# Università degli studi di messina
# MIFT Department


*Data mining*

*Prediction of Emotional States*

**Irmuunbilig Burenzevseg**

Academic year 2023/2024

# INTRODUCTION

This report presents the analysis and modelling of a dataset containing the physiological responses of 186 healthy subjects. The dataset, "Dataset_Study2", includes features extracted from ECG signals and aims to predict emotional states based on these physiological responses. The main goal of this project is to demonstrate the application of various data mining techniques to preprocess, analyse, and model the dataset for accurate classification.

## Dataset Description

The dataset contains 558 observations and 20 columns, including 19 features and 1 target variable (Label). The features are extracted from the RR interval of ECG signals, and the labels represent different emotional states:

- Baseline condition = -1
- Threat = 209
- Anger = 208
- Low-approach positive emotions = 308
- High-approach positive emotions = 309
- Neutral State = 108

The extracted features include time-domain, frequency-domain, and non-linear features.

# Methodology

## 1) Data Exploration and Pre-Processing

### 1.1 Data Loading and Initial Inspection

- The dataset was loaded into a Pandas DataFrame
- Summary statistics and information about the dataset were generated to identify data types and check for missing values.
- Initially there were no missing values

### 1.2 Outlier Detection and Replacement

- Boxplots were used to visualise potential outliers.

- Outliers were replaced with median values to ensure robust analysis.

### 1.3 Addressing  Zero Values

- Zero values in specific columns were replaced with NaN, as zero may be biologically implausible for these measurements.
- Missing values were imputed using the median of their respective columns.

### 1.4 Label Mapping

- The labels were mapped to new values for easier interpretation and analysis.

### 1.5 Correlation Matrix

- A correlation matrix is visualized using a heatmap to understand relationships between features.
- The df.corr() method typically uses Pearson's correlation coefficient by default
- The correlation matrix of HRV metrics reveals strong positive correlations between several pairs, such as HRV_MeanNN with HRV_Prc20NN (0.99) and HRV_SDNN with HRV_RMSSD (0.97), indicating that these metrics often increase together. Conversely, some metrics show negative

correlations, such as HRV_LFHF with HRV_DFA_alpha1 (-0.39), suggesting inverse relationships.

### 1.6 Label Distribution

- The HRV metrics display a mix of normal (e.g., HRV_MeanNN, HRV_SDNN, HRV_RMSSD) and right-skewed distributions (e.g., HRV_LF, HRV_HF, HRV_TP)
- The distribution of the target variable (Label) was visualized using a count plot to identify class imbalance.

### 1.7 Data Scaling

To prepare the dataset for machine learning, the features were standardized to ensure uniform contribution to the model. The following steps were performed:

1. Splitting the Data:
    - The dataset was divided into features (X) and labels (y), where X contains all columns except the Label, and y contains only the Label column.
2. Standardizing the Features:
    - A StandardScaler from the sklearn library was utilized to standardize the features.

## 2) Feature Engineering

### 2.1 Feature Selection

1. SelectKBest Method:
    - The SelectKBest function from the sklearn library was employed to select the top k features that have the highest score in terms of their relationship with the target variable.
    - In this case, f_classif was used as the scoring function to measure the relevance of each feature to the classification

task.

2. Selecting Top Features:
    - The number of top features (k) was set to 10, meaning the 10 features with the highest scores were selected.
    - The fit_transform method was applied to the standardized feature set (X_scaled), resulting in a new dataset (X_kbest) that contains only the selected features.

### 2.2 Principal Component Analysis (PCA)

To further reduce the dimensionality of the data and capture the most important variance components, PCA was applied:

- Applying PCA:
    - PCA was performed on the selected features (X_kbest) to retain components that explain 95% of the variance.
    - The fit_transform method was used, resulting in a transformed dataset (X_pca) that captures the essential patterns while reducing complexity.
- Explained Variance Ratio:
    - The explained variance ratio for each principal component was plotted to visualize the contribution of each component to the total variance.
    - the first principal component explaining approximately 40% of the variance.

Observations from the 2D PCA Plot

- The scatter plot of the first two principal components shows that the data points are spread across the plot without clear, distinct clusters.
- there is considerable overlap among the labels

Observations from the 3D PCA Plot

- While the data points still show overlap among different labels, the third principal component helps to further differentiate the data

points, giving slightly better visual separation than the 2D plot.

## 3) Model Building and Evaluation

### 3.1 Handling Class Imbalance

- SMOTE (Synthetic Minority Over-sampling Technique) was used to handle class imbalance by generating synthetic samples for the minority classes.

### 3.2 Data Splitting

- The data was split into training and testing sets with a 70-30 split, ensuring stratification to maintain class proportions.

### 3.3 Model Training and Hyperparameter Tuning

- Various models were trained using GridSearchCV for hyperparameter tuning:
  - Logistic Regression
  - K-Nearest Neighbors (KNN)
  - Decision Tree
  - Random Forest
  - Gradient Boosting

### 3.4 Ensembling:

- A Voting Classifier was used to combine the predictions of the individual models for improved accuracy.

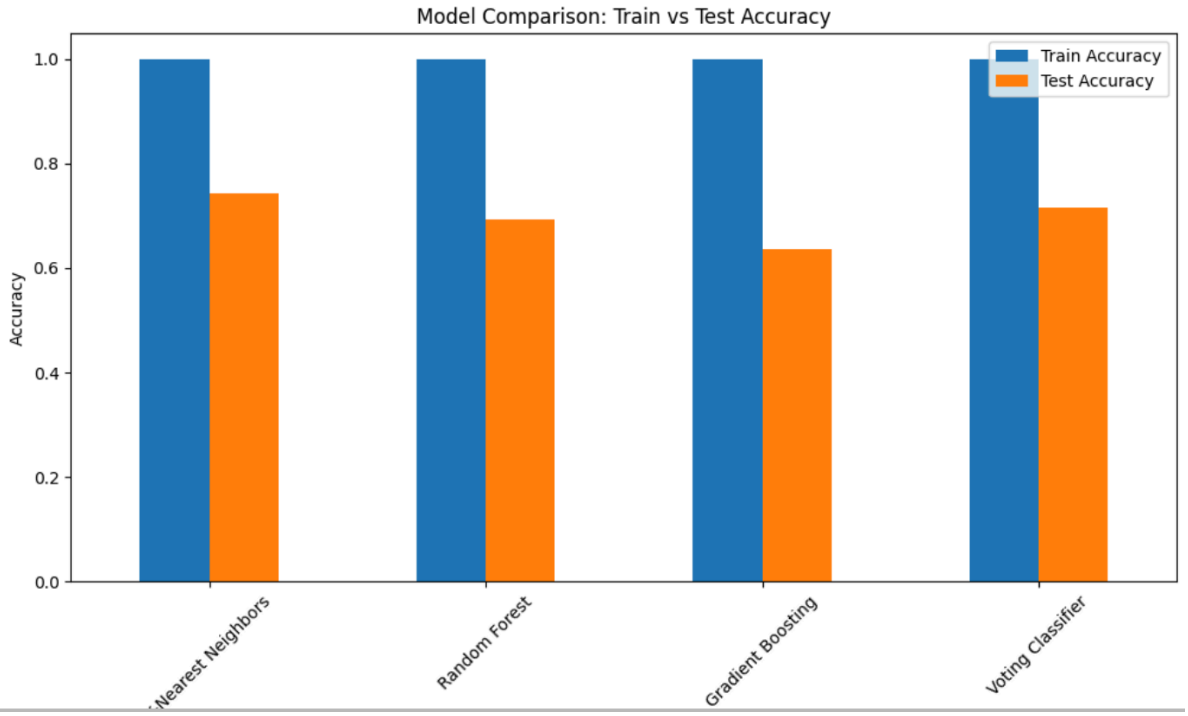
### 3.5 Model Evaluation:

- The models were evaluated using training accuracy, and test accuracy
- Confusion matrices were plotted for each model to visualise the performance.

## 4) Results

The evaluation metrics for each model are summarized below:

| | Train Accuracy | Test Accuracy |
|---|---|---|
| K-Nearest Neighbors | 0.99872 | 0.743284 |
| Random Forest | 0.99872 | 0.692537 |
| Gradient Boosting | 0.99872 | 0.635821 |
| Voting Classifier | 0.99872 | 0.716418 |




# CONCLUSION

K-Nearest Neighbors (KNN) demonstrates the best overall performance, achieving the highest test accuracy. It effectively handles various classes with fewer misclassifications compared to other models. The Voting Classifier and other methods perform well but do not surpass the accuracy of KNN in this scenario.