UNIVERSITÄT ZU LÜBECK

MDS4AGT
Activity Monitoring
SS25

Scenario 1: Human activity recognition in Python

Sara Farshi, Narges Shafieyoun, Sarah Petersen, Yasser Ibourk

19.05.2025

Content

1. Introduction

Human Activity Recognition (HAR) using wearable sensors has become a key research domain at the intersection of ubiquitous computing, machine learning, and healthcare. By analyzing motion signals collected from devices such as smartphones, smartwatches, and smartglasses, HAR systems aim to automatically detect and classify physical activities performed by individuals. This technology offers promising applications in digital health monitoring, physical rehabilitation, ambient assisted living, and personalized fitness tracking.

In this project, we focus on the classification of 55 behavioral activities based on the *Cognitive Village (CogAge)* dataset, a multi-sensor dataset containing annotated human motion data. Unlike datasets restricted to simple posture detection (e.g., sitting or standing), the CogAge dataset includes a wide range of complex motor activities recorded under controlled conditions using multiple body-worn devices. An overview of these activities is provided in Figure 1.

Our work follows the structure of a classical *Pattern Recognition Chain* (PRC), which includes signal preprocessing, feature extraction, feature selection, and supervised classification. In the preprocessing step, we unify and normalize sensor signals from different devices. We then extract statistical representations from the time series data and apply feature selection methods to reduce redundancy and improve generalization. The resulting feature vectors are used to train and evaluate machine learning classifiers.

For the classification task, we focus on two ensemble learning methods: Random Decision Forest (RDF) and Extreme Gradient Boosting (XGB). Both models are applied in two settings: first using the full set of extracted features, and then using only the subset selected through feature selection. This allows us to assess not only the raw performance of the models but also the effect of dimensionality reduction on classification accuracy and robustness. Evaluation is conducted using accuracy and average F1-score as key metrics.

2. Pattern Recognition Chain (PRC)

For this first Scenario, we follow a structured activity recognition pipeline based on the concept of the Pattern Recognition Chain *(PRC)*. The PRC describes the sequential steps needed to transform raw sensor signals into meaningful predictions. It begins with the preprocessing of multi-modal time series data and continues through handcrafted feature extraction, feature selection, and finally supervised classification. The entire pipeline was implemented in Python, combining established machine learning libraries with custom signal processing functions. Two ensemble methods—Random Decision Forest (RDF) and Extreme Gradient Boosting (XGB)—were used for classification.

2.1. Data Preprocessing

To prepare the data for the further steps, we first loaded and unified the sensor streams from the CogAge dataset. Only three-dimensional arrays with shape (N, T, S)- where N is the number of activity samples, T the number of time steps, and S the number of sensor channels- were processed. This ensured a consistent structure across all sources, including smartphone, smartwatch, and smartglasses. We decided to use all sensor modalities, as we assume that some activities are better captured by specific sensors (e.g., “Close Tap Water” is better detected by the smartwatch).

Each sensor signal was then normalized independently using z-score standardization, based on the global mean and standard deviation computed over all time steps and samples. This normalization was essential to align the differing value ranges of the sensors (e.g., accelerometer vs. gyroscope) and prevent any single modality from dominating the model due to scale alone. Additionally, centering the data around zero and scaling it to unit variance improves numerical stability during training, particularly for models relying on gradient-based optimization.

By removing amplitude-based biases, normalization ensures that the classifier learns from the dynamic patterns of human motion rather than from sensor-specific magnitudes. This preprocessing step was therefore both technically justified and essential to ensure that the subsequent classification models could operate fairly and reliably across multi-modal sensor inputs.

The preprocessed data served as the input for the feature extraction module, which builds a compact representation of each signal segment for classification.

2.2. Feature Extraction and Feature Selection

To transform raw multi-modal sensor data into a format suitable for machine learning, we implemented a structured feature extraction pipeline. For each temporal segment of sensor input—including accelerometer, gyroscope, magnetometer, and related modalities—a total set of 14 statistical and signal-derived features were computed.

Nine of these features were adopted from the original set of features proposed by Li et al. [1], which includes widely used statistical metrics. To complement this set, we selected five additional, more complex features—such as zero-crossing rate, mean absolute change, and complexity-invariant distance (CID)—to enhance the descriptive power of the representation.

All features were extracted per sensor and per channel, then concatenated to form a unified feature vector for each sample. As the sensor modalities are temporally synchronized, the resulting vectors consistently represent individual labeled instances across modalities and are directly usable for classification.

To investigate whether the models efficiency improves by using only a subset of the computed features, feature selection was employed for the top-feature variants of the models. Specifically, Recursive Feature Elimination with Cross-Validation (RFECV) was utilized to identify the most salient features. This process was conducted using both RandomForestClassifier and XGBClassifier as base estimators, depending on the model configuration.

2.3. Classification with Ensemble-Methods

For activity classification, we employed ensemble-based models, leveraging their ability to handle high-dimensional, non-linear, and heterogeneous sensor data. Specifically, we utilized Random Decision Forest (RDF) and Extreme Gradient Boosting (XGB) to train and evaluate classifiers on both full and reduced feature sets.

2.3.1. Random Decision Forest (RDF)

Random Forest is a bagging-based ensemble algorithm that constructs multiple decision trees during training and outputs the mode of their predictions. In our implementation, we used the RandomForestClassifier from scikit-learn with the default values. The model was evaluated using stratified train/validation splits and tested on a hold-out test set. Additionally, when feature selection was applied, the input to the model was transformed using the learned selector pipeline.

2.3.2. Extreme Gradient Boosting (XGB)

Extreme Gradient Boosting (XGBoost) is a boosting-based ensemble method that builds decision trees sequentially, with each new tree aiming to correct the residual errors of its predecessors. In this project, we used the XGBClassifier with the same hyperparameters as for the Random Decision Forest to ensure a fair comparison between the two ensemble methods.

Our choice to include XGBoost was motivated by findings from Hasan et al. [2], where XGBoost—without any extensive hyperparameter tuning—achieved some of the highest performance scores in terms of both accuracy and macro-averaged F1 score.

3. Results

The classification performance of both ensemble models—Random Forest (RDF) and Extreme Gradient Boosting (XGB)—was evaluated under two conditions: using the full feature set (AllFeat) and using a reduced feature subset selected via Recursive Feature Elimination with Cross-Validation (TopFeat). Evaluation metrics include validation accuracy, test accuracy, and macro-averaged F1-score. The detailed results are summarized in Table 1.

Classifier	**Validation Accuracy [%]**	**Test Accuracy [%]**	**Validation F1 Macro [%]**	**Test F1 Macro [%]**
RDF(AllFeat)	77	67	77	66
RDF(TopFeat)	80	63	79	63
XGBoost(AllFeat)	70	62	68	61
XGBoost(TopFeat)	74	61	72	60

Table 1: Evaluation Metrics for Random Forest and XGBoost Models

Random Forest showed stronger and more stable performance across all configurations. The RDF model trained on all features achieved the best test results overall, with a macro-F1 score of 66% and a validation accuracy of 77%. When feature selection was applied (RDF TopFeat), validation accuracy improved to 80%, though test metrics saw a slight drop, indicating improved model efficiency but some loss in generalization. In contrast, XGBoost demonstrated lower performance in both configurations. While feature selection modestly improved its validation metrics (e.g., F1-score rose from 68% to 72%), test accuracy and F1-score remained relatively unchanged or declined, suggesting that XGBoost was more sensitive to the reduction in feature space. This may point to a reliance on the full feature richness to effectively model the complex activity patterns. Previous studies also indicate that ensemble methods like Random Forest are particularly robust in high-dimensional sensor environments, whereas boosting algorithms may require more careful tuning [3, 4]. The confusion matrices (Figures 2–5) provide further insight into class-level performance. Certain activity classes were misclassified more frequently across models, likely due to similarities in motion patterns or insufficient feature differentiation. This highlights the importance of exploring more expressive features or class-specific learning strategies in future work.

4. Conclusion

This study presents a comprehensive approach to human activity recognition (HAR) using multi-modal sensor data from smartphones, smartwatches, and smart glasses. By implementing a structured Pattern Recognition Chain (PRC) in Python—including preprocessing, handcrafted feature extraction, feature selection, and classification—we evaluated two ensemble learning methods: Random Forest (RDF) and Extreme Gradient Boosting (XGB). The results demonstrate that Random Forest consistently outperformed XGBoost across all metrics, particularly in terms of test accuracy and macro-averaged F1-score. This suggests that RDF's bagging strategy is more robust to noise and redundant features often found in wearable sensor data. While feature selection using Recursive Feature Elimination with Cross-Validation (RFECV) slightly improved validation scores, it did not substantially enhance generalization performance on the test set—particularly in the case of XGBoost. This underlines the need for model-specific optimization and possibly a richer or more expressive feature space. From a practical perspective, the proposed system shows promise for deployment in health monitoring, ambient assisted living, and personalized fitness applications. However, some activity classes remain challenging to classify, likely due to overlapping motion patterns or class imbalance. The analysis of confusion matrices highlights the need for more refined modeling strategies or class-specific enhancements. Looking forward, future work could incorporate deep learning architectures—such as convolutional or recurrent neural networks—to automatically learn temporal and spatial features directly from raw sensor signals, as demonstrated in recent studies [5, 6]. Furthermore, addressing class imbalance and exploring online or real-time processing capabilities would make the system more suitable for real-world, on-device deployment. Ultimately, the presented pipeline forms a transparent, modular, and extensible baseline for scalable and interpretable human activity recognition.

5. Structure of the Python Code

The structure of our code, along with some usage instructions, can be found in the README file. For clarity, the following table summarizes the key components of each notebook and its role in the activity recognition pipeline:

Notebook Name	Model Type	Feature Set	Key Components
`RDFAllFeat.ipynb`	Random Forest (RDF)	All features	Preprocessing, feature extraction, model training, evaluation, visualization
`RDFTopFeat.ipynb`	Random Forest (RDF)	Selected features	Same as above, with additional RFECV-based feature selection (precomputed)
`XGBAllFeat.ipynb`	XGBoost (XGB)	All features	Same pipeline structure as RDFAllFeat, adapted to XGB
`XGBTopFeat.ipynb`	XGBoost (XGB)	Selected features	Feature selection using RFECV, model training, performance evaluation

Table 2: Overview of Experiment Notebooks and Their Functional Scope

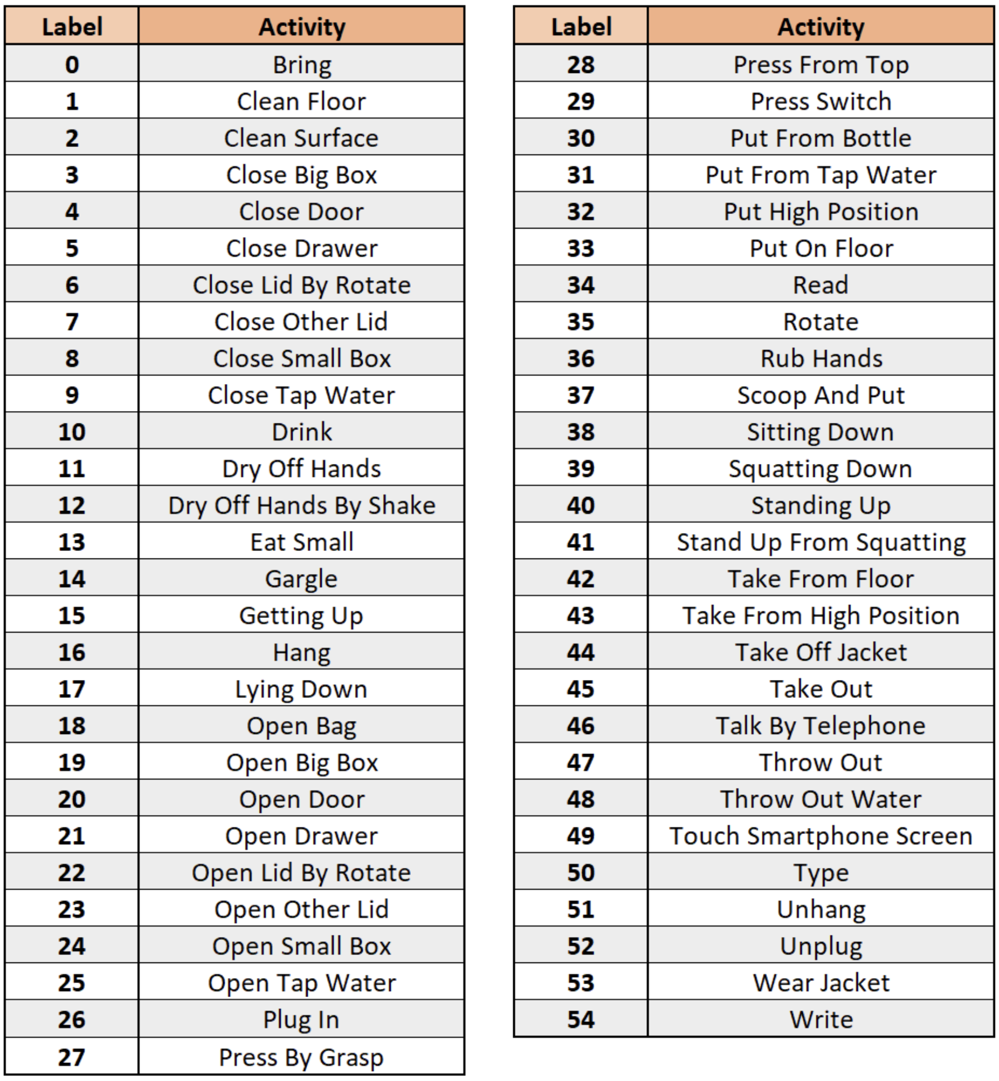
All notebooks share a common structure that includes data preprocessing through normalization of multi-modal sensor signals, extraction of 14 statistical and signal-based features per sensor and channel, and—where applicable—feature selection using RFECV. For the TopFeat variants, the selection step is precomputed and the resulting models are reloaded automatically to ensure reproducibility and efficiency. Each notebook proceeds with model training, evaluation using accuracy and macro-averaged F1-score, and visual analysis via confusion matrices. Commented sections in the code clearly indicate where file paths need to be adapted to the local environment. This unified structure supports flexible experimentation and facilitates isolated testing or modification of individual pipeline components.

6. References

[1] Frederic Li, Kimiaki Shirahama, Muhammad Adeel Nisar, Lukas Köping, and Marcin Grzegorzek. Comparison of Feature Learning Methods for Human Activity Recognition Using Wearable Sensors. *Sensors* 2018. doi: 10.3390/s18020679 URL https://www.mdpi.com/1424-8220/18/2/679

[2] Tasnimul Hasan, Md. Faiyed Bin Karim, Mahin Khan Mahadi, Mirza Muntasir Nishat, Fahim Faisal. Employment of Ensemble Machine Learning Methods for Human Activity Recognition. 2022. doi: 10.1155/2022/6 URL https://onlinelibrary.wiley.com/doi/full/10.1155/2022/6963891

7. Appendix



Label	Activity
0	Bring
1	Clean Floor
2	Clean Surface
3	Close Big Box
4	Close Door
5	Close Drawer
6	Close Lid By Rotate
7	Close Other Lid
8	Close Small Box
9	Close Tap Water
10	Drink
11	Dry Off Hands
12	Dry Off Hands By Shake
13	Eat Small
14	Gargle
15	Getting Up
16	Hang
17	Lying Down
18	Open Bag
19	Open Big Box
20	Open Door
21	Open Drawer
22	Open Lid By Rotate
23	Open Other Lid
24	Open Small Box
25	Open Tap Water
26	Plug In
27	Press By Grasp
28	Press From Top
29	Press Switch
30	Put From Bottle
31	Put From Tap Water
32	Put High Position
33	Put On Floor
34	Read
35	Rotate
36	Rub Hands
37	Scoop And Put
38	Sitting Down
39	Squatting Down
40	Standing Up
41	Stand Up From Squatting
42	Take From Floor
43	Take From High Position
44	Take Off Jacket
45	Take Out
46	Talk By Telephone
47	Throw Out
48	Throw Out Water
49	Touch Smartphone Screen
50	Type
51	Unhang
52	Unplug
53	Wear Jacket
54	Write

Figure 1: 55 activities characterizing the behaviour of an individual

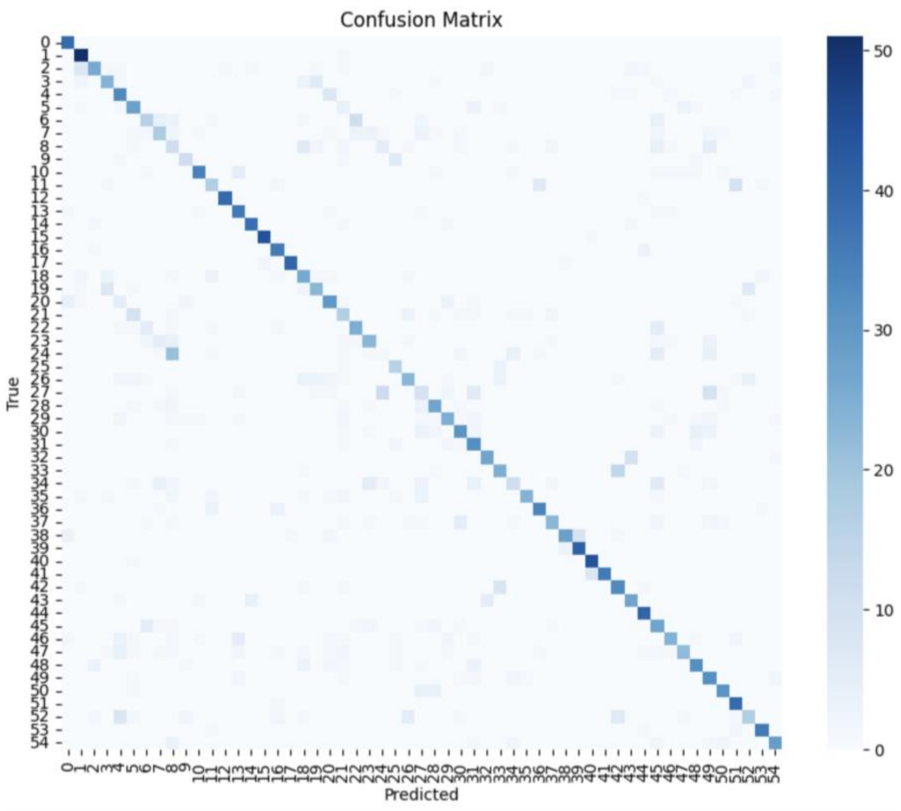


Figure 2: RDF on all Features

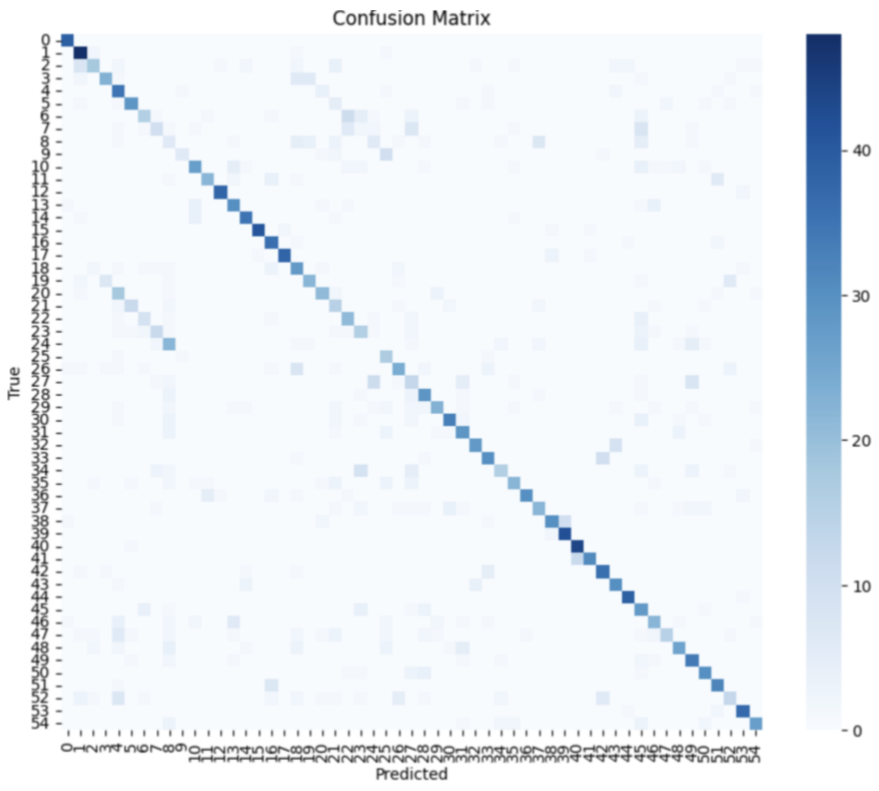


Figure 3: RDF on Top Features

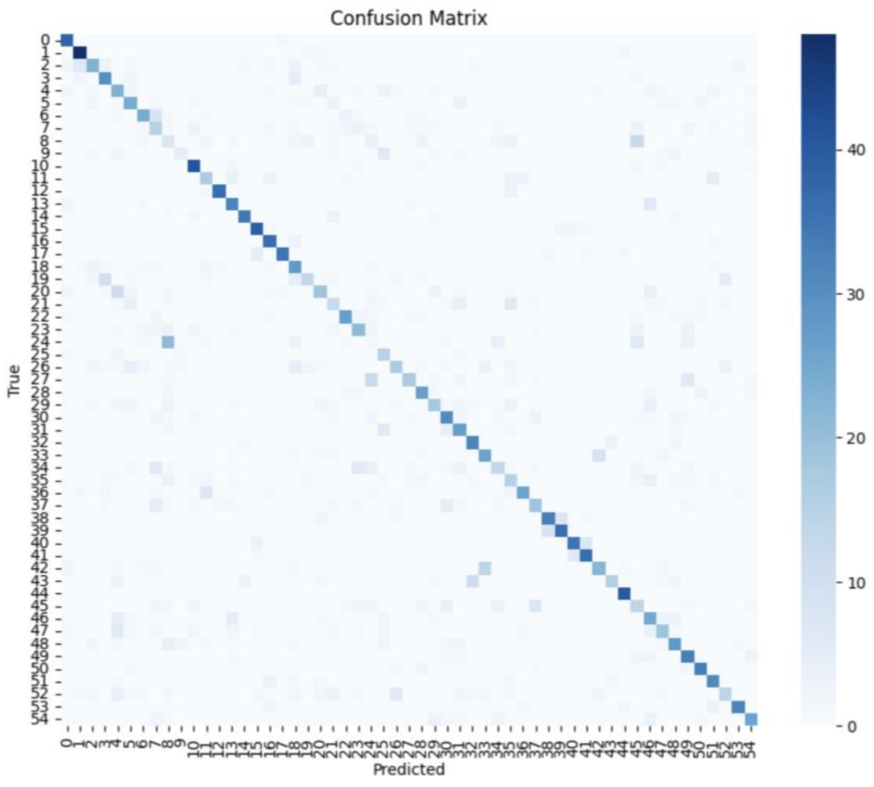


Figure 4: XGB on all Features

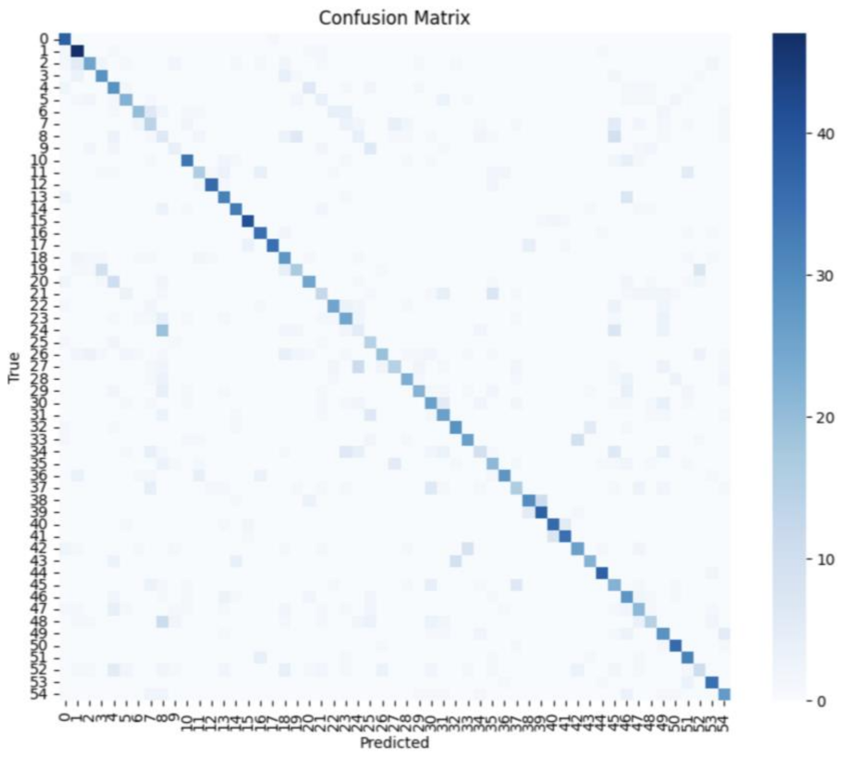


Figure 5: XGB on Top Features