**Air Quality Data**

|  |  |  |
| --- | --- | --- |
| **Group Number: 07** | | |
| Last name | First name | Matriculation Number |
| Derler | Anna | 12114827 |
| Fankhauser | Sebastian |  |
| Wallner | Selina | 12111842 |

**Introduction**

In our project we used the “Air Quality” Dataset we got from the UCI Machine Learning Repository. The Dataset contains the responses of an array of 5 metal oxide chemical sensors inside an Air Quality Chemical Multisensor Device, which is deployed in a significantly polluted area in an Italian city. The records contain the hourly averaged responses and the gas concentrations references.

We chose this dataset because we are curious about the environment and pollution levels.

**Code Design**

The classes, methods, and properties in the provided code are designed to be modular, reusable, clear, and easy to maintain.

The first class, "DatasetPreprocessor," handles data preprocessing to ensure the data is clean, formatted, and ready for analysis. This class organizes all preprocessing tasks in one place, making it easy to update the steps without affecting other parts of the code. Private methods hide the internal workings from the user.

The second class, "SimpleBaselineClassifier," offers a simple classifier to compare against more complex models, using common strategies like "most frequent," "uniform," and "constant."

The third class, "Classifier," manages the training and evaluation of different classifiers on the dataset. It provides an interface to train, evaluate, and compare various machine learning models, ensuring clarity and separation of tasks with specific methods for each classifier.

If you were given another two months of time to redesign your code, what would you change?

**Experimental Setup**

We used the “RidgeClassifier”, the “PassiveAggressive” Classifier, the “DecisionTree” Classifier, the “SGDClassifier” and the “SimpleBaselineClassifier”.

From the Dataset we used the CO(GT), C6H6(GT), NMHC(GT), NOx(GT) and the NO2(GT) features.

20% were used for testing and 80% were used for training.

To handle the missing values, we set the values < 0 to 0.0.

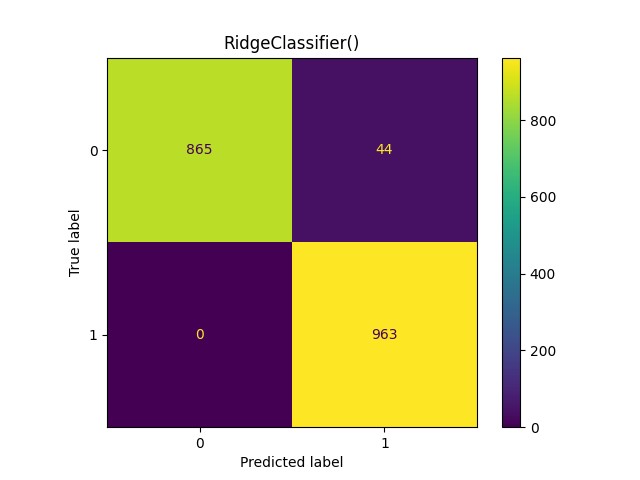
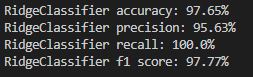
Which metrics did you rely on for evaluation and why? Accuracy, Precision, Recall, F1 Score

Our baseline was the “SimpleBaselineClassifier” with the strategy “uniform”.

**Results and Discussion**

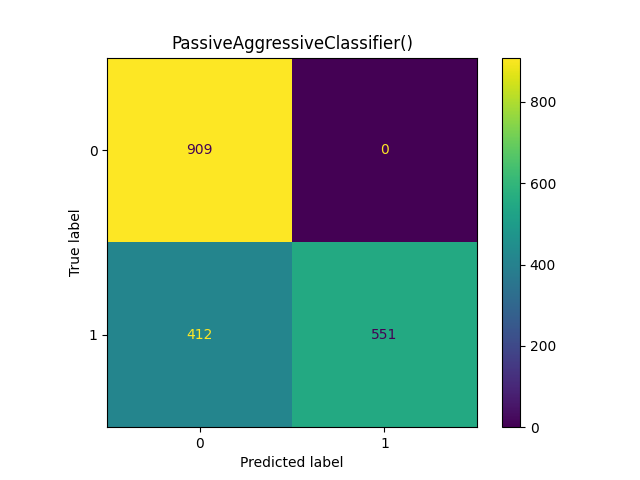
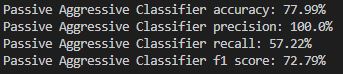
The best Classifier were the “DecisionTree” Classifier and the “SGDClassifer”. The worst Classifier was the “SimpleBaselineClassifier”. Does this change if you use a different metric for evaluation?

In Plot 1, you can see the Confusion Matrix of the "RidgeClassifier." For instance, it shows that in 963 cases, the true label was 1 and the model correctly predicted 1. Additionally, there were 0 cases where the true label was 1 and the model incorrectly predicted 0. This indicates that the classifier is performing well.



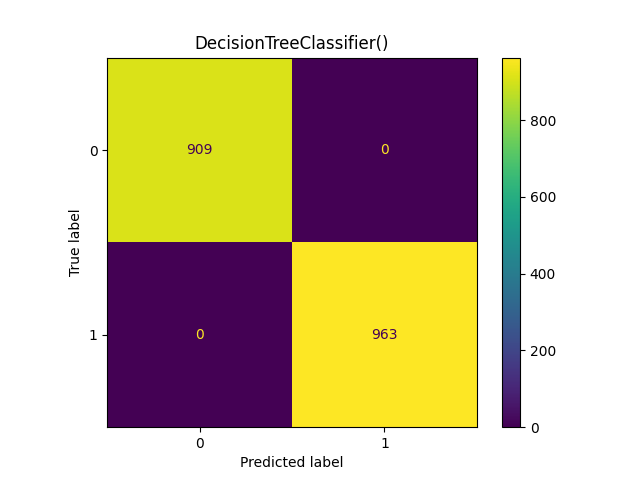
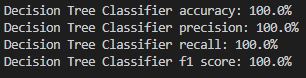
Plot 1: Ridge Classifier Terminal Output 1: Ridge Classifier

In Plot 2, you can see the Confusion Matrix of the "PassiveAggressive" classifier. Notably, there is a high number of true negatives, which is a positive outcome. However, the classifier also shows a high number of false negatives. This suggests that the classifier prioritizes minimizing false positives, potentially at the cost of increasing false negatives.



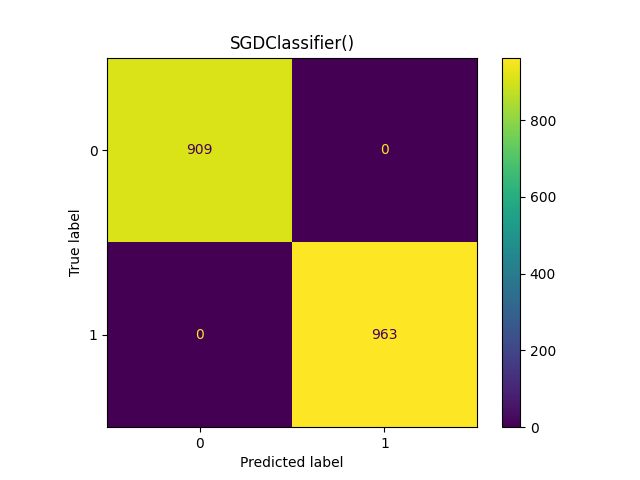
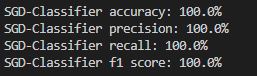
Plot 2: Passive Aggressive Classifier Terminal Output 2: PassiveAggressiveClassifier

In Plot 3, you can see the Confusion Matrix of the "DecisionTree" classifier. The matrix shows only true negatives and true positives, indicating that the classifier performed perfectly without any errors.



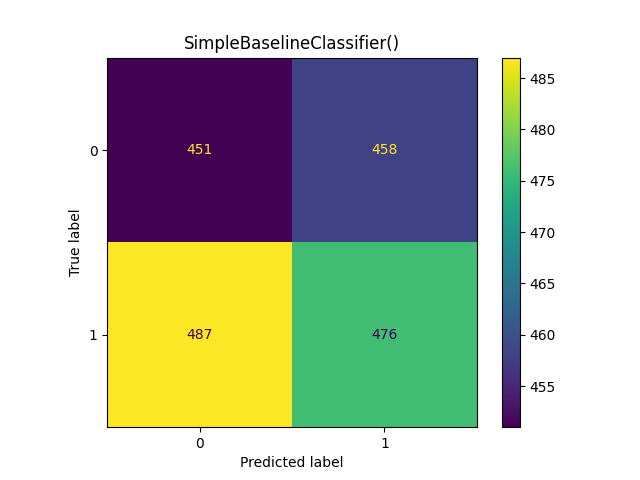
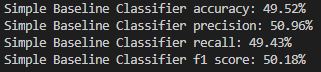
Plot 3: Decision Tree Classifier Terminal Output 3: Decision Tree Classifier

In Plot 4, you can observe the Confusion Matrix of the "SGDClassifier." It displays only true negatives and true positives, indicating flawless performance by the classifier without any errors.



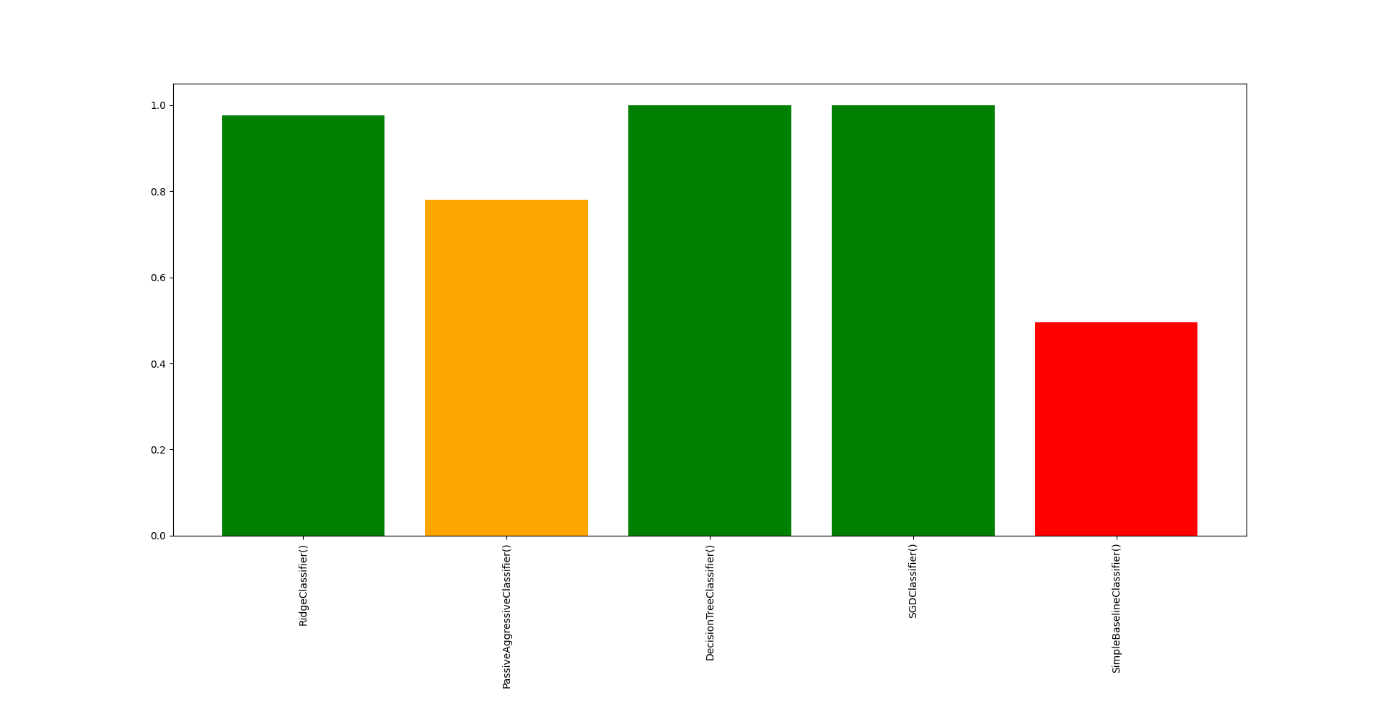
Plot 4: SGD Classifier Terminal Output 4: SGD Classifier

In Plot 5, the Confusion Matrix of the "SimpleBaselineClassifier" is displayed. It reveals that instances for true negatives, false negatives, true positives, and false positives are nearly equal. This indicates that the classifier is not performing well, as there are many false positives and negatives.



Plot 5: Simple Baseline Classifier Terminal Output 5: Simple Baseline Classifier

Plot 6 compares the five classifiers, and it is clear that all of them perform better than the baseline classifier.



Plot 6: Comparison of the Classifiers

**Conclusion**