BCI-Based Epileptic Seizure Detection and Warning System

Team 27

Lidia Podoluk, Igor Jakus, Kyrylo Goroshenko

Our task

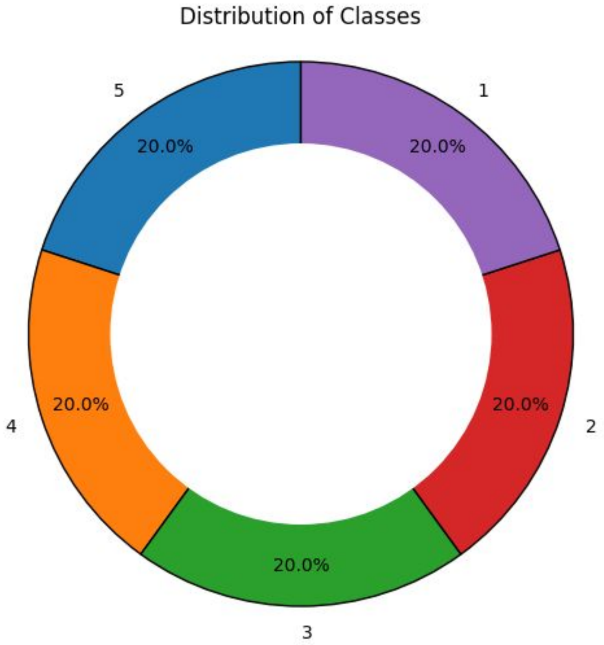
Goal and motivation:

- Provide timely alerts for individuals with epilepsy and their relatives.
- Improve the quality of life and reduce risks associated with seizures.
- Desire to win Neurohackathon

Info about the data:

- EEG dataset from a referenced research paper.
- Data includes labeled examples of pre-seizure and non-seizure brain activity.

Dataset info



Total Samples: 11,500.

Features: 178 per sample (EEG signals).

5 - Eyes open, healthy.

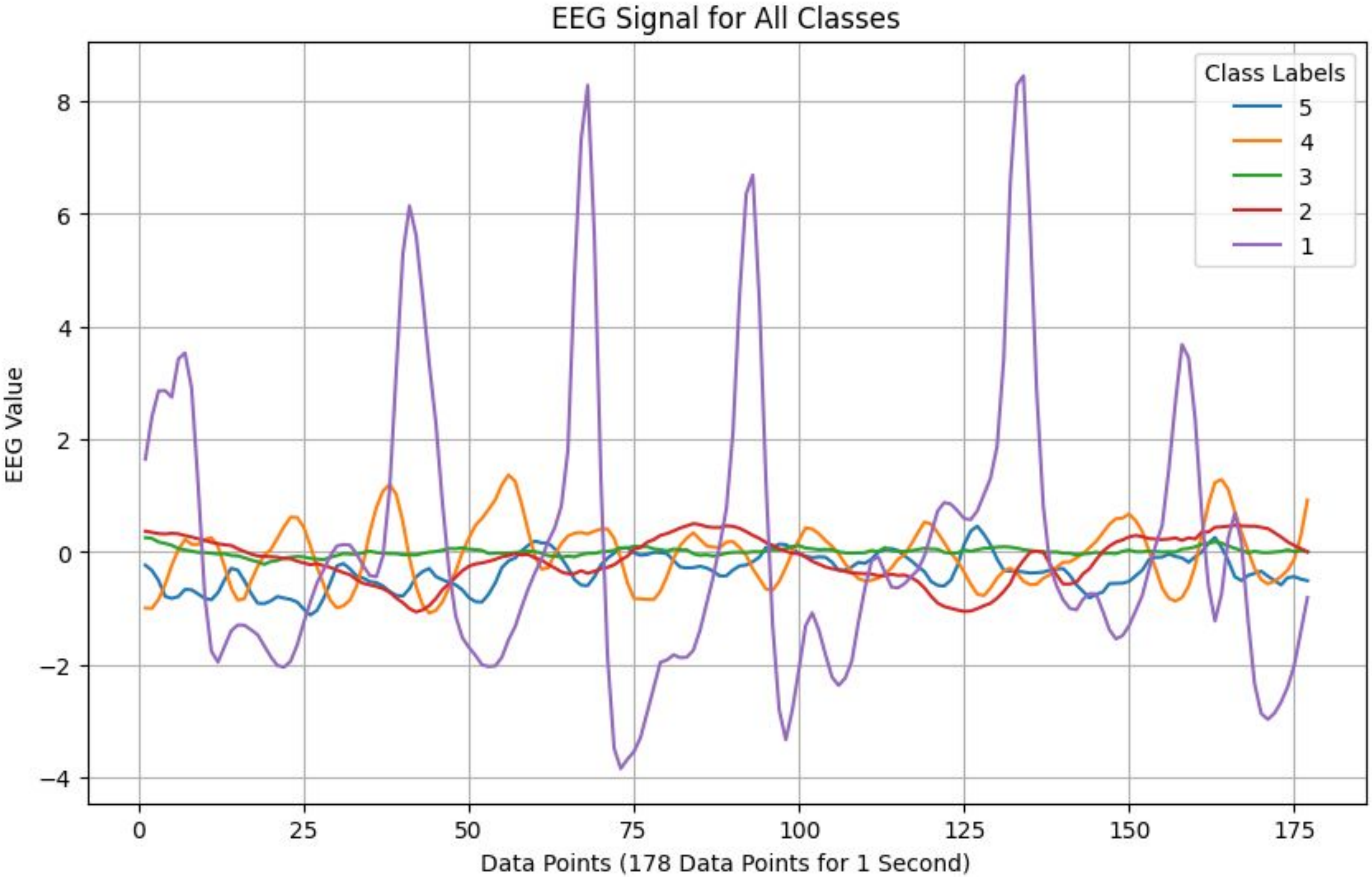
4 - Eyes closed, healthy.

3 - Recording of the EEG activity from the healthy brain area.

2 - Recording of the EEG from the area where the tumor was located

1 - Recording of seizure activity

All subjects falling in classes 2, 3, 4, and 5 are subjects who did not have epileptic seizure. Only subjects in class 1 have epileptic seizure.


EEG Signal for All Classes
EEG Value
Data Points (178 Data Points for 1 Second)
Class Labels
5
4
3
2
1
8
6
4
2
0
−2
−4
0
25
50
75
100
125
150
175

Preprocessing done:

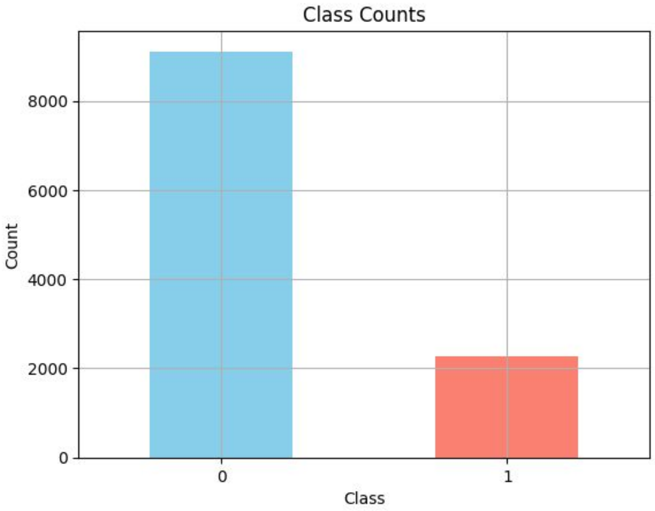
Relabelling to binary:

- 1 = Seizure (only class “1” in original data)
- 0 = No Seizure (every other class)

Data Imbalance: Resolved using SMOTE to oversample minority class (seizures).

Feature Scaling: StandardScaler helped normalize feature contributions, critical for SVM and k-NN.

Data transformation: Some time series models needed some other data shapes.



Methods for binary classification

Used **over 15** Machine Learning algorithms, including:

- Support Vector Machine
- k-Nearest Neighbors
- Logistic Regression
- Decision Trees
- Random Forest
- AdaBoost
- XGBoost
- K-Means (unsupervised)
- Recurrent Neural Network
- Naive Bayes

Model performance

Evaluation metrics:

- Accuracy
- Precision
- Recall
- F1-Score
- Confusion Matrix

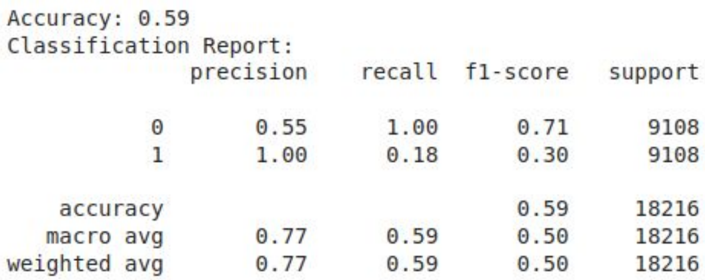
$$Recall = \frac{TP}{TP + FN}$$

Context in Epilepsy Detection

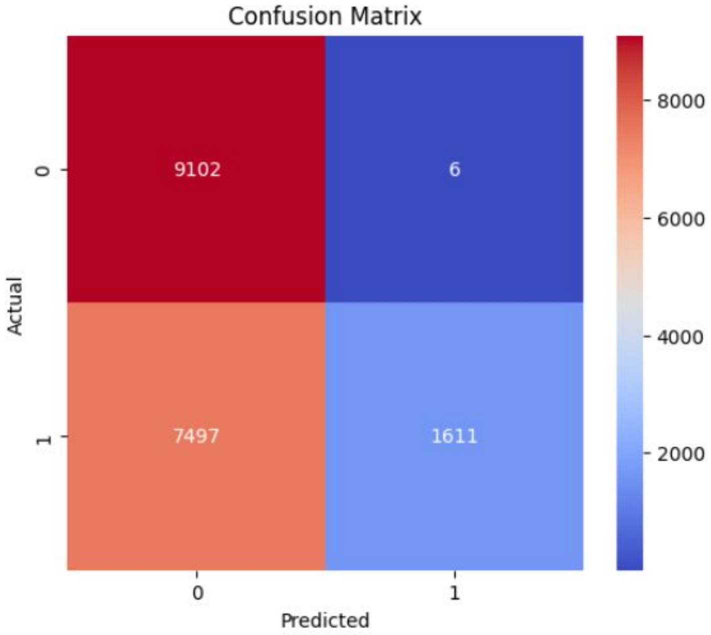
- **High recall is more important than precision** → Missing a seizure (FN) is more dangerous than a false alarm (FP).
- **F1-Score: Balanced precision and recall are ideal** → Too many FPs (low precision) lead to unnecessary anxiety for the patient.
- **Confusion matrix helps visualize both risks** (false negatives vs. false positives).

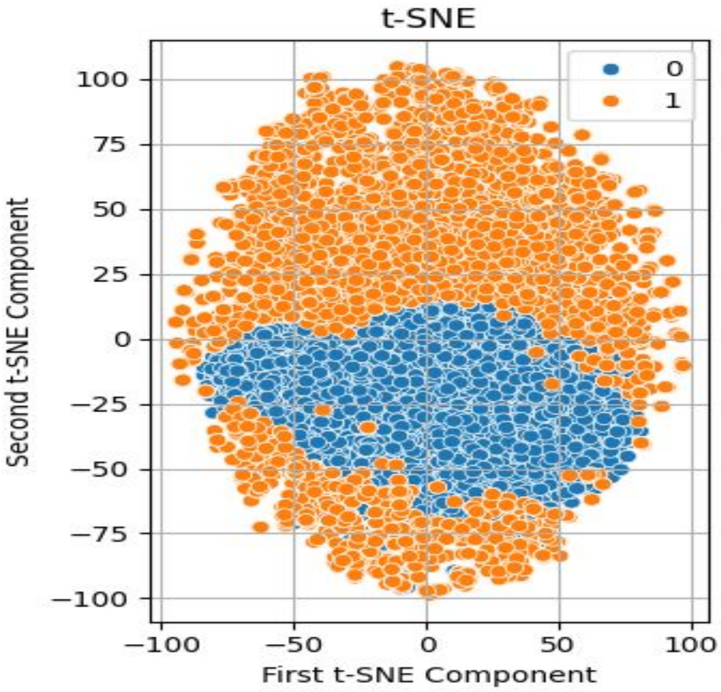
For seizure detection, we prioritize **high recall** while keeping precision reasonable to avoid unnecessary alerts. 🚨

K-Means



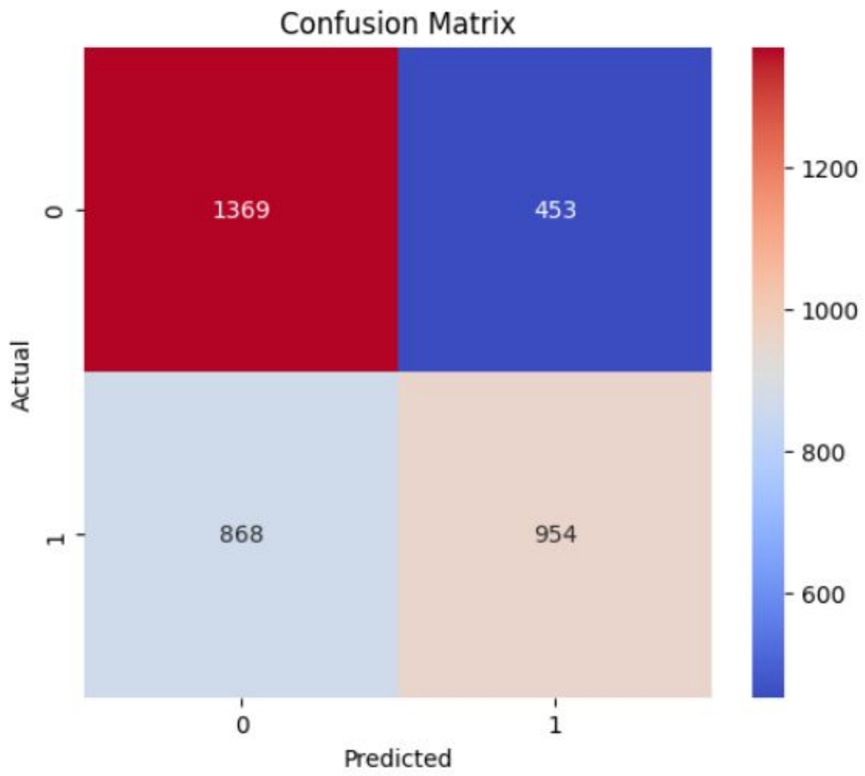
```
Accuracy: 0.59
Classification Report:
              precision    recall  f1-score   support

           0       0.55      1.00      0.71      9108
           1       1.00      0.18      0.30      9108

    accuracy                           0.59     18216
   macro avg       0.77      0.59      0.50     18216
weighted avg       0.77      0.59      0.50     18216
```

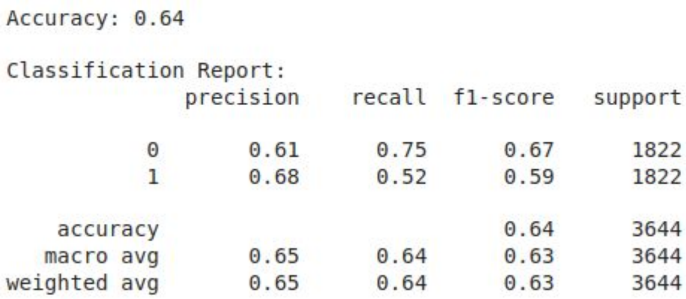




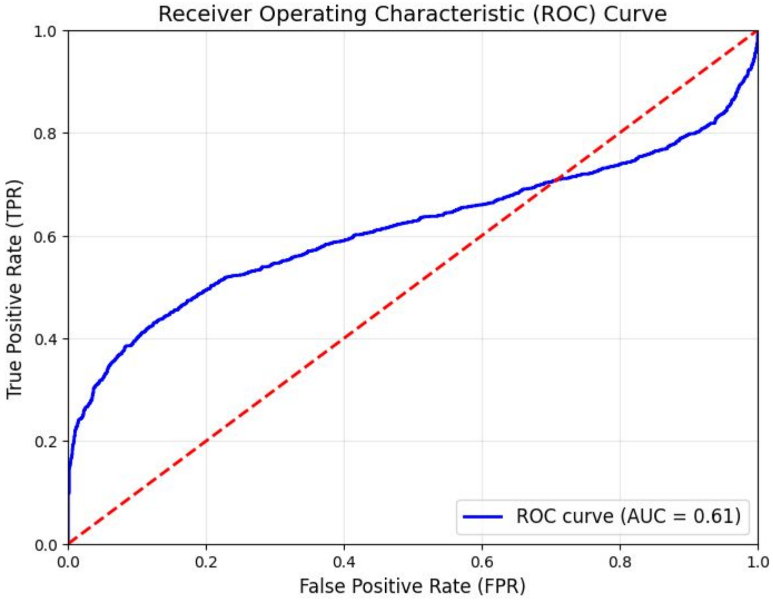
Unsupervised Learning

Logistic Regression

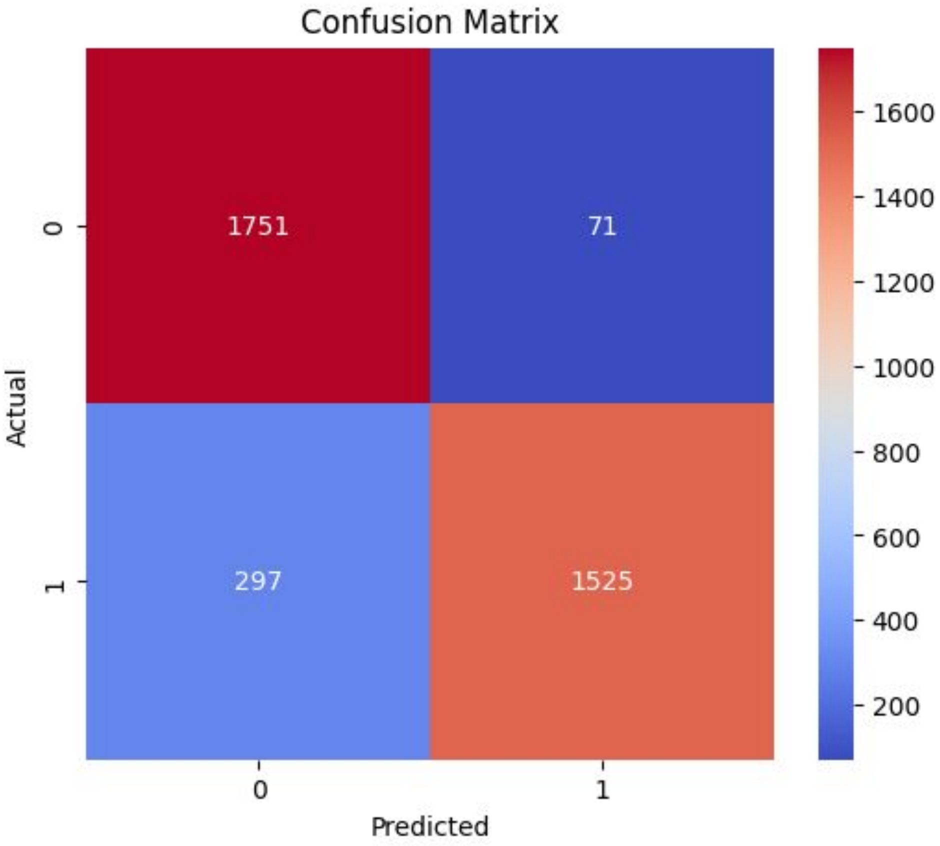




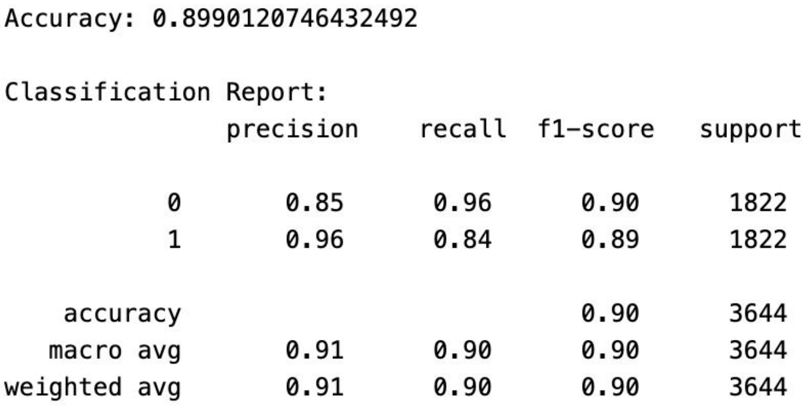
```
Accuracy: 0.64

Classification Report:
              precision    recall  f1-score   support

           0       0.61      0.75      0.67      1822
           1       0.68      0.52      0.59      1822

    accuracy                           0.64      3644
   macro avg       0.65      0.64      0.63      3644
weighted avg       0.65      0.64      0.63      3644
```



Naive Bayes (Gaussian)

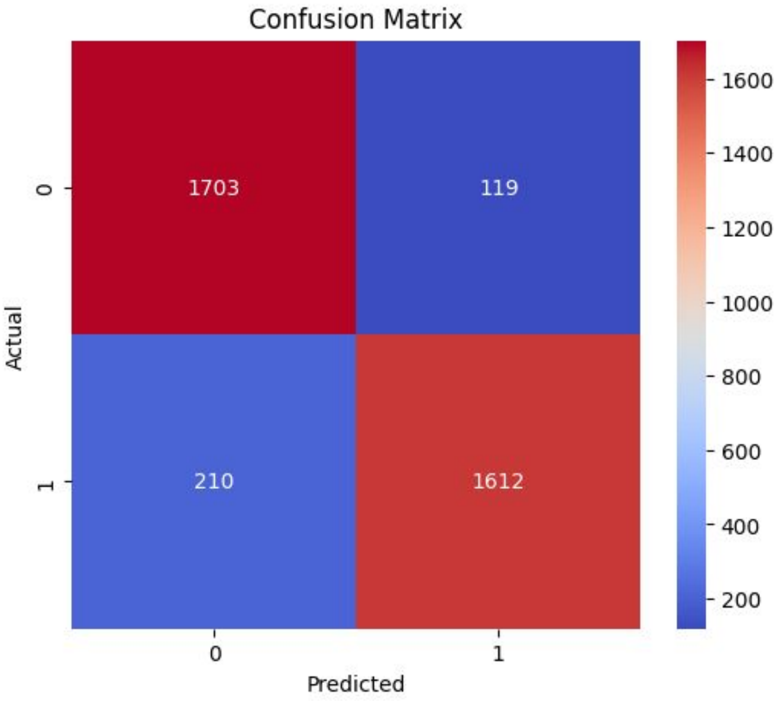


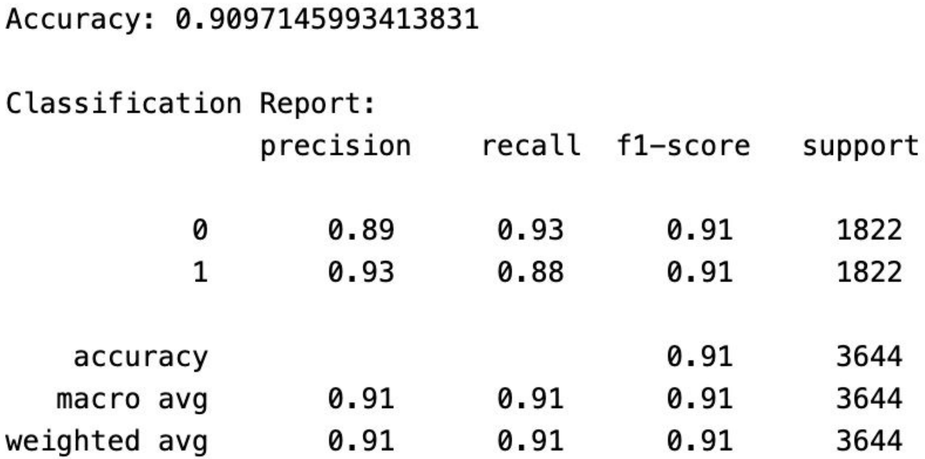
when you start machine learning without rpis



```
Accuracy: 0.8990120746432492

Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.96      0.90      1822
           1       0.96      0.84      0.89      1822

    accuracy                           0.90      3644
   macro avg       0.91      0.90      0.90      3644
weighted avg       0.91      0.90      0.90      3644
```

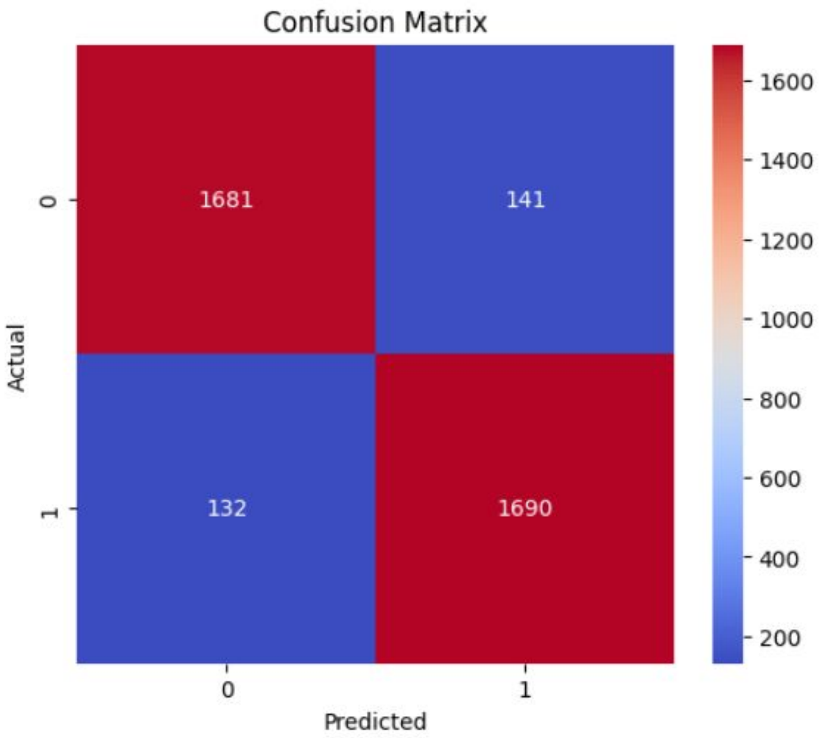
AdaBoost

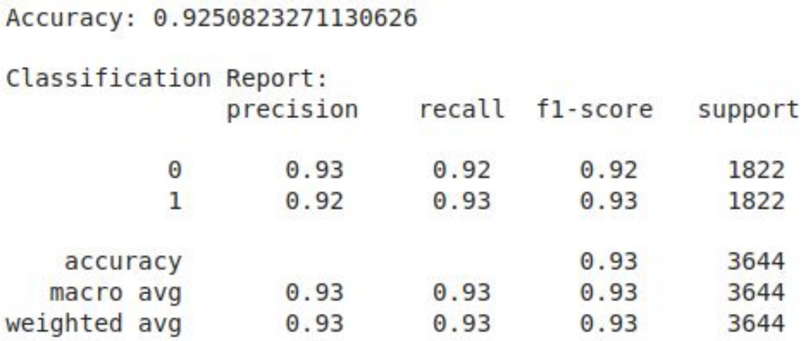




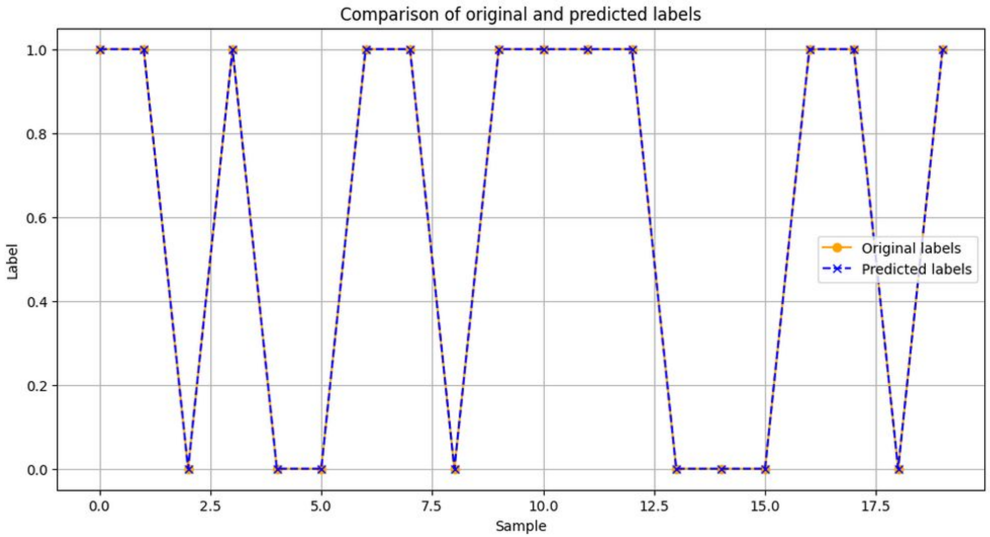
```
Accuracy: 0.9097145993413831

Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.93      0.91      1822
           1       0.93      0.88      0.91      1822

    accuracy                           0.91      3644
   macro avg       0.91      0.91      0.91      3644
weighted avg       0.91      0.91      0.91      3644
```

Decision trees

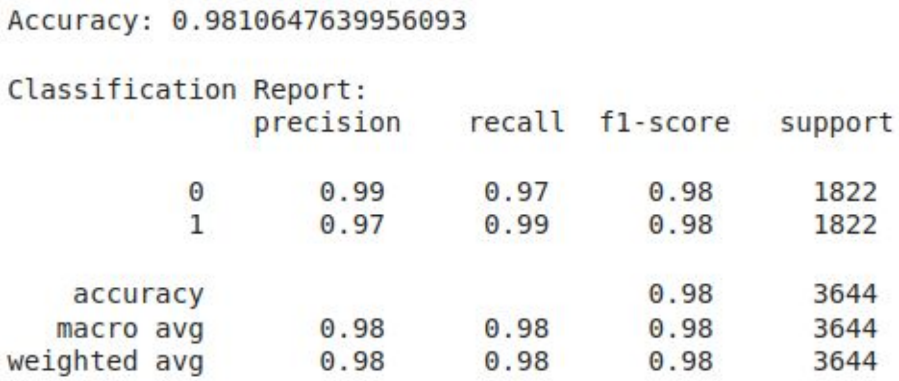




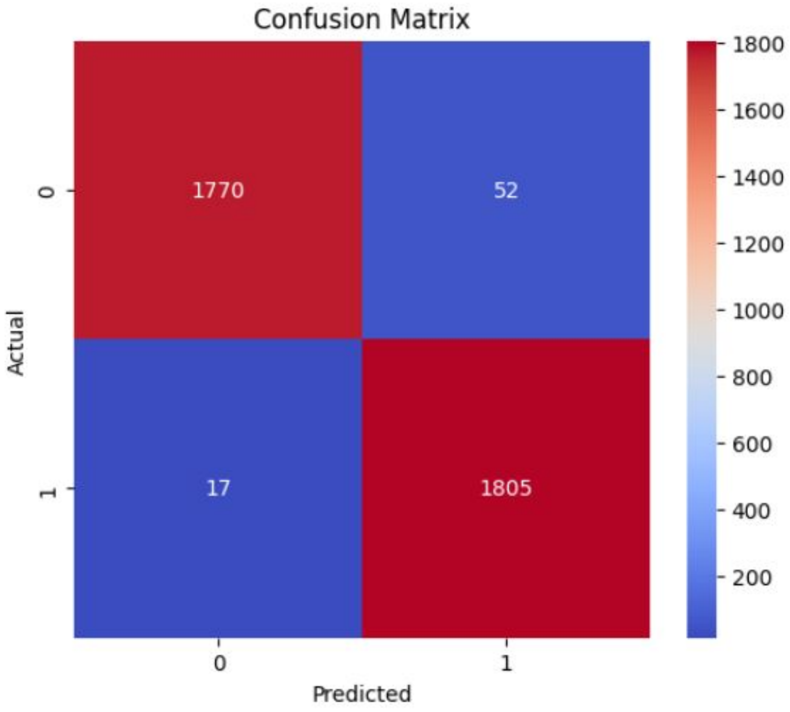
```
Accuracy: 0.9250823271130626

Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.92      0.92      1822
           1       0.92      0.93      0.93      1822

    accuracy                           0.93      3644
   macro avg       0.93      0.93      0.93      3644
weighted avg       0.93      0.93      0.93      3644
```

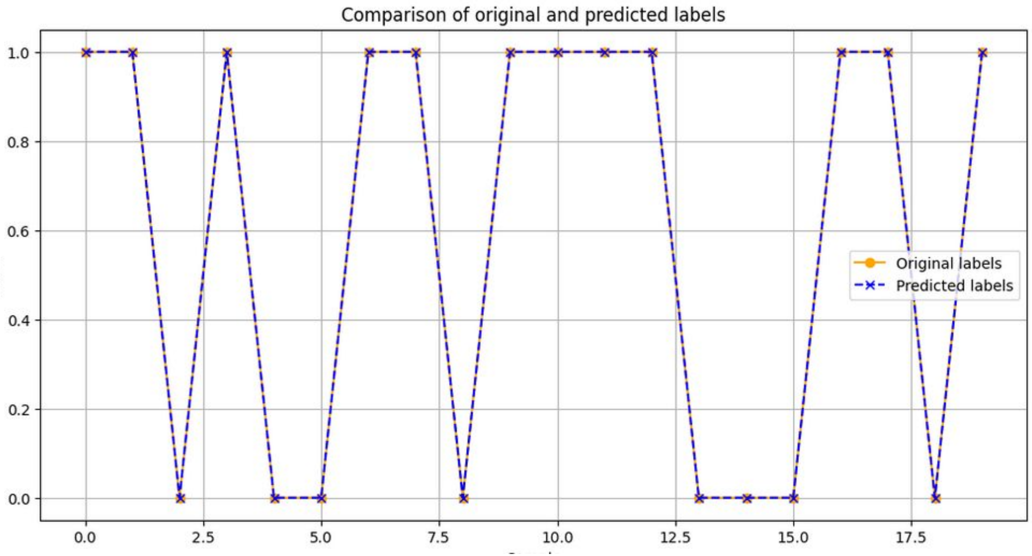


Random Forest

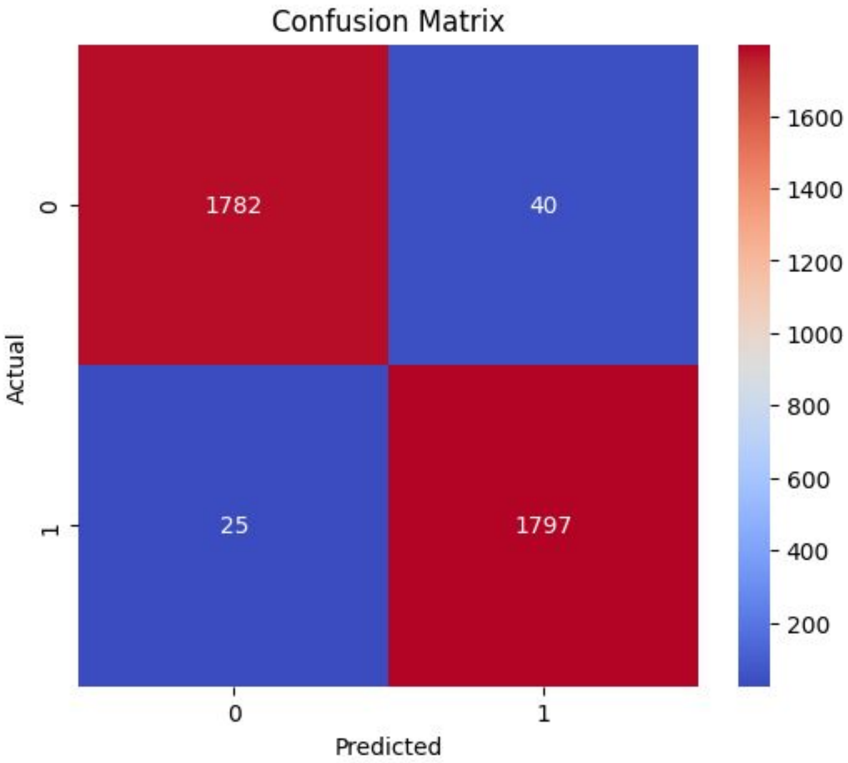


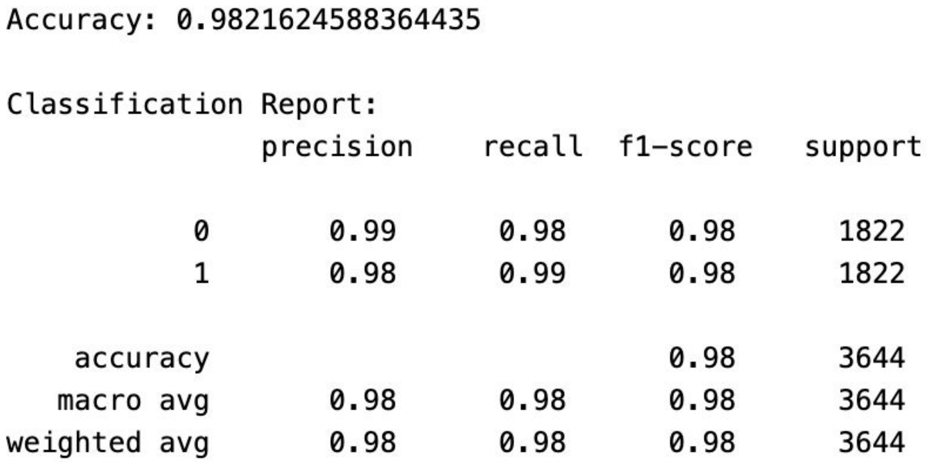
```
Accuracy: 0.9810647639956093

Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.97      0.98      1822
           1       0.97      0.99      0.98      1822

    accuracy                           0.98      3644
   macro avg       0.98      0.98      0.98      3644
weighted avg       0.98      0.98      0.98      3644
```





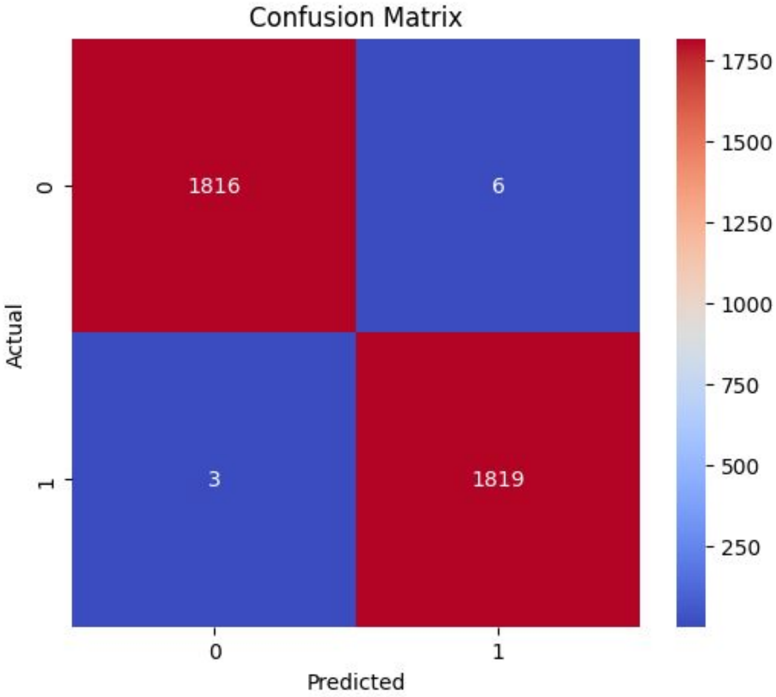
XGBoost

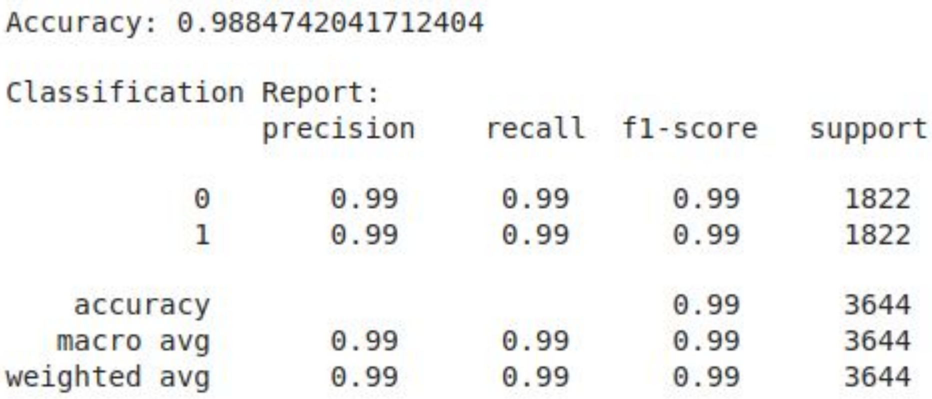




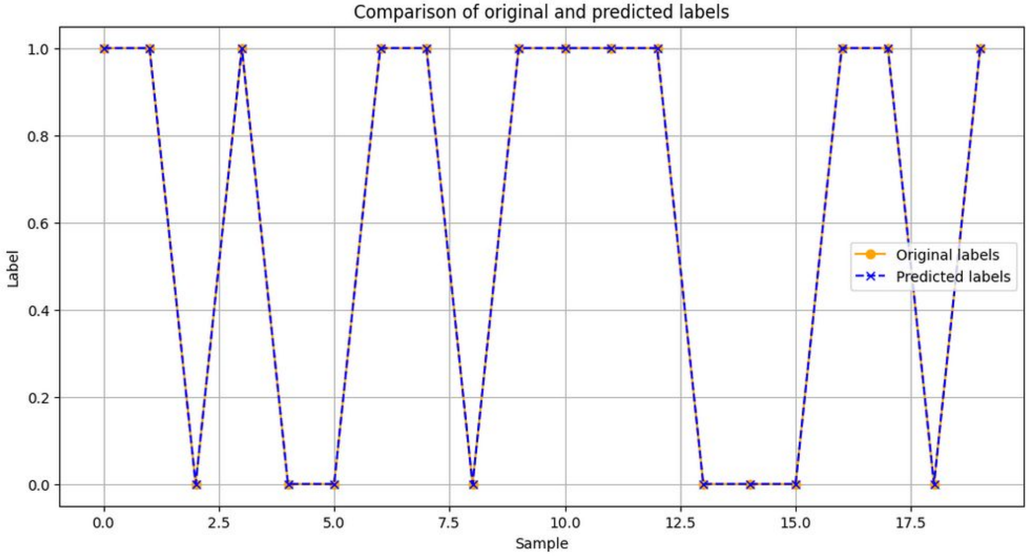
```
Accuracy: 0.9821624588364435

Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.98      0.98      1822
           1       0.98      0.99      0.98      1822

    accuracy                           0.98      3644
   macro avg       0.98      0.98      0.98      3644
weighted avg       0.98      0.98      0.98      3644
```

K-Nearest Neighbors

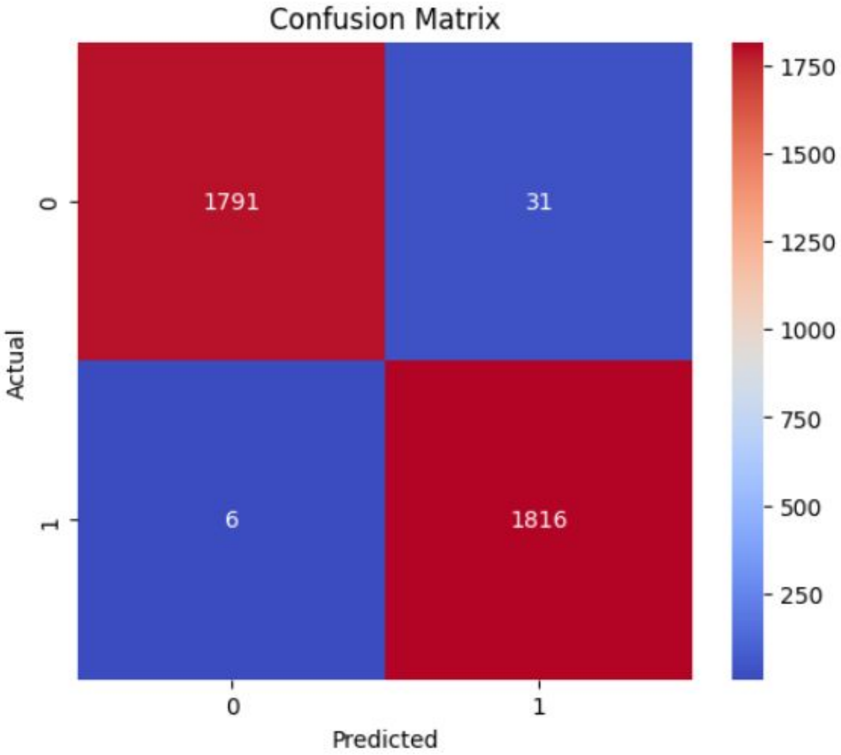


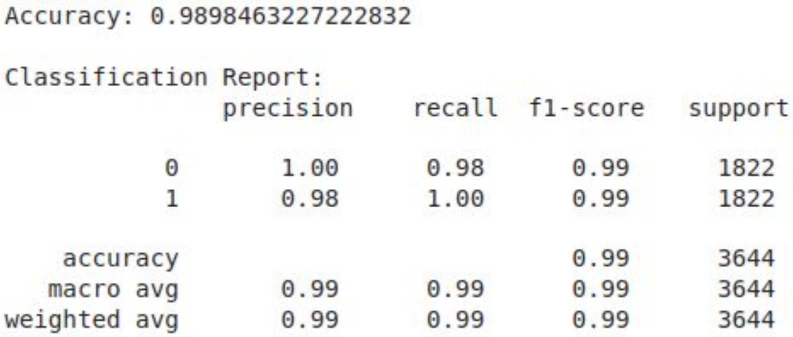


```
Accuracy: 0.9884742041712404

Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      1822
           1       0.99      0.99      0.99      1822

    accuracy                           0.99      3644
   macro avg       0.99      0.99      0.99      3644
weighted avg       0.99      0.99      0.99      3644
```

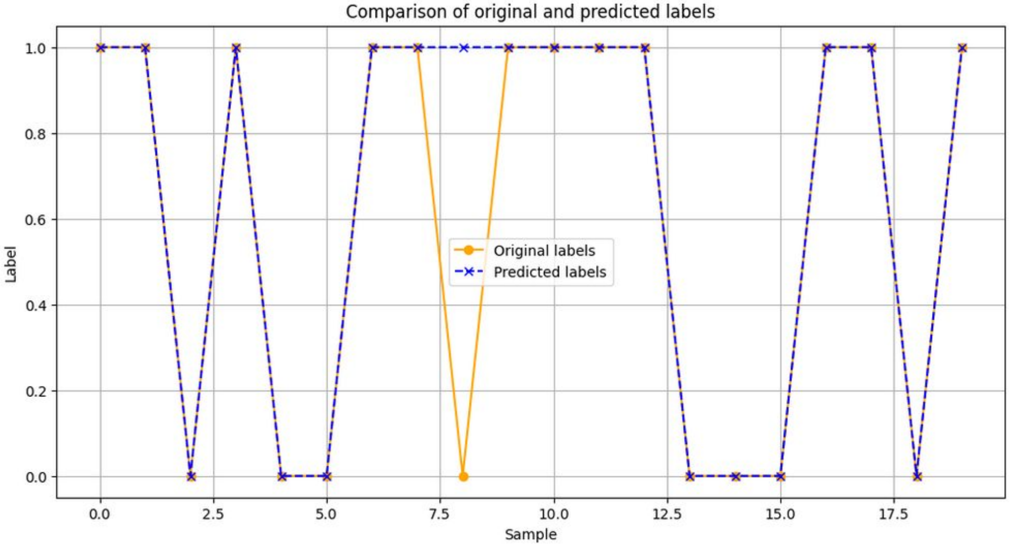


Support Vector Machine





```
Accuracy: 0.9898463227222832

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.98      0.99      1822
           1       0.98      1.00      0.99      1822

    accuracy                           0.99      3644
   macro avg       0.99      0.99      0.99      3644
weighted avg       0.99      0.99      0.99      3644
```

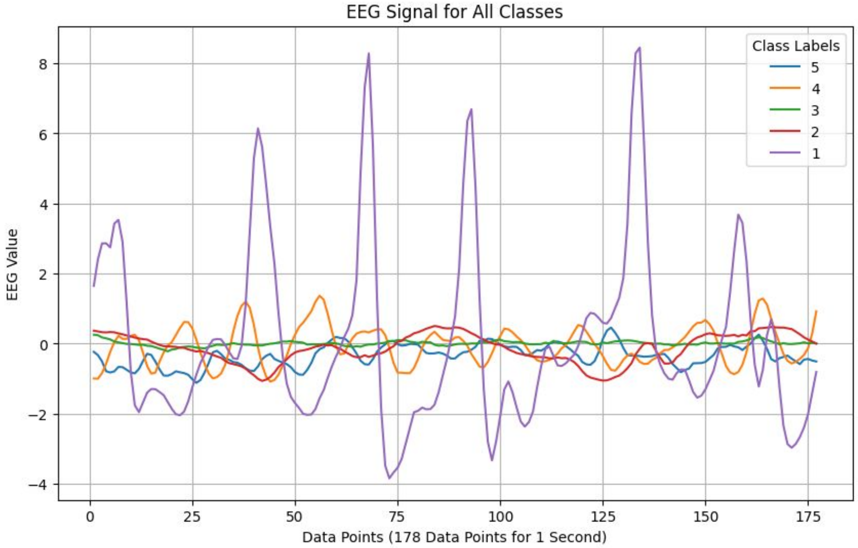


Challenge: division into 5 classes

2-5 pretty similar :(

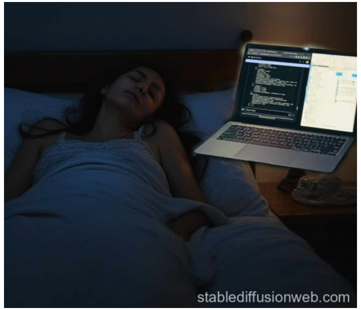
Models (tested with many parameters):

- KNN with DTW
- RNN
- SVM with feature extraction
- Shapelets (2 versions)
- Random Forest with feature extraction
- A LOT OF ensembles



Distance metrics:

- euclidean
- manhattan
- cross-correlation
- dtw -> veeery slow ->



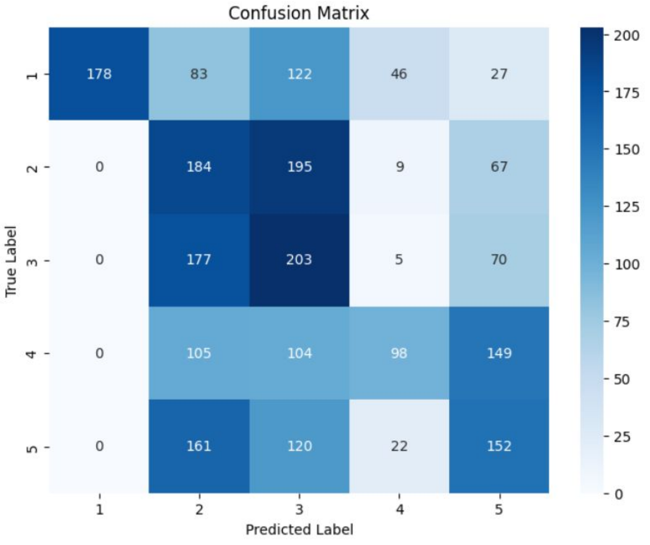
Scalers:

- StandardScaler
- TimeSeriesScalerMinMax

Feature extraction:

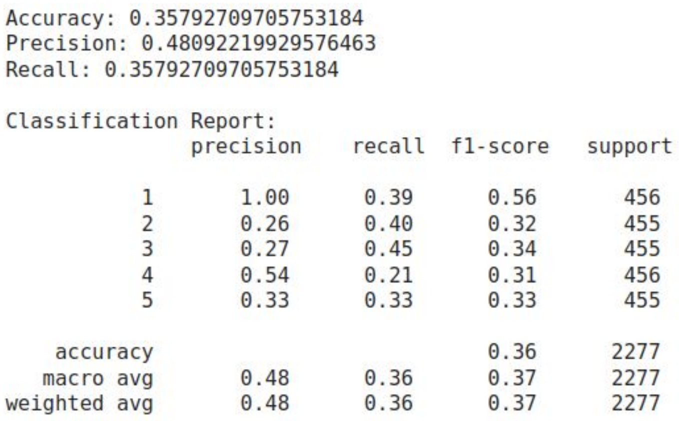
- custom with statistic features
- from TSFresh library

Shapelets



Confusion Matrix

True Label \ Predicted Label	1	2	3	4	5
1	178	83	122	46	27
2	0	184	195	9	67
3	0	177	203	5	70
4	0	105	104	98	149
5	0	161	120	22	152

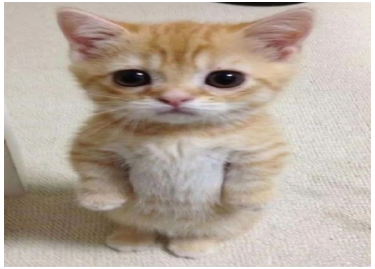


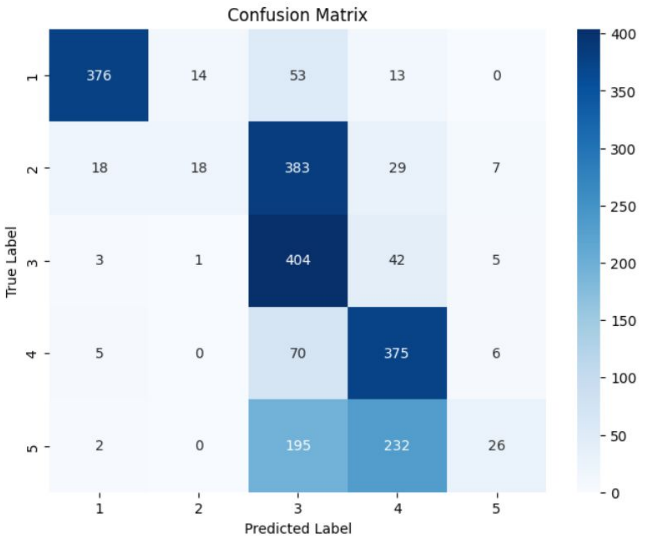
```
Accuracy: 0.35792709705753184
Precision: 0.48092219929576463
Recall: 0.35792709705753184

Classification Report:
              precision    recall  f1-score   support

           1       1.00      0.39      0.56       456
           2       0.26      0.40      0.32       455
           3       0.27      0.45      0.34       455
           4       0.54      0.21      0.31       456
           5       0.33      0.33      0.33       455

    accuracy                           0.36      2277
   macro avg       0.48      0.36      0.37      2277
weighted avg       0.48      0.36      0.37      2277
```

<- Custom

Library ->

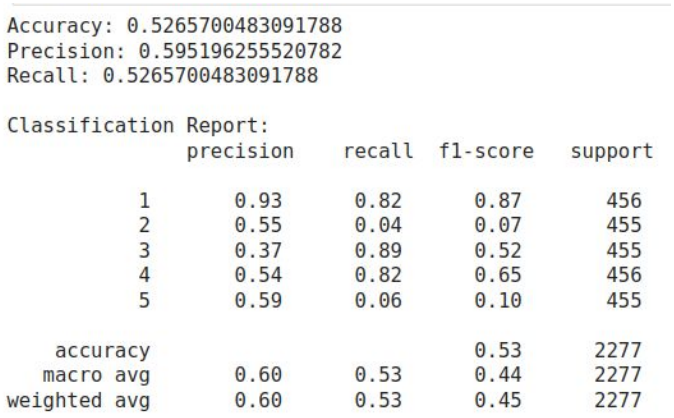
We tried different lengths, numbers of Shapelets, different metrics, etc, unfortunately they didn’t improve the results at all





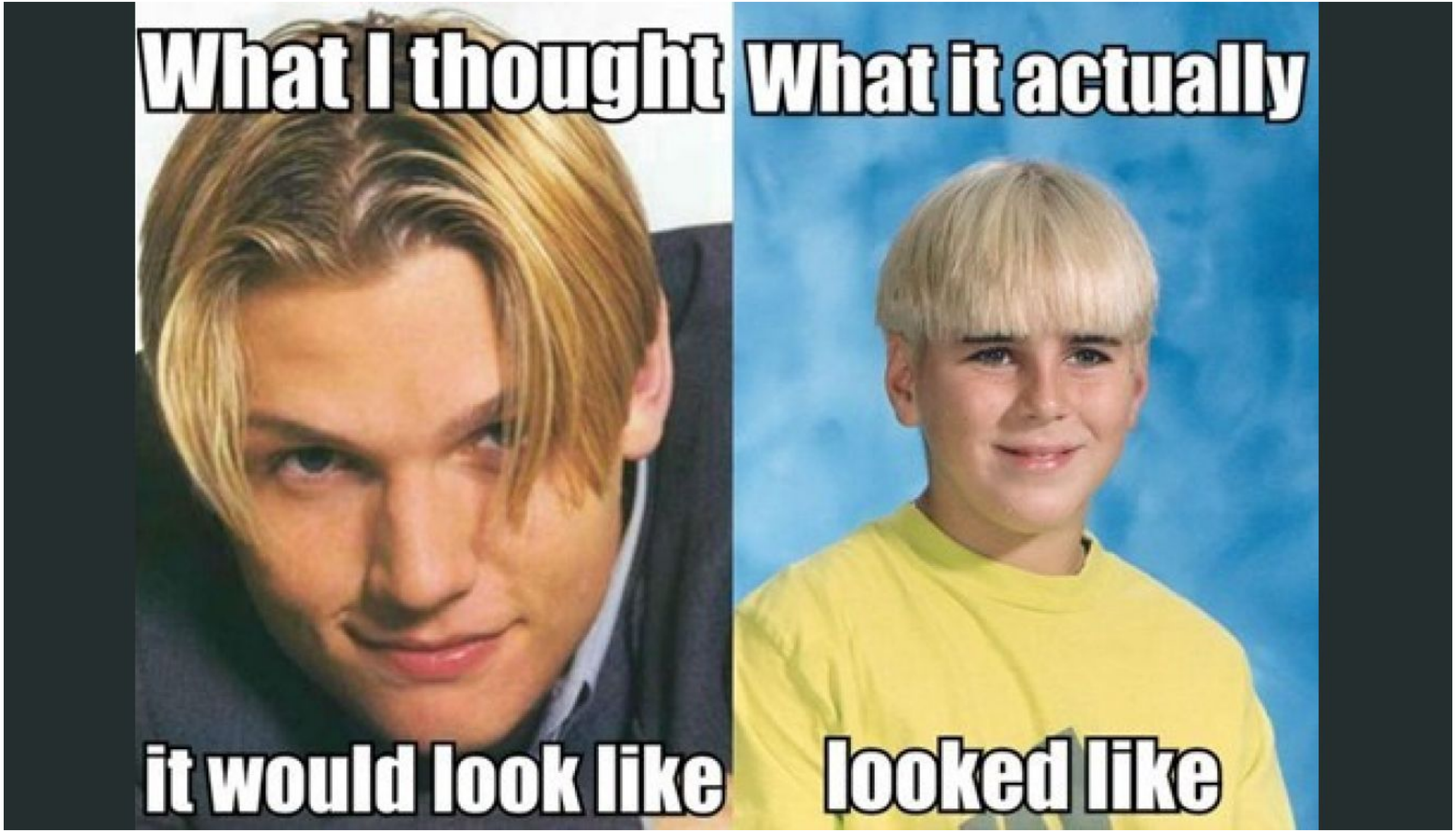
Confusion Matrix

True Label \ Predicted Label	1	2	3	4	5
1	376	14	53	13	0
2	18	18	383	29	7
3	3	1	404	42	5
4	5	0	70	375	6
5	2	0	195	232	26

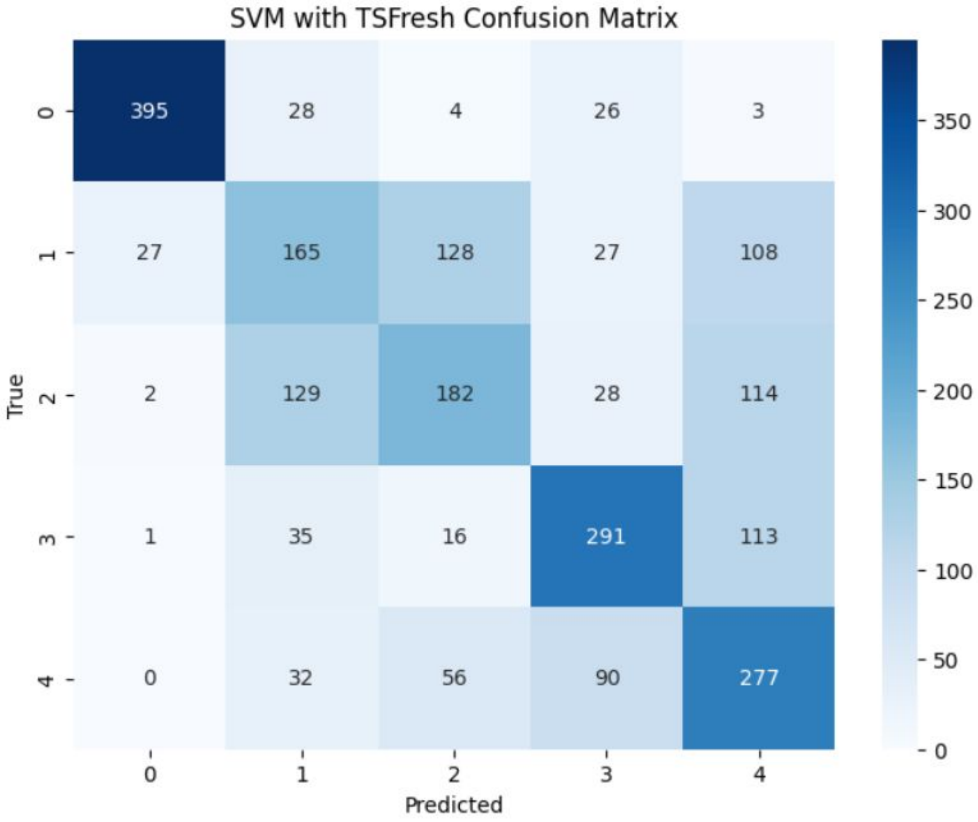


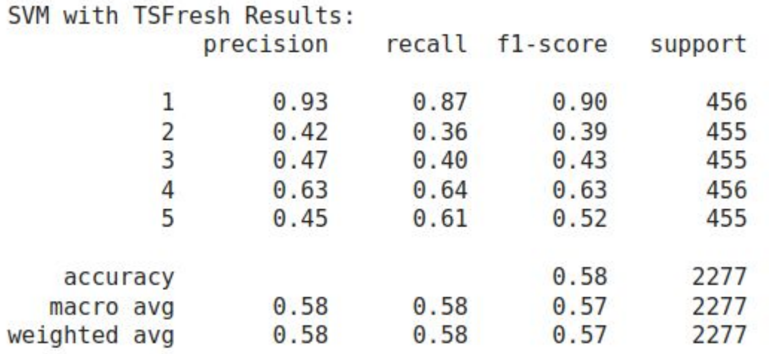
```
Accuracy: 0.5265700483091788
Precision: 0.595196255520782
Recall: 0.5265700483091788

Classification Report:
              precision    recall  f1-score   support

           1       0.93      0.82      0.87       456
           2       0.55      0.04      0.07       455
           3       0.37      0.89      0.52       455
           4       0.54      0.82      0.65       456
           5       0.59      0.06      0.10       455

    accuracy                           0.53      2277
   macro avg       0.60      0.53      0.44      2277
weighted avg       0.60      0.53      0.45      2277
```

Our experience with shapelets…



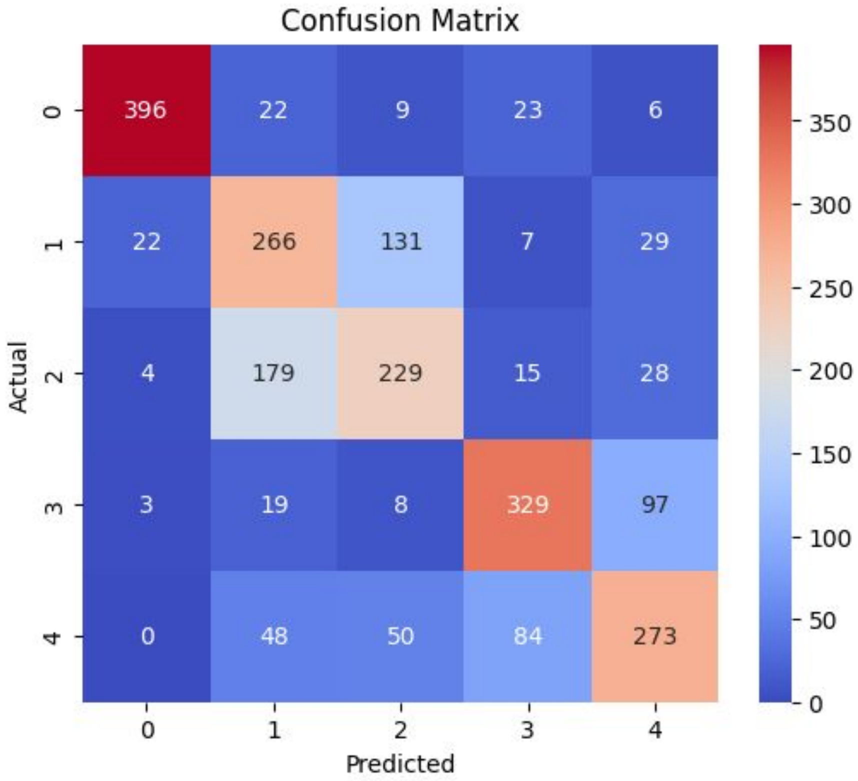
SVM with feature extraction from TSFresh

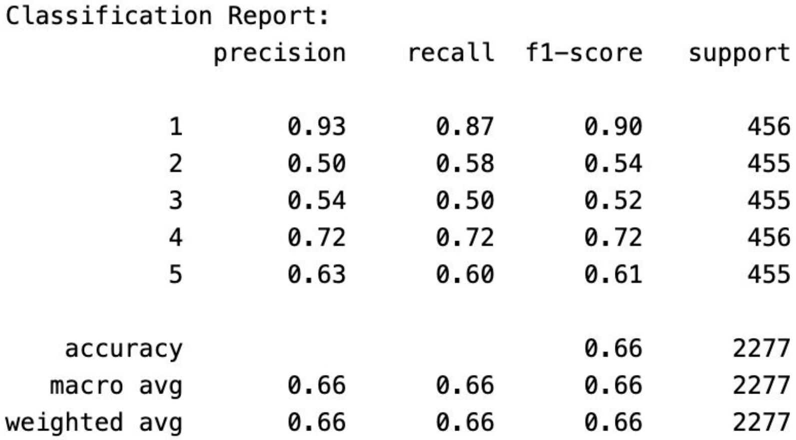




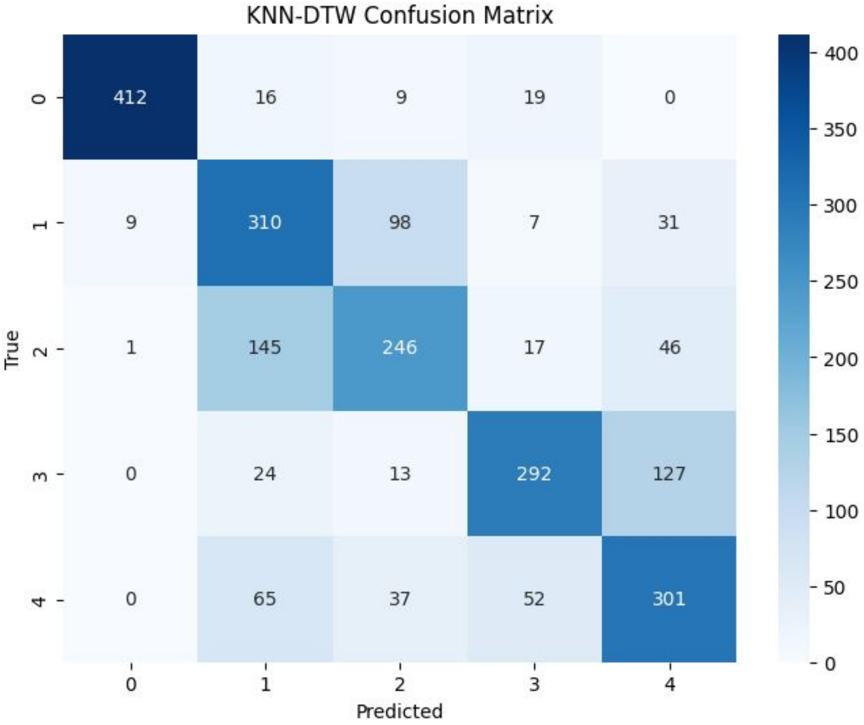
```
SVM with TSFresh Results:
              precision    recall  f1-score   support

           1       0.93      0.87      0.90       456
           2       0.42      0.36      0.39       455
           3       0.47      0.40      0.43       455
           4       0.63      0.64      0.63       456
           5       0.45      0.61      0.52       455

    accuracy                           0.58      2277
   macro avg       0.58      0.58      0.57      2277
weighted avg       0.58      0.58      0.57      2277
```

Recurrent Neural Network




```
Classification Report:
              precision    recall  f1-score   support

           1       0.93      0.87      0.90       456
           2       0.50      0.58      0.54       455
           3       0.54      0.50      0.52       455
           4       0.72      0.72      0.72       456
           5       0.63      0.60      0.61       455

    accuracy                           0.66      2277
   macro avg       0.66      0.66      0.66      2277
weighted avg       0.66      0.66      0.66      2277
```

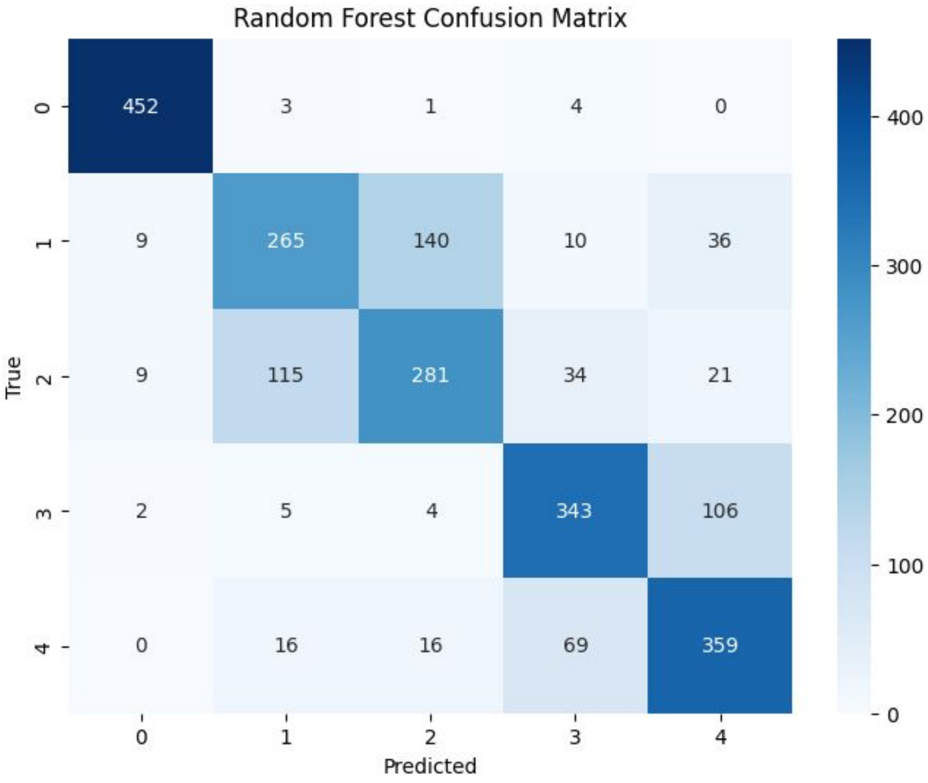
KNN with Dynamic Time Warping

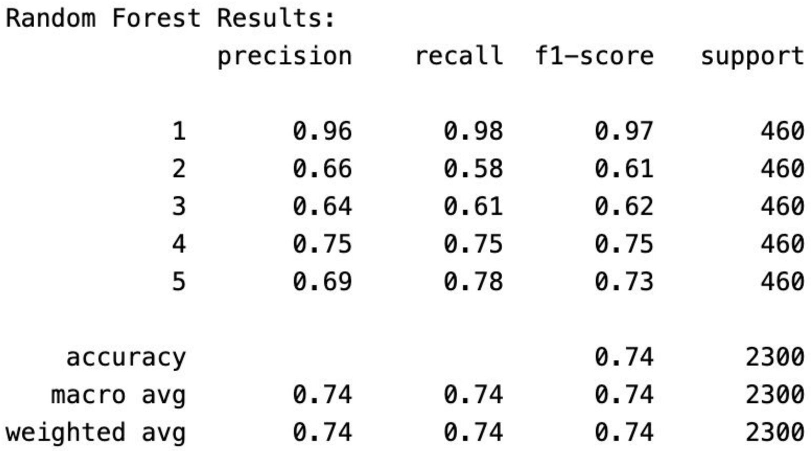

KNN-DTW Confusion Matrix

True \ Predicted	0	1	2	3	4
0	412	16	9	19	0
1	9	310	98	7	31
2	1	145	246	17	46
3	0	24	13	292	127
4	0	65	37	52	301

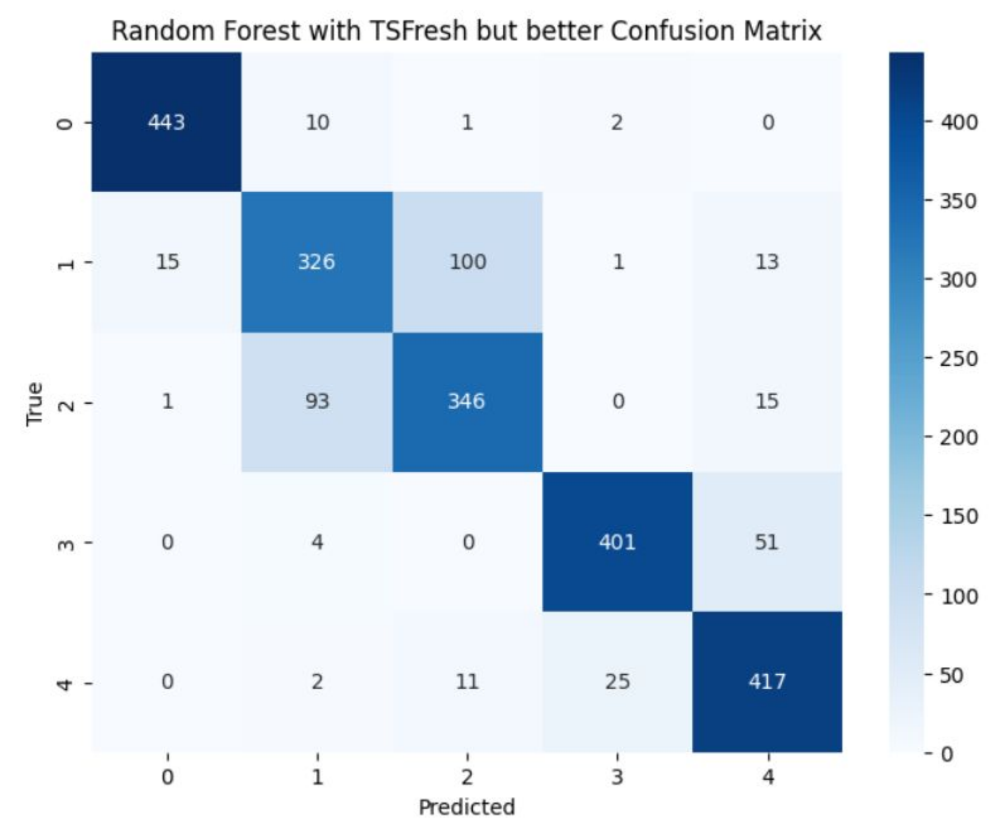
```
              precision    recall  f1-score   support

           1       0.98      0.90      0.94       456
           2       0.55      0.68      0.61       455
           3       0.61      0.54      0.57       455
           4       0.75      0.64      0.69       456
           5       0.60      0.66      0.63       455

    accuracy                           0.69      2277
   macro avg       0.70      0.69      0.69      2277
weighted avg       0.70      0.69      0.69      2277
```

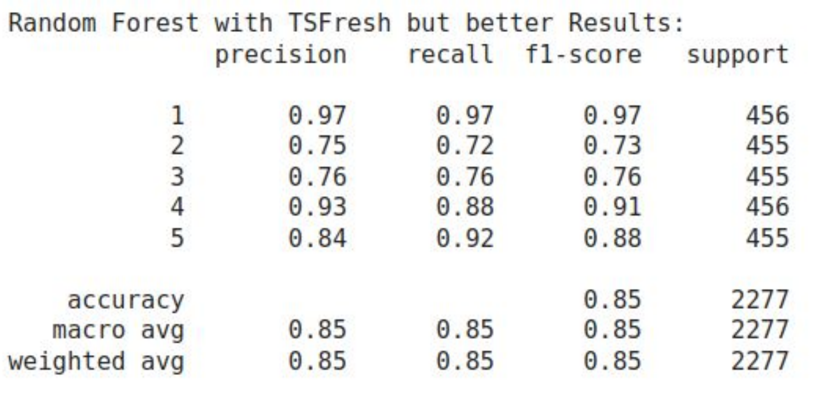
Random Forest with custom feature extraction




```
Random Forest Results:
              precision    recall  f1-score   support

           1       0.96      0.98      0.97       460
           2       0.66      0.58      0.61       460
           3       0.64      0.61      0.62       460
           4       0.75      0.75      0.75       460
           5       0.69      0.78      0.73       460

    accuracy                           0.74      2300
   macro avg       0.74      0.74      0.74      2300
weighted avg       0.74      0.74      0.74      2300
```

Random Forest with feature extraction from TSFresh

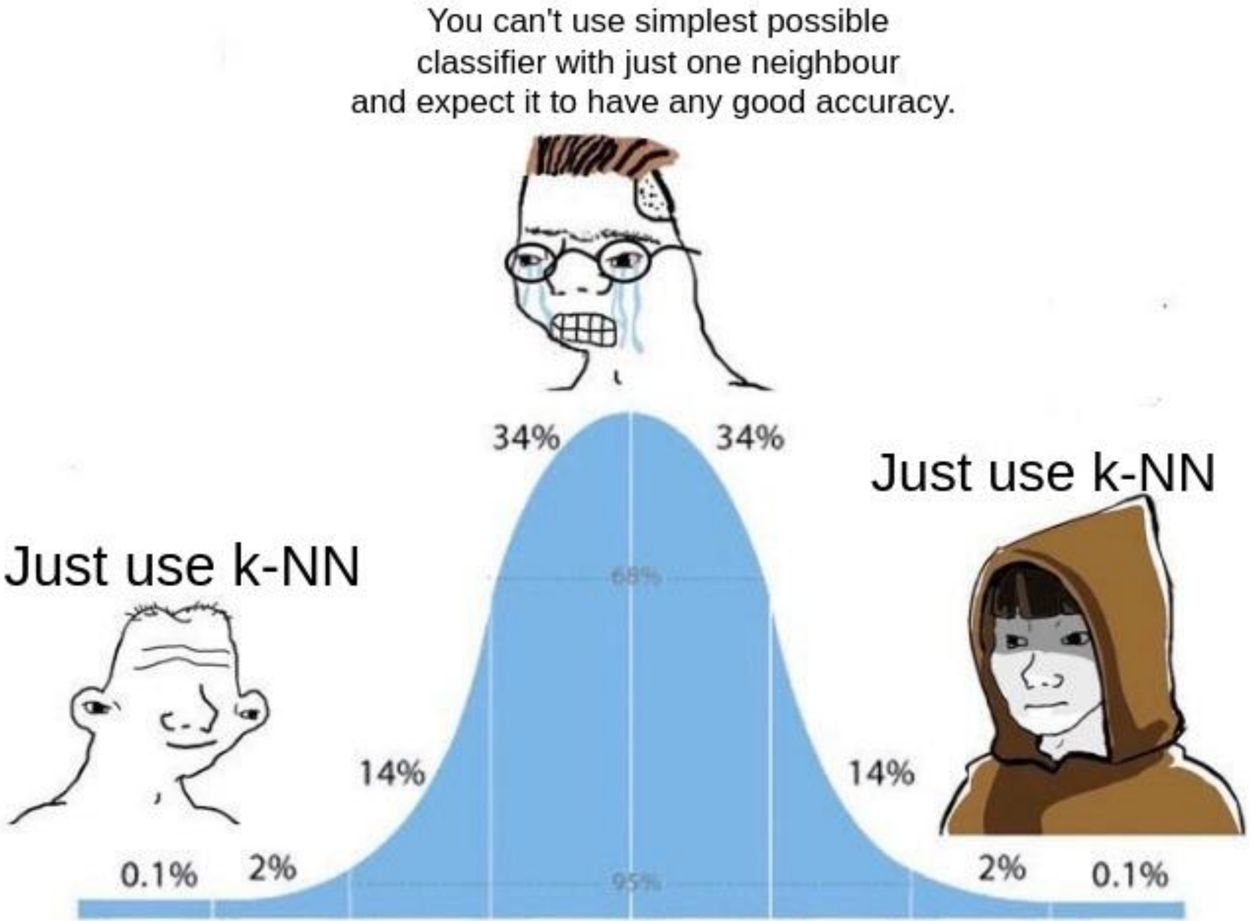




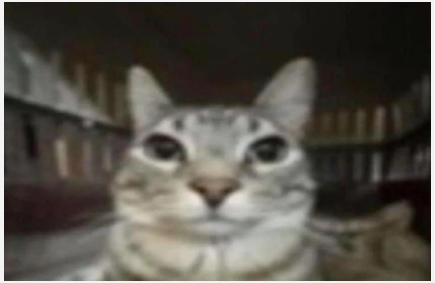
```
Random Forest with TSFresh but better Results:
              precision    recall  f1-score   support

           1       0.97      0.97      0.97       456
           2       0.75      0.72      0.73       455
           3       0.76      0.76      0.76       455
           4       0.93      0.88      0.91       456
           5       0.84      0.92      0.88       455

    accuracy                           0.85      2277
   macro avg       0.85      0.85      0.85      2277
weighted avg       0.85      0.85      0.85      2277
```

Most important in context of seizure risk detection: **not classify 1,2,3 as 4,5**

So effectively it’s much better than 85% accuracy shows


You can't use simplest possible classifier with just one neighbour and expect it to have any good accuracy.
Just use k-NN
Just use k-NN
34%
34%
68%
14%
14%
0.1%
2%
95%
2%
0.1%

```python
# Train KNN model
knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(X_train, y_train)

# Predict and evaluate
y_pred = knn.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```



```
Accuracy: 0.997530186608123

Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00      1822
           1       1.00      1.00      1.00      1822

    accuracy                           1.00      3644
   macro avg       1.00      1.00      1.00      3644
weighted avg       1.00      1.00      1.00      3644
```

Conclusions

- Implemented multiple machine learning models to detect seizures.

- Achieved **state-of-the-art** performance with SVM and 1-NN

- Successfully experimented with classifying into 5 classes using advanced time series techniques.

- Achieved almost* **state-of-the-art** performance **for 5 classes** division with Random Forest with feature extraction

- **Beat all Kaggle competitors :-)**

binary:
99,02 world best
we 99.753%

5 classes:
world: 0%
we 85%

Thank you for your attention !

github.com/igorjakus/neurohackathon

Politech
Wrocław
EPILEPTIC
SYSTEM