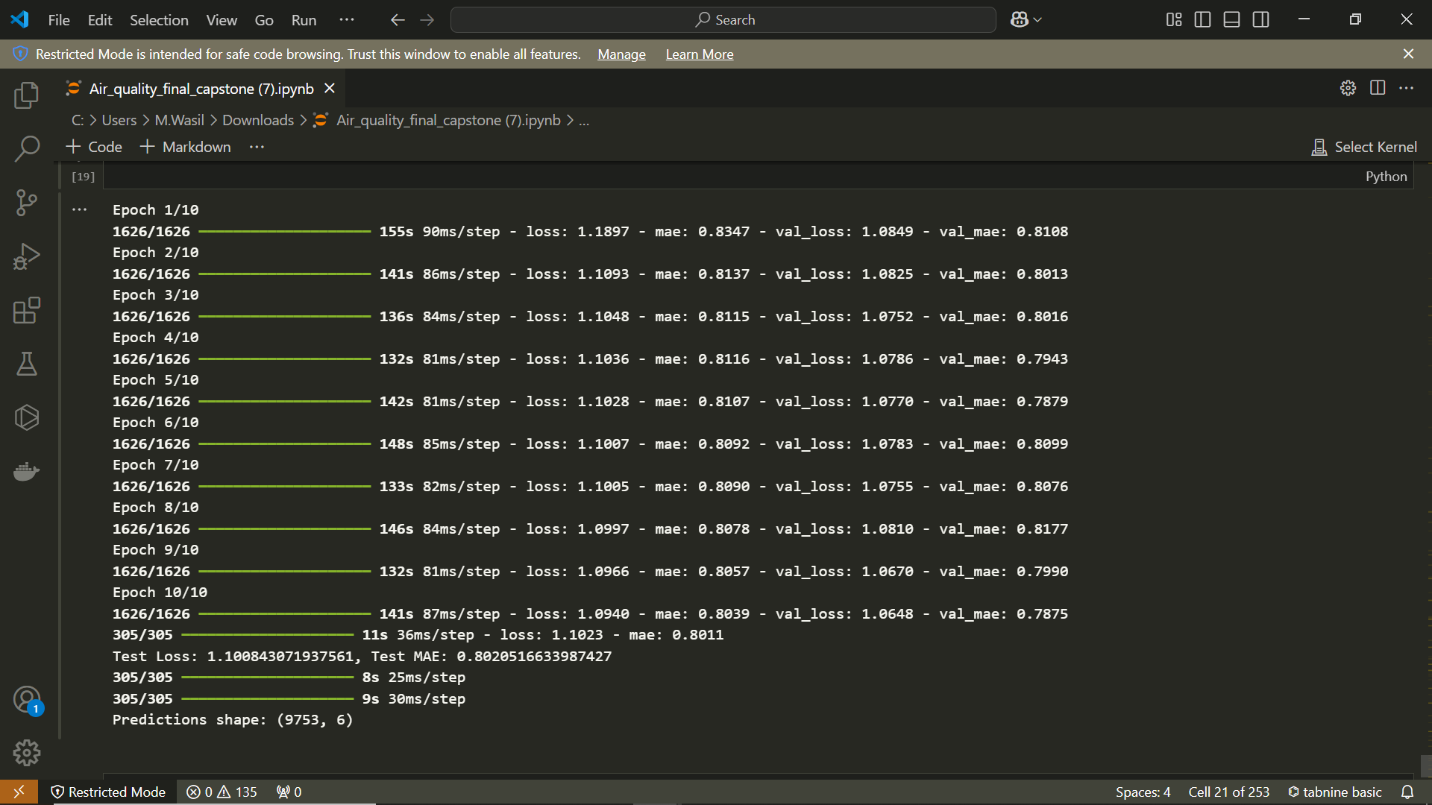
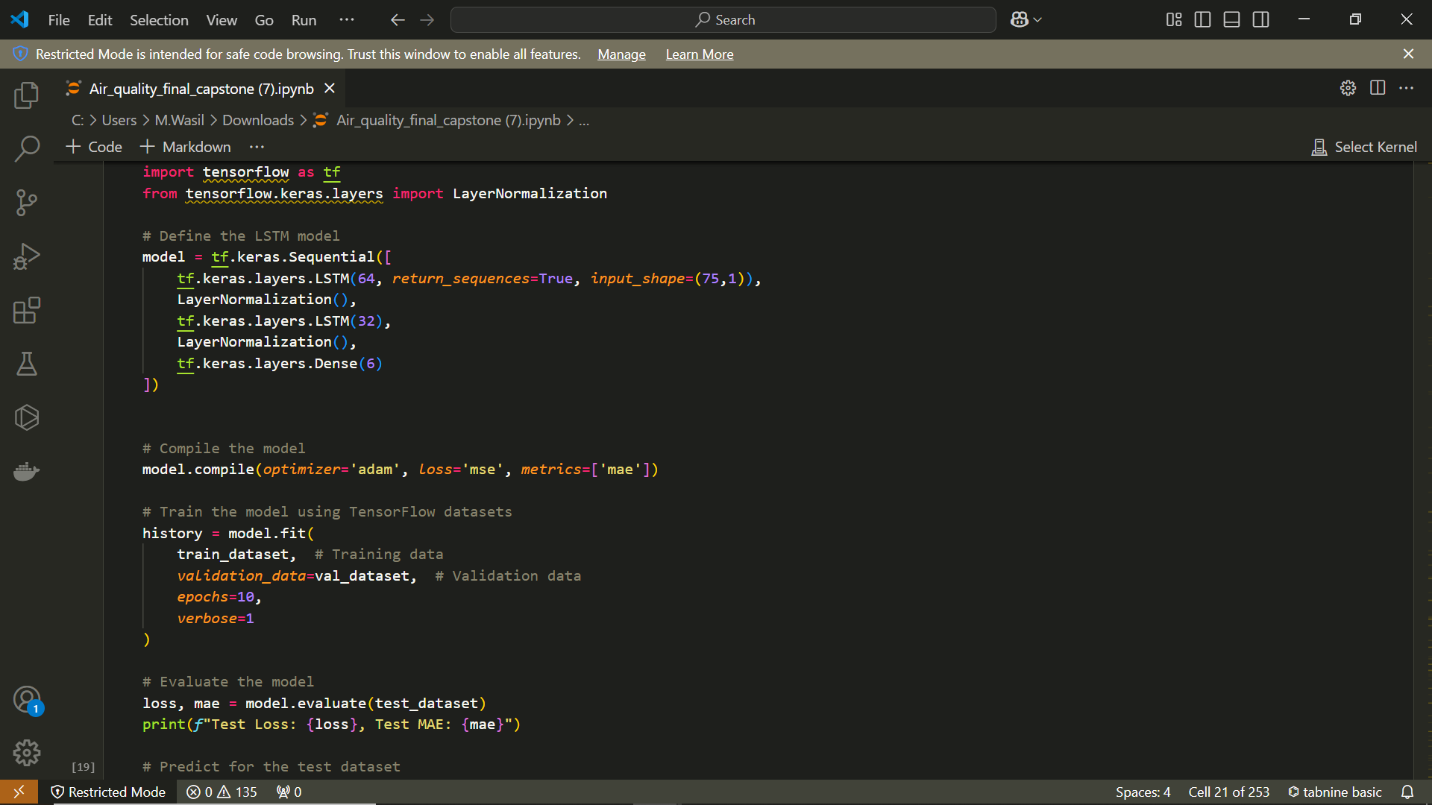
Here are some key code snippets:

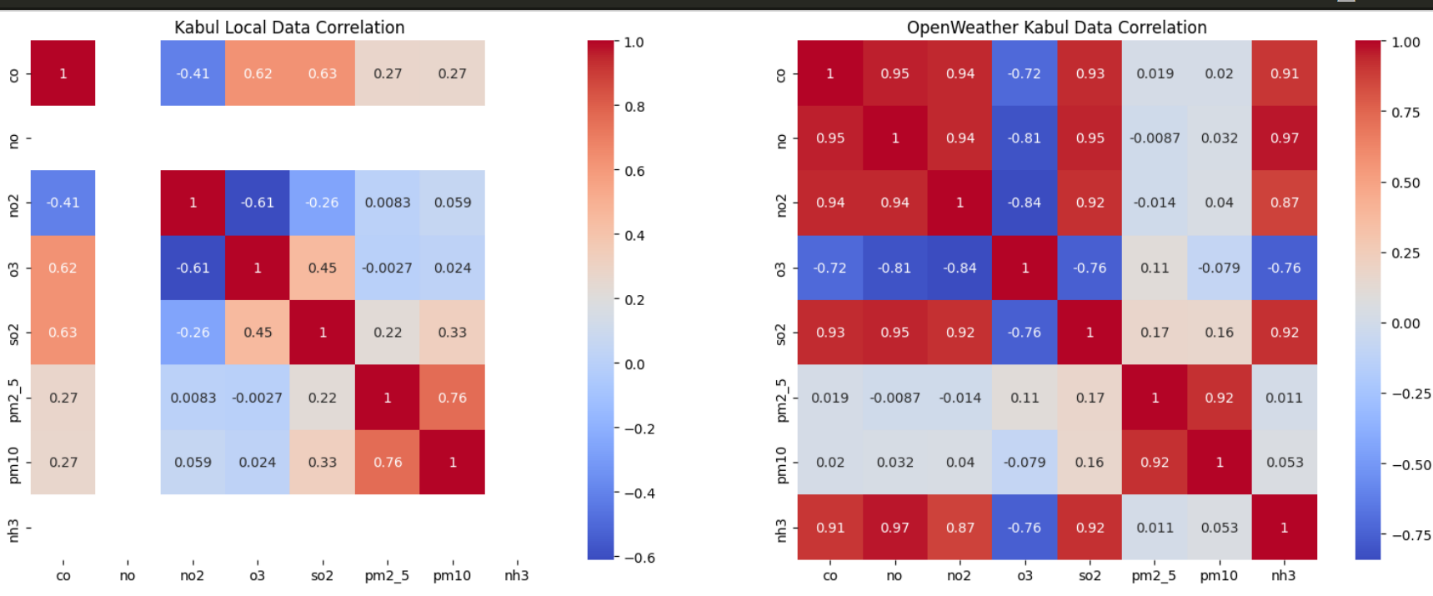
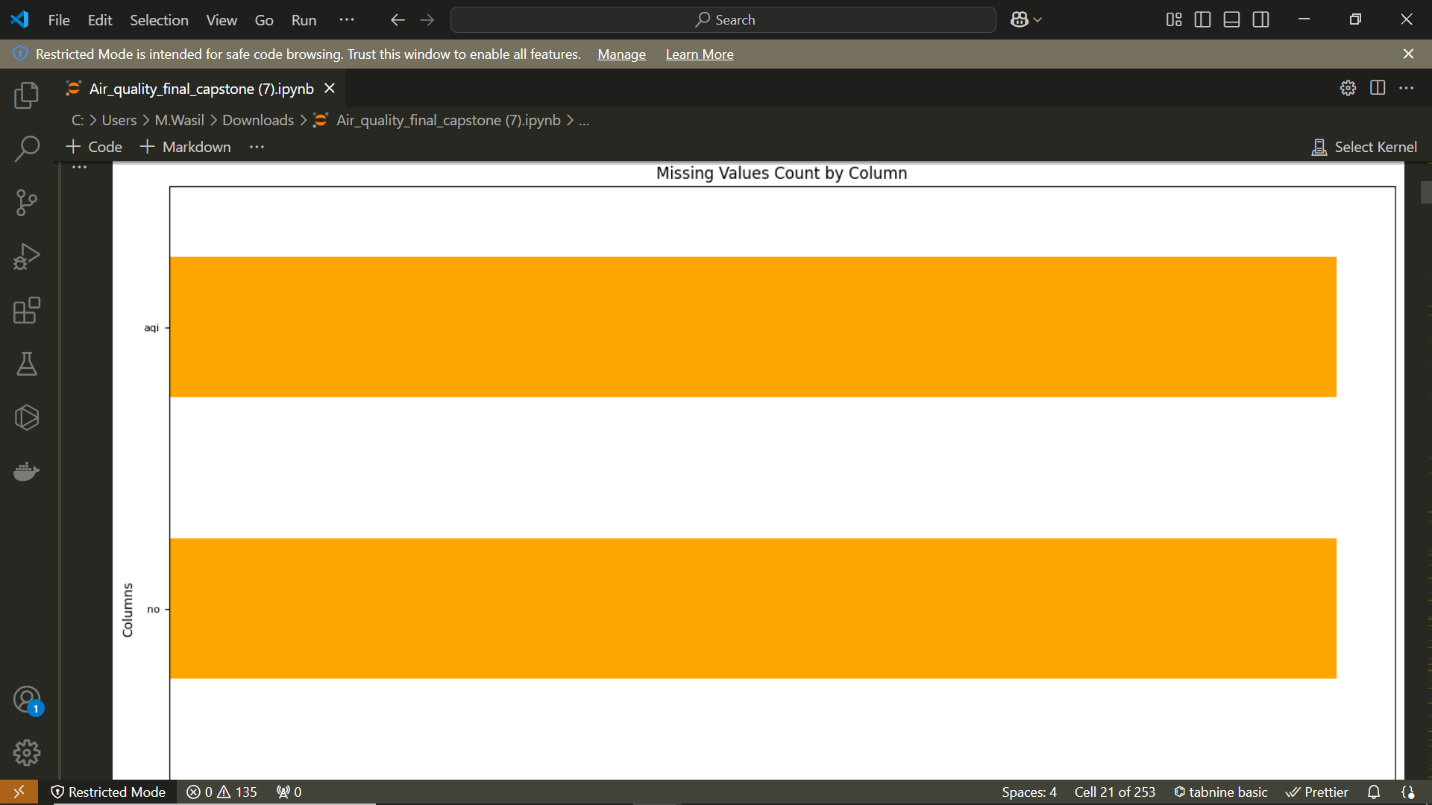


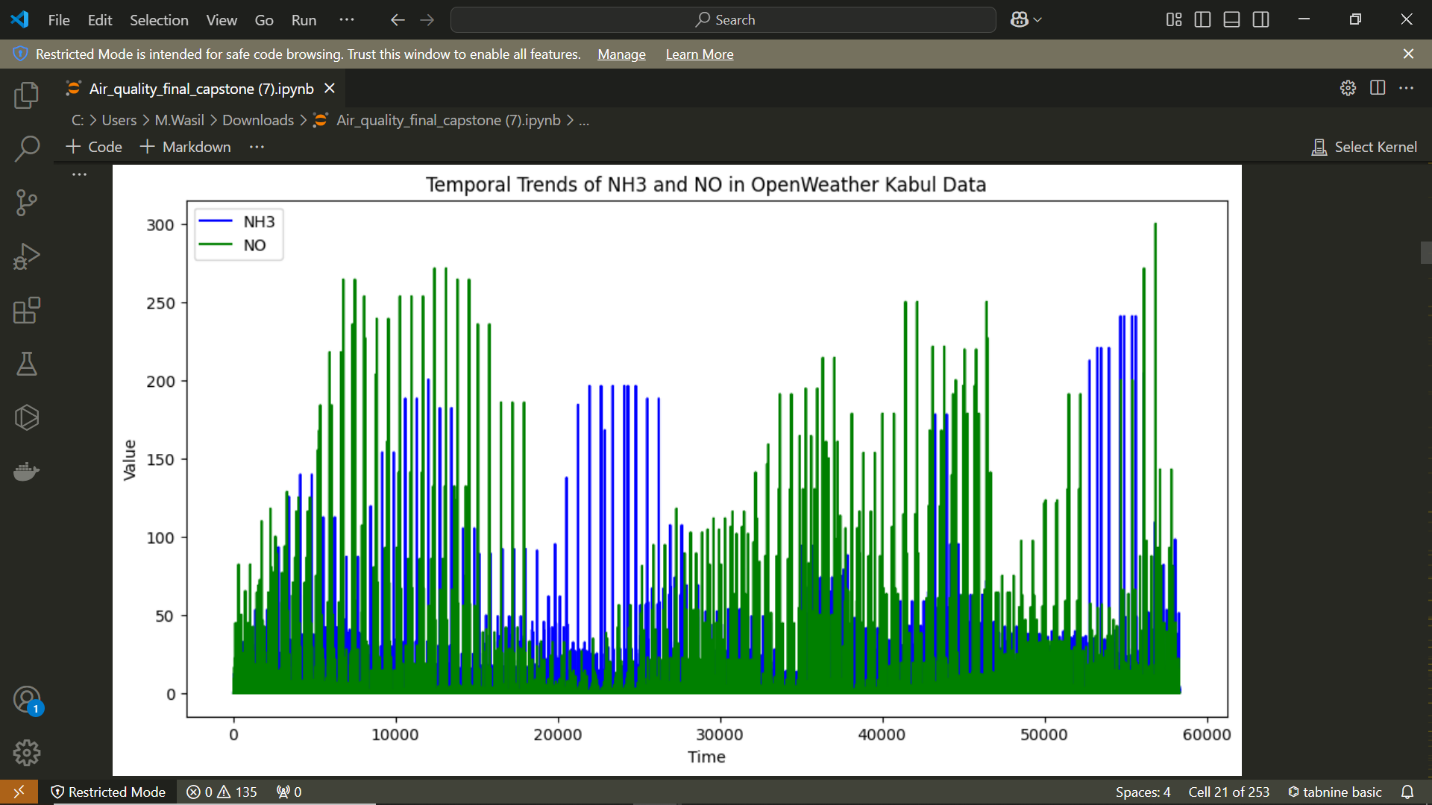
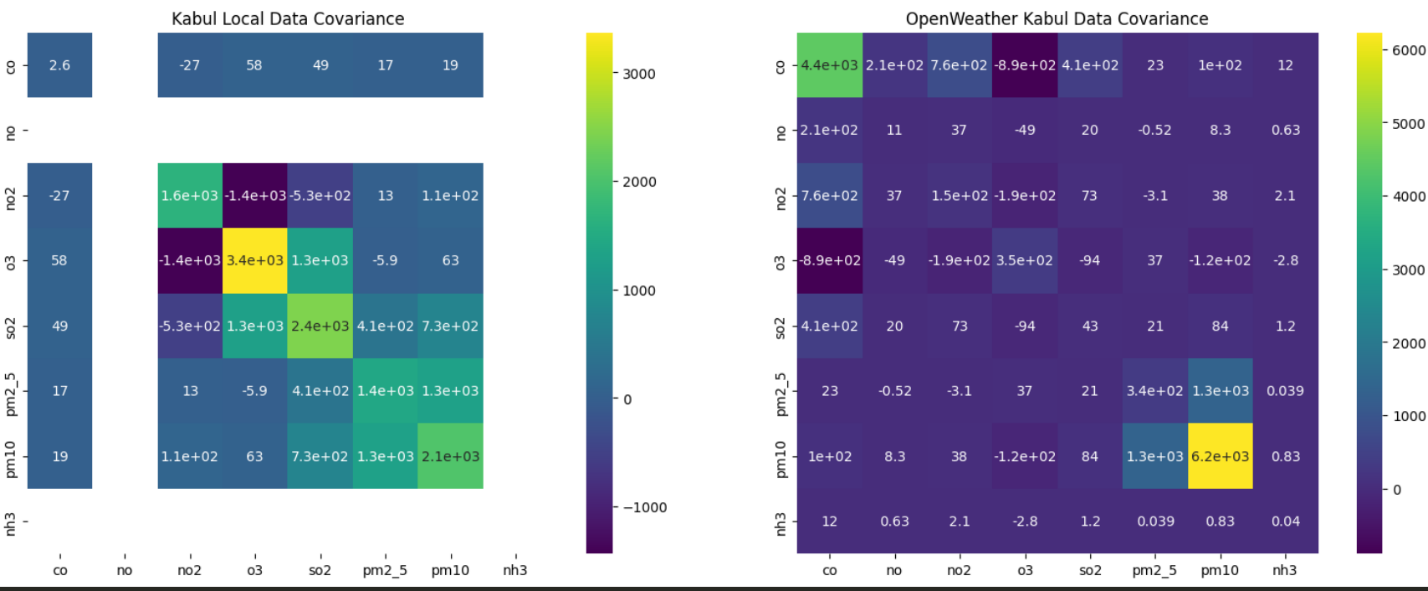


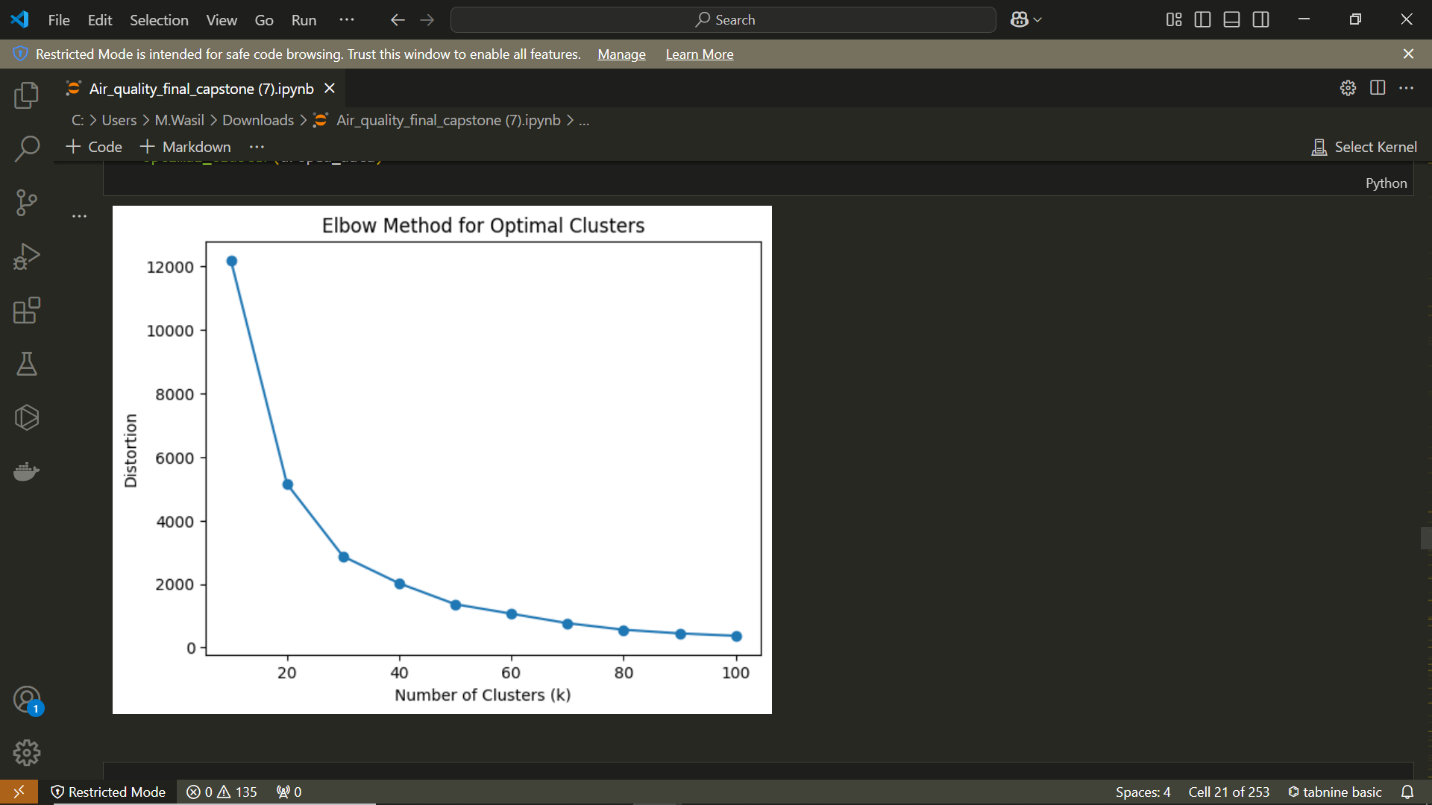
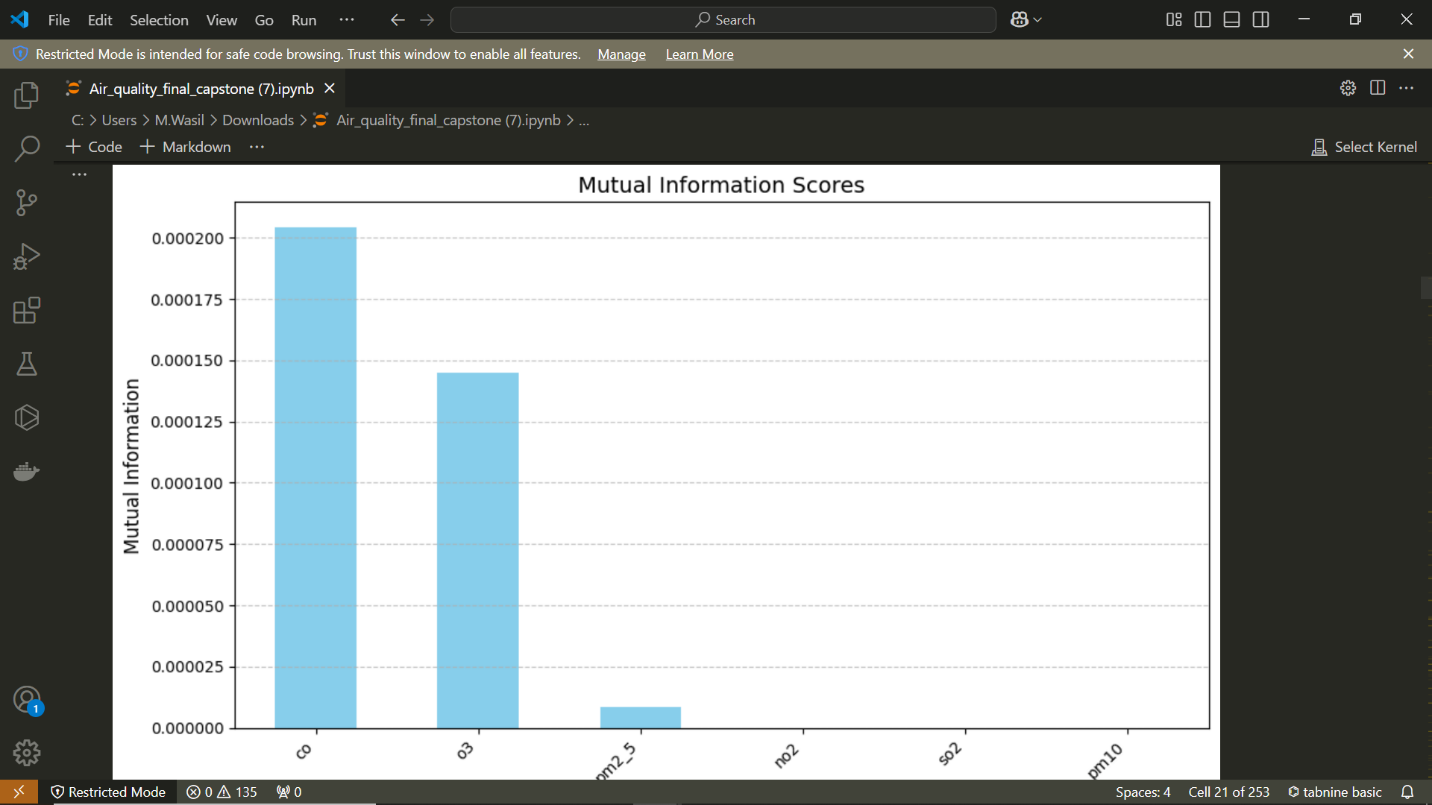


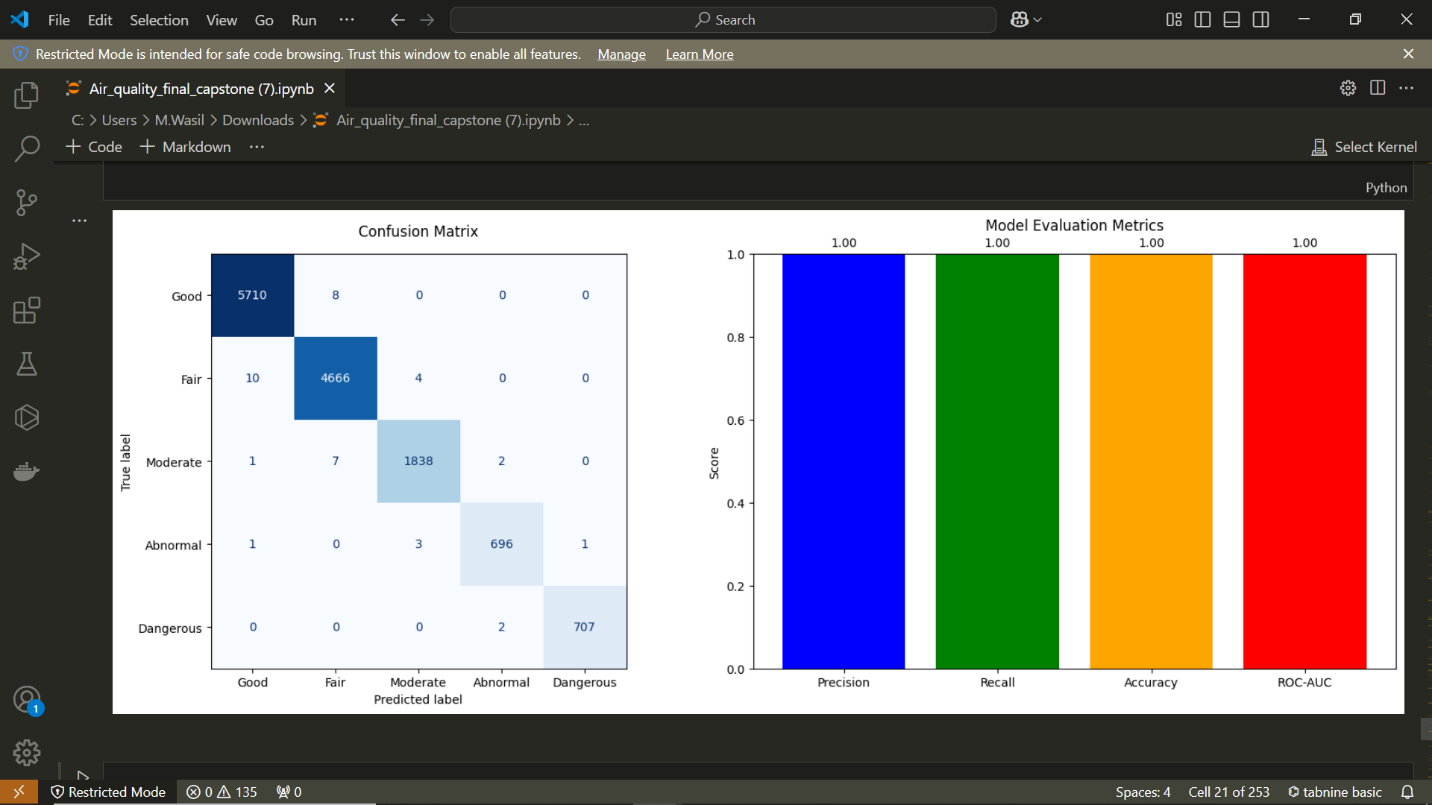
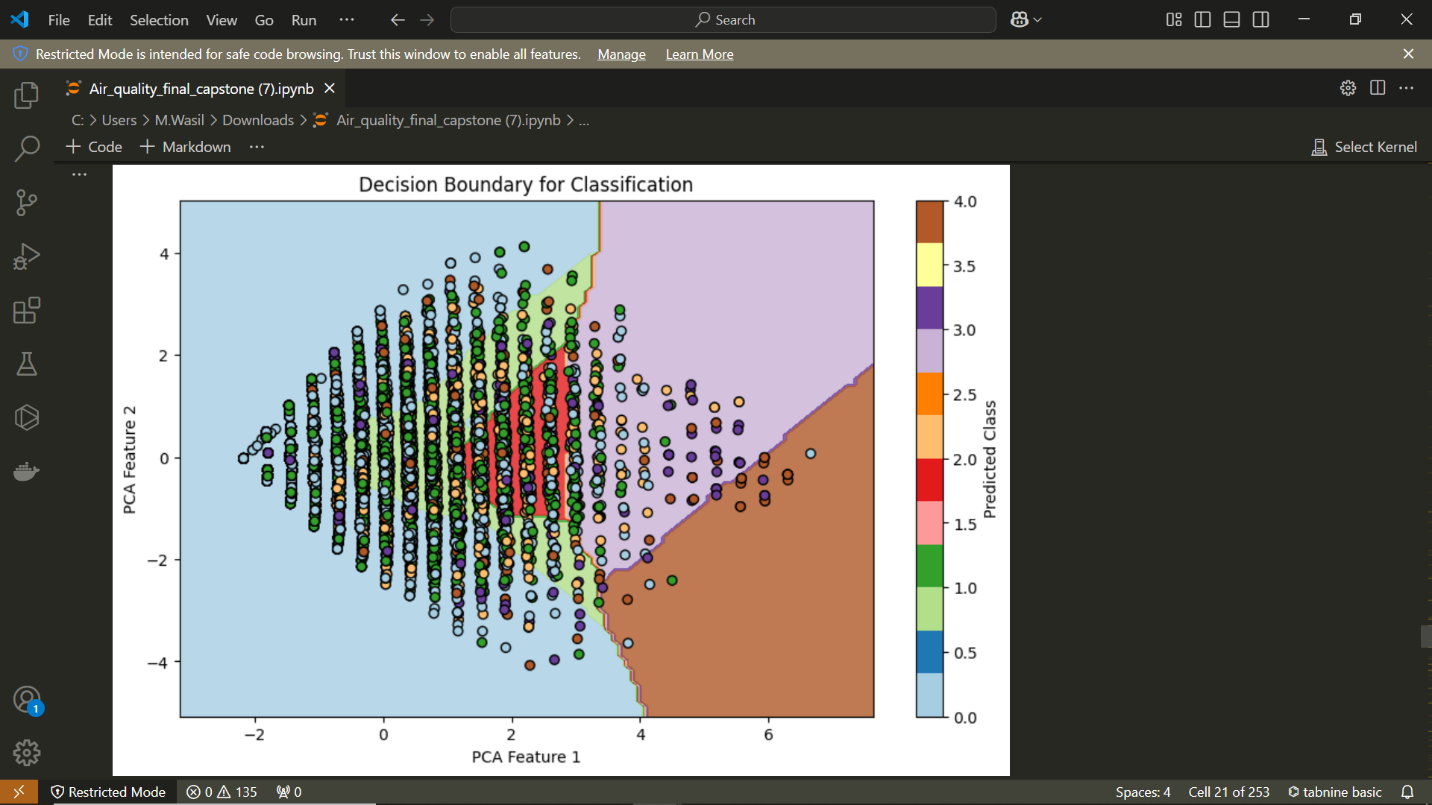












**Conclusion**

The Model Refinement and Test Submission phases were crucial in improving the overall performance of the real-time air quality prediction system. The initial models were refined through hyperparameter tuning, better cross-validation strategies, and feature selection. After training and refining the models, we successfully applied them to the test dataset, where they exhibited significant improvements. The final model, a RandomForestClassifier for classification tasks and TSMixer for regression, was deployed in a real-time monitoring system that accurately predicts air quality.

Challenges encountered during this phase included:

* Addressing overfitting in the initial models.
* Balancing the trade-off between model complexity and performance.

Ultimately, the final model demonstrated strong performance on both the training and test datasets, ensuring reliable predictions in real-world scenarios.

**References**

* "Introduction to Machine Learning with Python" by Andreas C. Müller and Sarah Guido
* Scikit-learn documentation (<https://scikit-learn.org/>)
* OpenWeather API Documentation (<https://openweathermap.org/api>)
* NEPA dataset (provided under official collaboration)
* Yeo-Johnson Transformation and Robust Scaling techniques