Fire Alarm Activation System

Project Report – Data Mining

Laurian Duma

Bachelor of Computing Science
Radboud University

student number: s1091563

Project Group :

PR - 119

Alexandru Aioanei

Bachelor of Computing Science
Radboud University

student number: s1103357

ABSTRACT

In this research proposal, we introduce an intelligent fire alarm activation system that utilizes environmental parameters to make informed decisions about fire alarm activation. The proposed system aims to enhance fire safety by providing early detection and alert mechanisms based on data-driven analysis. We plan to use Python's scikit-learn library and various machine learning algorithms, such as decision trees and neural networks, to evaluate and compare their effectiveness in making fire alarm activation decisions. We will employ a Jupyter Notebook for the implementation of these powerful techniques, as well as for gaining insights in the dataset we will be dealing with. By blending machine learning with fire safety, we seek to contribute to more efficient and reliable fire alarm systems.

Application Domain and Research Problem

The rise of technology has increased the importance of fire alarm systems, as they are intended to save human lives. However, designing such systems is crucial, as they have to correctly recognize a fire hazard and not falsely disturb the occupants with their loud and annoying noise. Therefore, every building, room, or indoor area that requires a fire alarm system should be equipped with one that is reliable and accurate.

Related previous work

1) Islam Gomaa et al. proposed a framework for an intelligent fire detection and evacuation system that uses artificial intelligence to make short-term predictions on fire behavior and structural integrity. (https://link.springer.com/article/10.1007/s10694-021-01157-3)

2) M. A. Khan et al. used machine learning to reduce false alarms in fire alarm systems. (https://uobrep.openrepository.com/handle/10547/625006)

3) S. S. Kulkarni and A. S. Kulkarni described a fire detection system using SVM and CNN. The system uses a video camera to capture the images of fire and classify them using support vector machine and convolutional neural network. The system also sends an SMS alert to the authorities. (https://ieeexplore.ieee.org/document/9885405)

DataSet and Data Collection

Our approach is inspired by the methods and techniques used by the previous work, but we use a different dataset and features for our analysis. The dataset that we chose is the Smoke Detection Dataset from Kaggle, which contains various environmental parameters such as temperature, humidity, TVOC, eCO2, pressure, and PM levels. It can be found at the following URL address: https://www.kaggle.com/datasets/deepcontractor/smoke-detection-dataset/data.

The data has been collected using various IoT devices. These have the purpose to measure the environmental parameters related to smoke and fire detection. Two examples of such IoT devices are:

1) Photoelectric Smoke Detector

2) Ionization Smoke Detector

Ensuring a good data set for our analysis is the reason for sampling different environments and fire sources, like: Normal indoor, Normal outdoor, Indoor wood fire, firefighter training area, Indoor gas fire etc.

As it is important to understand the meaning of every feature that is present in our dataset, we make use of a detailed feature description which can be observed in the **Appendix**.

Data Exploration

During our data exploration phase, we conducted a comprehensive analysis of the dataset to gain a deeper understanding of its characteristics and their implications for our predictive modeling task. Our analysis yielded several important findings:

1. **Skewed Distributions:** Some variables, such as PM1.0, PM2.5, NC0.5, NC1.0, and NC2.5, exhibited right-skewed distributions, indicating a concentration of lower values with fewer instances of higher readings. In contrast, Humidity[%], Raw H2, Raw Ethanol, and Pressure[hPa] displayed leftward skewness, suggesting a concentration of higher values with fewer low-value outliers.

2. **Range of Values:** Raw H2 and Raw Ethanol showcased a wider range of values compared to particulate matter and number concentration variables. This suggests the presence of diverse information, potentially valuable for our predictive models.
3. **Correlations:** Strong positive correlations were observed among particulate matter and number concentration metrics, implying redundancy. Feature selection will be considered to address potential overfitting.
4. **Modest Correlations:** Certain variables, such as Raw H2, TVOC, and eCO2, exhibited modest correlations with others. Despite weak linear relationships, these features may provide unique insights relevant to our classification goals.
5. **Class Imbalance:** The 'Fire Alarm' variable demonstrated class imbalance, with significantly more instances in one class than the other. This imbalance will be accounted during model training and evaluation.
6. **Outliers:** Outliers, particularly in air quality measures and particulate matter readings, may correspond to critical environmental events, such as potential fires, which our predictive model aims to identify. Thus, we decided to keep them for our further analysis.
7. **Duplicates and Missing Values:** Our analysis did not uncover any duplicates or missing values in this specific dataset, requiring no further actions will be taken in this regard.

Data Preprocessing

In our data preprocessing phase, we employed various techniques to prepare the dataset for effective analysis and modeling. To address the sensitivity of certain data mining algorithms to attribute magnitude, we performed both normalization and standardization on our dataset. Standardization was applied to achieve a normal distribution and reduce the impact of outliers, resulting in data with a mean of 0 and unit variance. On the other hand, normalization was employed to limit attribute values to the [0, 1] range.

Additionally, we recognized the need for dimensionality reduction due to the presence of highly correlated particulate matter attributes. To achieve this, we applied Principal Component Analysis (PCA), which reduced the dimensionality from five correlated attributes to two principal components, denoted as 'PCA1' and 'PCA2.' These components captured approximately 99% of the total variance, retaining the essential information while reducing multicollinearity. This dimensionality reduction not only mitigated the risk of overfitting but also enhanced computational efficiency.

Furthermore, we addressed the issue of skewness in the feature distributions. For positively skewed features such as 'TVOC[ppb],' 'eCO2[ppm],' and 'PCA1,' we applied a log transformation to reduce the influence of extreme high values. This transformation brought the distributions closer to normality, aligning with the assumptions of our predictive modeling (Figures 6 – 11 from Appendix). For features within a specified skewness range, more exactly the interval [-4, 4], we opted to leave them unaltered, considering them sufficiently normal for our analysis.

To strengthen the robustness of our model evaluation and effectively tackle the issue of class imbalance, we have adopted a stratified 10-fold cross-validation strategy. Stratified cross-validation offers a distinct advantage when confronting class imbalance, since it ensures that each fold maintains the same distribution of class labels as the original dataset. This approach not only enhances the accuracy and reliability of our model evaluation but also strengthens our confidence in the model's ability to generalize effectively to unseen data.

Data Modelling and Data Evaluation

We combined these two main parts of the project to train and test different machine learning models on various datasets. This way, we can assess the impact of our data preprocessing methods on the model performance. We used seven datasets for this purpose: df (raw dataset), std (standardized dataset), std_pca (standardized dataset with dimensionality reduction by PCA), std_pca_trans (std_pca with log transformation to reduce skewness), norm (normalized dataset), norm_pca (normalized dataset with dimensionality reduction by PCA), df_pca (raw dataset with dimensionality reduction by PCA).

1. CatBoost Classifier

The selection of the CatBoost classifier as our primary model for training and evaluating our datasets is well-founded, considering the specific characteristics of our data and the classifier's robust capabilities. CatBoost is a gradient boosting framework that sequentially constructs decision trees to correct errors, allowing it to capture complicated relationships within the data. The inclusion of built-in L2 regularization mitigates the risk of overfitting, while its symmetric tree structure enhances its resistance to outliers. By optimizing learning rates, we can further strengthen the model's robustness and generalization. Additionally, CatBoost's customizable loss function and overfitting detection mechanisms increase its overall effectiveness.

Hence, taking advantage of CatBoost's versatile capabilities, we hypothesize that it will achieve outstanding results when applying it to previously unseen data.

Analyzing performance metrics

Across all seven datasets, our CatBoost classifier consistently exceptional performance, achieving near-perfect accuracy scores in each case (Figure 12 from Appendix). What's particularly remarkable is the consistency of these results, even when various preprocessing techniques are applied to our raw data. These findings yield valuable insights:

1. **Robust Model Performance:** These results reaffirm CatBoost's reputation for robustness and its capacity to handle noisy or unprocessed data effectively.
2. **Sufficient Complexity:** It appears that the datasets and tasks may not necessitate extensive preprocessing due to CatBoost's inherent ability to capture complex data relationships.

However, an intriguing observation emerges when comparing the 'df_pca' dataset to the preprocessed ones. The classifier trained and tested on 'df_pca' consistently outperforms the others across various metrics, including accuracy, precision, and F1 score. This phenomenon may be attributed to specific dataset characteristics that align exceptionally well with CatBoost's strengths.

The variations in performance between datasets like 'std' and 'std_pca' compared to 'df_pca' could be attributed to multiple factors. It's possible that certain preprocessing steps introduced noise or altered the data's natural structure, resulting in decreased performance. Additionally, overfitting might also occur when the classifier is trained and tested on the latter dataset.

While these results are promising, we remain vigilant about overfitting. To make informed decisions about deploying the CatBoost classifier, we will conduct a detailed analysis of the most influential features during training, focusing on the 'df_pca' dataset, which exhibits the highest performance metrics. This analysis will provide valuable insights into the model's generalization capabilities.

Analyzing the most significant features

Feature importance scores serve as a metric for assessing the significance of individual attributes within a machine learning model's decision-making process. A higher importance score indicates a more substantial impact on the model's predictions. In our provided list of feature importance (Figure 13 from Appendix), 'Pressure[hPa]' prominently stands out with a score of 41.85, making it the most crucial predictor, closely followed by 'TVOC[ppb]' at 20.91.

Upon a closer look, the significance of 'Pressure[hPa]' in the model's decision-making process becomes apparent, given its pivotal role in our domain of study. However, the dominance of this single feature may lead to over-reliance, potentially overshadowing the contributions of other attributes in determining fire alarm triggers. To investigate this further, we undertake an experiment by excluding the 'Pressure[hPa]' feature from our dataset and training a new CatBoost classifier. Subsequently, we evaluate the model's performance using various metrics and reassess the importance of each feature.

Analyzing the performance on datasets where the most significant feature is dropped

The exclusion of the 'Pressure[hpa]' feature has minimal impact on model performance, implying its non-crucial role in classification tasks or the presence of other informative attributes (Figure 14 from Appendix). While we do observe minor variations in metrics, the 'df_pca' dataset no longer maintains its performance lead after the removal of the 'Pressure[hPa]' feature, suggesting its role in classification. This is possibly due to its interactions with other dataset features. The slight decrease in metrics can be attributed to the remaining features in the datasets which still provide ample information and discriminatory power for the classifier to make accurate predictions.

Regarding feature importance, 'TVOC[ppb]' emerges as the most influential attribute, closely followed by 'Raw Ethanol' and 'Humidity[%]' (Figure 15 from Appendix).

Analyzing the Confusion Matrix over “std_pca” dataset

The provided confusion matrix (Figure 16 from Appendix) strongly supports the hypothesis that the CatBoost classifier is performing exceptionally well. With 17,863 true negatives and 44,753 true positives, the classifier demonstrates an outstanding ability to accurately classify both negative and positive classes. The presence of only 4 false negatives and 10 false positives reflects an exceptionally low rate of misclassifications, indicating an high level of accuracy. This outcome underscores the classifier's success in capturing the underlying data patterns.

Analyzing the learning curve over “std_pca” dataset

The learning curve of the CatBoost classifier, trained on the 'std_pca' dataset, provides valuable insights into its performance (Figure 17 from Appendix). The high mean training score of 0.9999 suggests that the classifier successfully captures the underlying patterns within the training data. However, the mean validation score of 0.8053, although still indicative of good predictive performance, reveals a noticeable gap when compared to the training score.

This discrepancy raises concerns about potential overfitting, where the model excels on familiar data but struggles to generalize to new, unseen data. Fortunately, the cross-validation score remains relatively stable with increasing training set sizes, manifesting a slight upward trend. This trend is promising and indicates that the model benefits from more data.

The abrupt increase in the cross-validation score towards the end of the learning curve may be attributed to various factors, including data characteristics, model capacity, random variation, but also overfitting.

The plateau observed at the end of the validation curve suggests that further increases in training data may not significantly improve model performance. This implies that the model has effectively learned generalizable patterns, and additional data offers diminishing results.

The minimal standard deviation in cross-validation scores is a positive indicator, signifying consistent model performance across different data subsets. This consistency reflects the model's stable accuracy, independent of specific training set samples.

2. Multilayer Perceptron Classifier

The Multilayer Perceptron (MLP) classifier is a versatile neural network model, well-suited for handling nonlinear data relationships in classification tasks.

Through data standardization, we expect the MLP classifier to improve classification accuracy and robustness. We anticipate strong performance across various datasets, thanks to the model's ability to uncover hidden patterns in standardized data, but not only.

Pre-pruning methods, including 'early_stopping' and related hyperparameters, are employed to prevent overfitting and promote generalization while using default values for other parameters.

Analyzing performance metrics

The MLP classifier's performance metrics vary across datasets, highlighting the significant impact of data preprocessing and features on its effectiveness (Figure 18 from Appendix). The MLP classifier trained and tested on datasets like 'std_pca' and 'std' achieves high accuracy, precision, recall, F1 Score, and AUC. In addition, it demonstrates the vital role of standardization and PCA transformation in enhancing classifier performance.

Conversely, the MLP classifier trained and tested on 'norm_pca', 'df_pca', and 'norm' datasets exhibits lower performance metrics, possibly due to the less beneficial impact of normalization or less effective PCA transformation, leading to suboptimal results.

The 'df' dataset, representing raw, unprocessed data, still achieves good performance, although slightly lower than standardized datasets.

Analyzing the learning curve over “std_pca” dataset

The learning curve for the MLP Classifier on the 'std_pca' dataset shows a high training accuracy of approximately 99.83%, contrasting with a lower cross-validation accuracy of about 0.8009. This discrepancy suggests potential overfitting (Figure 19 from Appendix).

The small standard deviation in validation scores (0.0077) indicates consistent performance across different training set sizes.

The curve reveals an initial sharp increase in cross-validation accuracy as the training set size grows, but this improvement levels off and may even dip slightly with more data. This implies that beyond a certain point, adding more data does not significantly enhance the model's ability to learn generalizable patterns. The spike at the curve's end is similar to the one encountered in the previous example and may be caused by the same reasons.

The plateau in the training score, regardless of increased training data, suggests that the model has the capacity to almost memorize the training data, indicating a potential overfitting issue.

Analyzing the Confusion Matrix over “std_pca” dataset

The confusion matrix for the MLP classifier on the 'std_pca' dataset (Figure 20 from Appendix) displays similarities to the one generated for the CatBoost classifier. Both classifiers tend to produce more false positive mistakes (Type I errors) than false negative misclassifications (Type II errors). This consistency in their error patterns is in line with their high-performance metrics, where they achieved impressive accuracy and precision scores.

While differences in the scale of errors between the MLP and CatBoost classifiers were anticipated due to their distinct learning mechanisms, their shared tendence on minimizing false negatives is particularly advantageous for our application. In our context, the consequences of failing to detect a potential fire hazard (false negatives) are more significant than occasionally classifying a normal situation as a fire hazard (false positives). This prioritization of minimizing false negatives aligns perfectly with our focused objective, ensuring a robust fire detection system that prioritizes early and accurate warnings.

3. Dummy Classifier

Before moving to the next project phase for a detailed analysis, we'll train a Dummy Classifier as a baseline comparison. This classifier assigns all instances to the majority class, making it unfit for ROC curve and AUC evaluation due to a lack of probability estimates and threshold variation. Thus, we will focus on other relevant metrics.

Analyzing performance metrics and Confusion Matrix

Our results arise from a highly imbalanced dataset, with the majority class as the positive class (Figure 21 from Apendix). The Dummy Classifier's 'majority class strategy' leads to these metrics. While accuracy seems reasonable, the model lacks predictive power for the negative class. High recall appears impressive but sacrifices correct negative identifications.

Thereby, while the Dummy Classifier serves as a baseline, its apparent success in recall and F1 score comes from predicting the majority class, providing no meaningful insights for the negative class. This highlights the need for diverse performance metrics, beyond accuracy.

4. Checking for statistical significance

In our upcoming analysis, we will perform a statistical significance test to explore the observed performance difference between the CatBoost and MLP classifiers. To elaborate this test, we establish the null hypothesis (H0) and the alternative hypothesis (H1). H0 suggests that no significant difference exists in error proportions between the two classifiers on the dataset under study, while H1 asserts a significant difference. This test aims to determine if the observed performance gap between the classifiers is statistically meaningful or merely due to chance.

Analyzing the results of the statistical significance test and the Contingency Matrix

The statistic value in the McNemar test represents the chi-squared statistic, reflecting the discrepancy in instances where one classifier is correct while the other is not. A larger statistic value indicates a more substantial difference between the classifiers. In our case, the statistic value is 65.419, signifying a notable performance contrast (Figure 22 from Apendix).

However, the statistic value's significance is assessed alongside the p-value. In our example, the extremely low p-value (0.000) indicates a significant performance difference. Therefore, we reject the null hypothesis and conclude that the CatBoost classifier consistently outperforms the MLP classifier.

Main Results and Insights

Robust Performance of CatBoost Classifier:

The CatBoost classifier demonstrated exceptional performance across various datasets, indicating its robustness and ability to handle complex and noisy data effectively.

The classifier's performance remained consistent even with different preprocessing techniques, highlighting its capacity to capture detailed relationships within the data.

Significance of PCA Preprocessing:

The 'df_pca' dataset, which supported PCA on the raw data, consistently outperformed other datasets in terms of accuracy, precision, and F1 score. This suggests that PCA preprocessing aligns well with CatBoost's strengths and may simplify the learning process, reducing the need for extensive preprocessing.

Feature Importance Analysis:

'Pressure[hPa]' emerged as the most crucial predictor, followed by 'TVOC[ppb]', indicating their significant impact on the model's predictions. The experiment excluding 'Pressure[hPa]' showed minimal impact on model performance, suggesting the presence of other informative attributes and the classifier's adaptability.

Performance Evaluation of MLP Classifier:

The MLP classifier showed varying performance across datasets, illustrating the impact of data preprocessing on its effectiveness. Datasets with standardization and PCA ('std_pca' and 'std') achieved better performance metrics, highlighting the importance of these preprocessing steps for MLP classifier.

Learning Curve and Confusion Matrix Analysis:

The learning curves for both classifiers revealed potential overfitting, indicated by high training scores but lower validation scores.

The confusion matrices showed a low rate of misclassifications, particularly for false negatives, aligning with the safety-focused objective of minimizing missed fire hazards.

Baseline Comparison with Dummy Classifier:

The Dummy Classifier's performance highlighted the importance of diverse performance metrics in imbalanced datasets. It served as a baseline but lacked predictive power for the negative class.

Statistical Significance Test Between Classifiers:

The McNemar test revealed a significant performance difference between the CatBoost and MLP classifiers, favoring the CatBoost classifier. This was indicated by a large chi-squared statistic value and a very low p-value.

Analysis of the results and evaluation of the project outcome

The project successfully demonstrated the applicability and effectiveness of advanced machine learning models in the domain of fire safety. The CatBoost classifier, in particular, emerged as a robust and reliable tool for fire alarm activation decisions.

The project also highlighted the critical role of appropriate data preprocessing and feature selection in optimizing classifier performance, aligning with our initial assumptions.

While our project outcomes generally met our expectations, we did encounter some overfitting concerns that were not initially anticipated. This gap can be attributed to the presence of outliers which we considered valuable for our analysis. However, we have discussed plans to address this issue in the future section.

Possible future improvements

Addressing Outliers:

Initially, we included all data objects in our analysis, despite observing potential outliers in the box plots. As we examined the learning curves for both classifiers, we noted abrupt peaks at the end of these curves. Additionally, our analysis hinted at possible overfitting, although its extent remains uncertain. To mitigate overfitting, conducting an anomaly detection analysis could prove valuable. This analysis would enable us to identify and remove anomalous data objects, significantly reducing the risk of overfitting.

Hyperparameter Tuning:

Further enhancing our classifiers' performance while protecting against overfitting can be achieved through hyperparameter tuning. Employing nested cross-validation, we can optimize hyperparameters to achieve a balance between model complexity and generalization.

Exploring Alternative Classifiers:

Expanding our horizons beyond CatBoost and MLP classifiers, exploring alternative models like Support Vector Machines and k Nearest Neighbors could offer valuable insights. Comparing their performance on our problem may provide a deeper understanding of how different classifiers address our specific challenges.

These proposed improvements align with our current findings, aiming to refine our approach and enhance model performance. However, it's essential to continuously evaluate and adapt our strategies as we uncover new insights and challenges in our dataset and problem domain.

Insights into the code used to generate the results

Please refer to the provided Jupyter Notebook for a detailed, step-by-step insight into our methodology and code implementation, ensuring a thorough understanding of our project's execution.

References

We found the following notebooks to be particularly helpful during the initial phases of our project, specifically in data exploration and preprocessing:

https://www.kaggle.com/code/hossamgalal68/smoke-detection-eda-score-100

https://www.kaggle.com/code/narminhumbatli/smoke-detection-classification

https://www.kaggle.com/code/utkarshsaxenadn/smoke-detection-dense-network-acc-99-98

https://www.kaggle.com/code/dinanabil811/ann-for-smoke-detection

In addition, we gathered information corresponding to the meaning of each feature used in our dataset, as well as details regarding the data collection process from the following URL address on Kaggle:

https://www.kaggle.com/datasets/deepcontractor/smoke-detection-dataset/data

Appendix

Feature description:

UTC - The time when experiment was performed.
Temperature - Temperature of Surroundings. Measured in Celsius
Humidity - The air humidity during the experiment.
TVOC - Total Volatile Organic Compounds. Measured in ppb (parts per billion)
eCo2 - CO2 equivalent concentration. Measured in ppm (parts per million)
Raw H2 - The amount of Raw Hydrogen present in the surroundings.
Raw Ethanol - The amount of Raw Ethanol present in the surroundings.
Pressure - Air pressure. Measured in hPa
PM1.0 - Paticulate matter of diameter less than 1.0 micrometer .
PM2.5 - Paticulate matter of diameter less than 2.5 micrometer.
NC0.5 - Concentration of particulate matter of diameter less than 0.5 micrometers.
NC1.0 - Concentration of particulate matter of diameter less than 1.0 micrometers.
NC2.5 - Concentration of particulate matter of diameter less than 2.5 micrometers.
CNT - Simple Count.
Fire Alarm - (Reality) If fire was present then value is 1 else it is 0.

Addressing skewness in variables distribution:

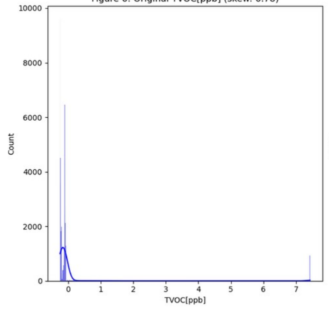


Figure 6: Original TVOC[ppb] (skew: 6.78)

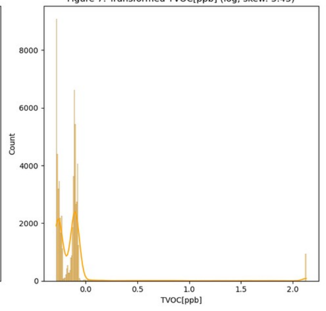


Figure 7: Transformed TVOC[ppb] (log, skew: 5.43)

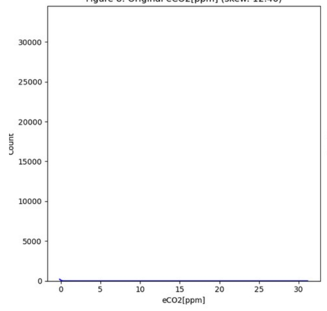


Figure 8: Original eCO2[ppm] (skew: 12.46)

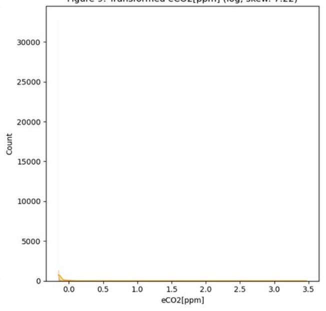


Figure 9: Transformed eCO2[ppm] (log, skew: 7.22)

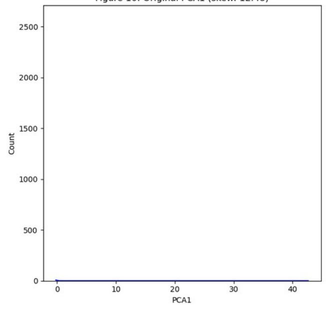


Figure 10: Original PCA1 (skew: 12.48)

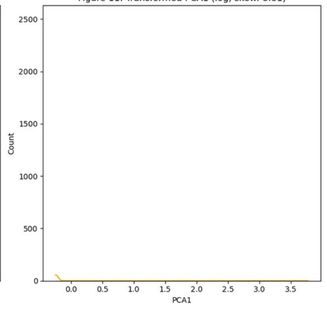
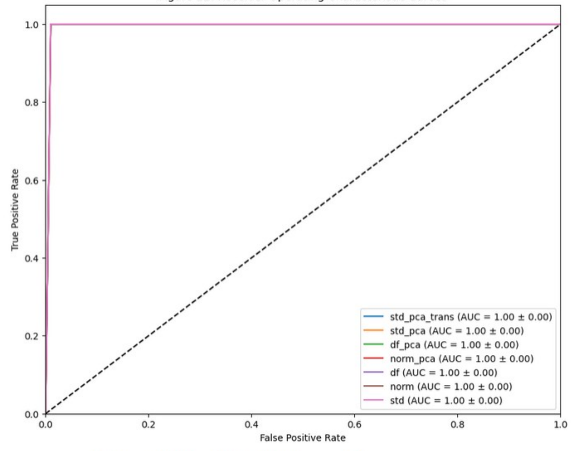


Figure 11: Transformed PCA1 (log, skew: 8.81)

CatBoostt Classifier performance metrics:

Figure 12: Receiver Operating Characteristic Curves



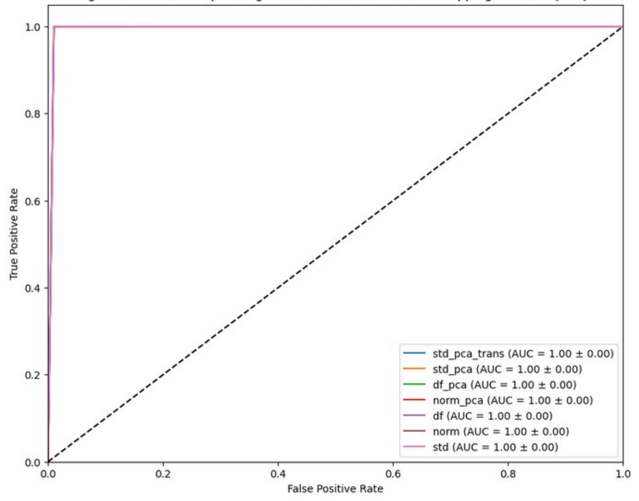
```
               Accuracy  Precision    Recall  F1 Score       AUC
std_pca_trans  0.999776   0.999777  0.999911  0.999844  0.999999
std_pca        0.999776   0.999777  0.999911  0.999844  0.999999
df_pca         0.999824   0.999844  0.999911  0.999877  0.999999
norm_pca       0.999792   0.999799  0.999911  0.999855  0.999999
df             0.999776   0.999777  0.999911  0.999844  0.999999
norm           0.999776   0.999777  0.999911  0.999844  0.999999
std            0.999776   0.999777  0.999911  0.999844  0.999999
```

Features significances:

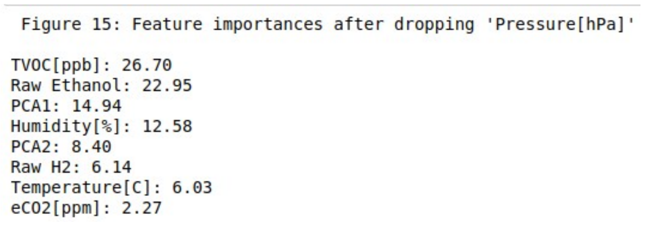
```
Figure 13: Features importances:

Pressure[hPa]: 41.85
TVOC[ppb]: 20.91
PCA1: 8.41
Raw Ethanol: 5.88
Humidity[%]: 4.02
Temperature[C]: 3.42
Raw H2: 2.08
eCO2[ppm]: 0.46
```

CatBoostt Classifier performance metrics on datasets with ‘Pressure[hpa]’ dropped and features significances:

Figure 14: Receiver Operating Characteristic Curves after dropping "Pressure[hPa]"



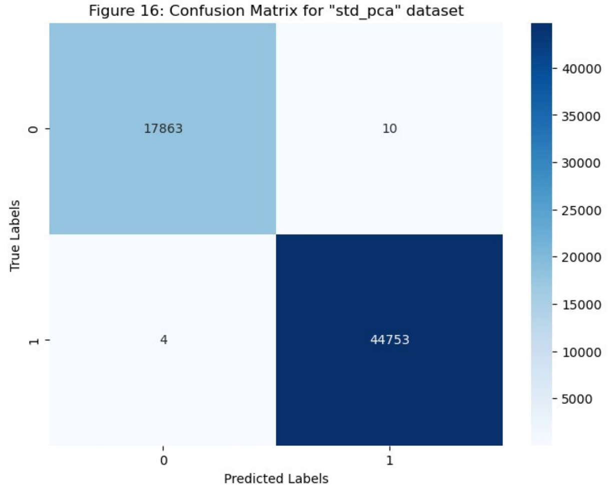
```
               Accuracy  Precision    Recall  F1 Score       AUC
std_pca_trans  0.999745   0.999754  0.999888  0.999821  0.999992
std_pca        0.999745   0.999754  0.999888  0.999821  0.999992
df_pca         0.999665   0.999687  0.999844  0.999765  0.999992
norm_pca       0.999745   0.999732  0.999911  0.999821  0.999990
df             0.999713   0.999710  0.999888  0.999799  0.999991
norm           0.999713   0.999710  0.999888  0.999799  0.999991
std            0.999713   0.999710  0.999888  0.999799  0.999991
```



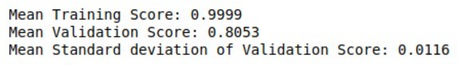
```
Figure 15: Feature importances after dropping 'Pressure[hPa]'

TVOC[ppb]: 26.70
Raw Ethanol: 22.95
PCA1: 14.94
Humidity[%]: 12.58
PCA2: 8.40
Raw H2: 6.14
Temperature[C]: 6.03
eCO2[ppm]: 2.27
```

Confusion Matrix over “std_pca” dataset (CatBoost Classifier):

Figure 16: Confusion Matrix for "std_pca" dataset

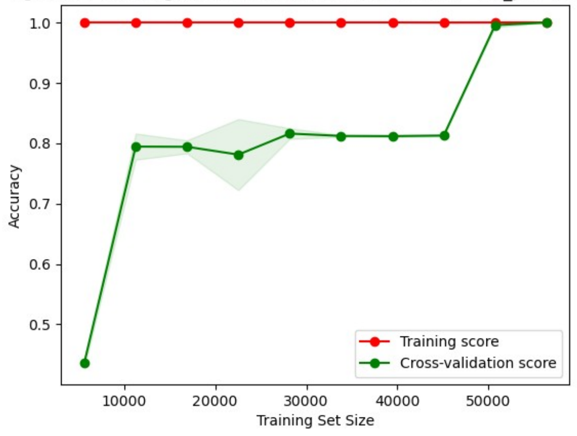


Learning curve over “std_pca” dataset:



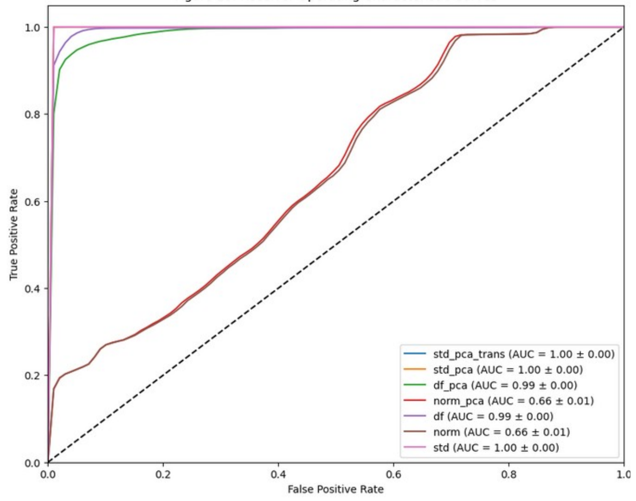
```
Mean Training Score: 0.9999
Mean Validation Score: 0.8053
Mean Standard deviation of Validation Score: 0.0116
```

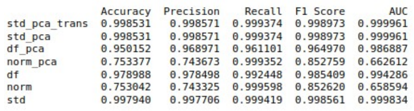
Figure 17: Learning curve for CatBoost classifier over "std_pca" dataset



CatBoostt Classifier performance metrics

Figure 18: Receiver Operating Characteristic Curves



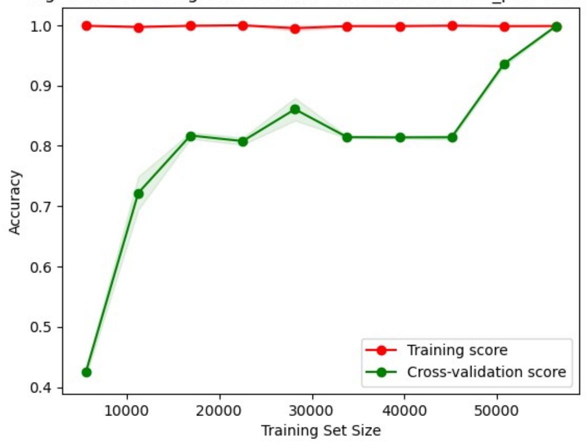


```
              Accuracy  Precision    Recall  F1 Score       AUC
std_pca_trans 0.998531   0.998571  0.999374  0.998973  0.999961
std_pca       0.998531   0.998571  0.999374  0.998973  0.999961
df_pca        0.950152   0.968971  0.961101  0.964970  0.986887
norm_pca      0.753377   0.743673  0.999352  0.852759  0.662612
df            0.978988   0.978498  0.992448  0.985409  0.994286
norm          0.753042   0.743325  0.999598  0.852620  0.658594
std           0.997940   0.997706  0.999419  0.998561  0.999834
```

Learning curve over “std_pca” dataset:

```
Mean Training Score: 0.9983
Mean Validation Score: 0.8009
Mean Standard deviation of Validation Score: 0.0077
```

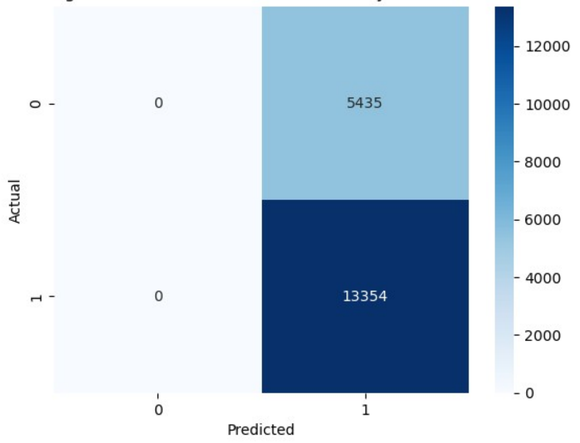
Figure 19: Learning curve for MLP Classifier over "std_pca" dataset



Performance metrics and Confusion Matrix of Dummy Classifier:

```
Dummy Classifier Metrics:
Accuracy: 0.71
Precision: 0.71
Recall: 1.00
F1 Score: 0.83
```

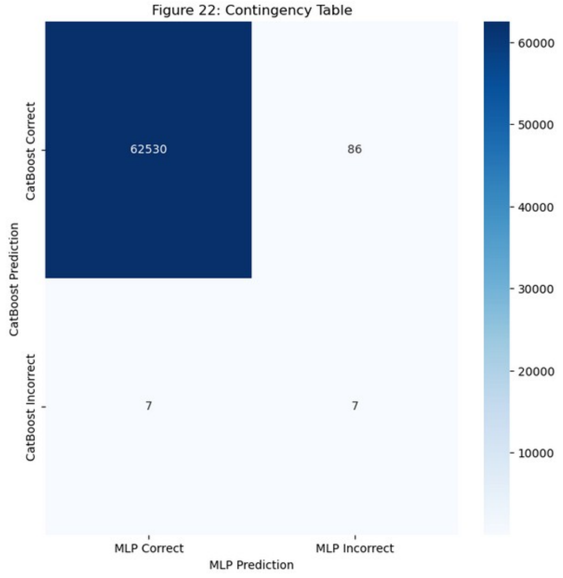
Figure 21: Confusion Matrix of Dummy Classifier



Statistical Significance Test and Contingency Matrix:

```
Statistic: 65.419, p-value: 0.000
Different proportions of errors (reject H0)
```

Figure 22: Contingency Table


CatBoost Correct
CatBoost Incorrect
CatBoost Prediction
MLP Correct
MLP Incorrect
MLP Prediction