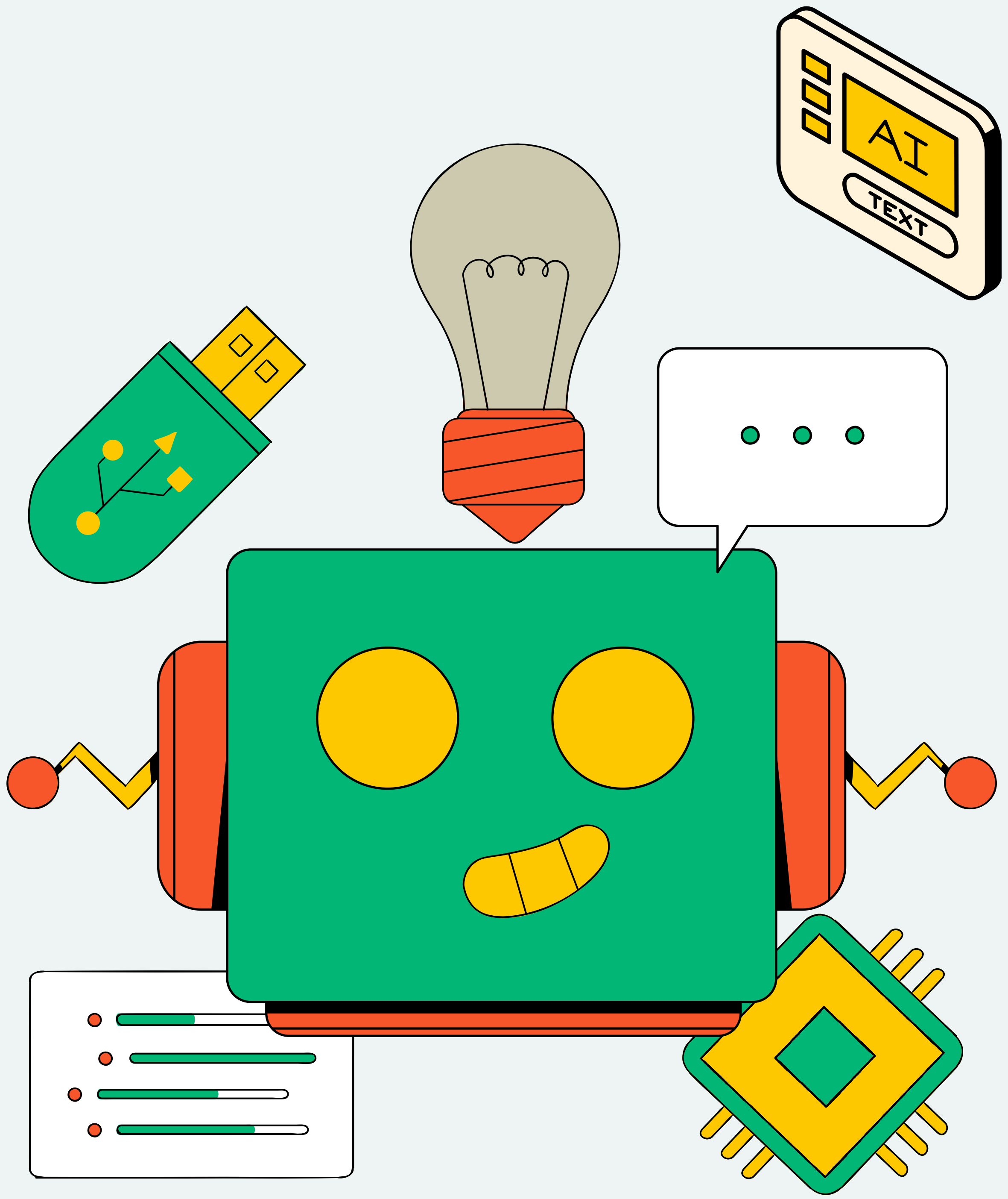
THINK UNLIMITED
WE LEARN FOR THE FUTURE

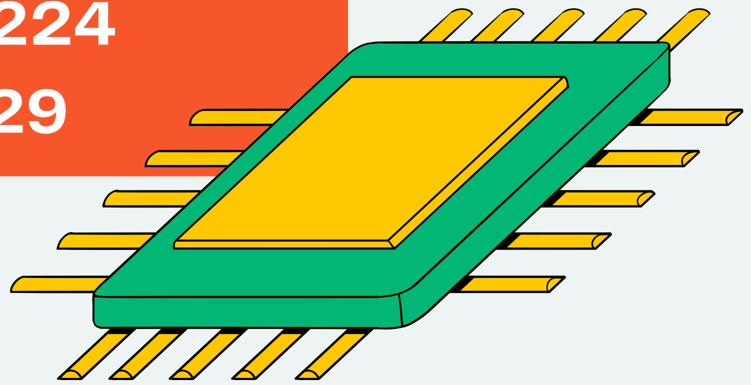


OBESITY RISK PREDICTION

CAPSTONE PROJECT

PRESENTED BY: GROUP 2

NGUYỄN TRỌNG HUY - 20210451
NGUYỄN CHÍNH MINH - 20215224
TRỊNH GIANG NAM - 20215229



PRESENTATION OUTLINE

- Introduction
- Dataset and data preprocessing
- Classification model
- Training model
- Experimental result
- Conclusion

INTRODUCTION

Obesity is a major public health concern globally, associated with numerous health conditions influenced by various genetic, environmental, and behavioral factors.

With the advent of machine learning (ML), there is an opportunity to leverage advanced algorithms to improve the accuracy of obesity prediction.

PROJECT'S OBJECTIVES

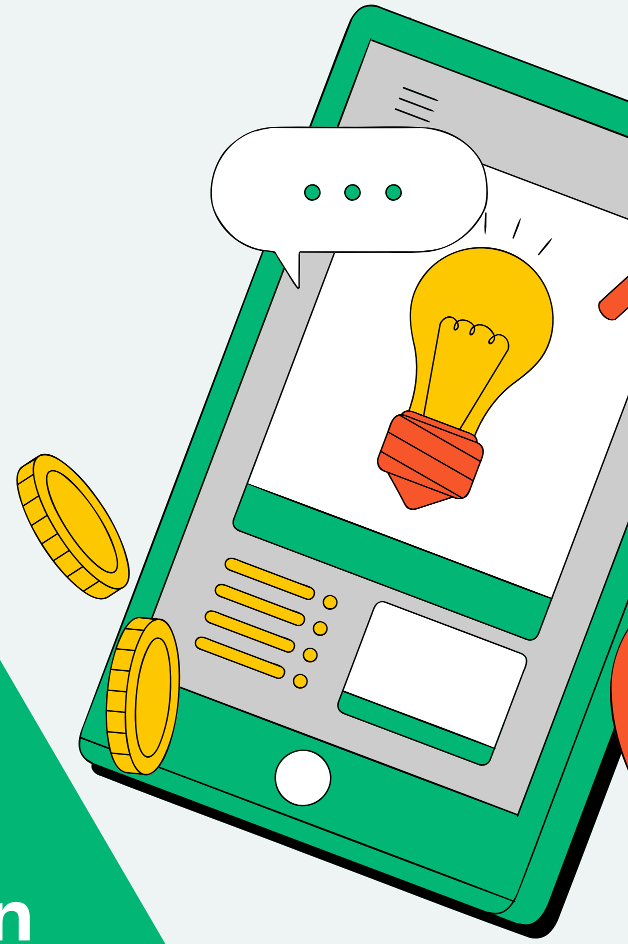
Data Collection and Preprocessing

Feature Selection and Engineering

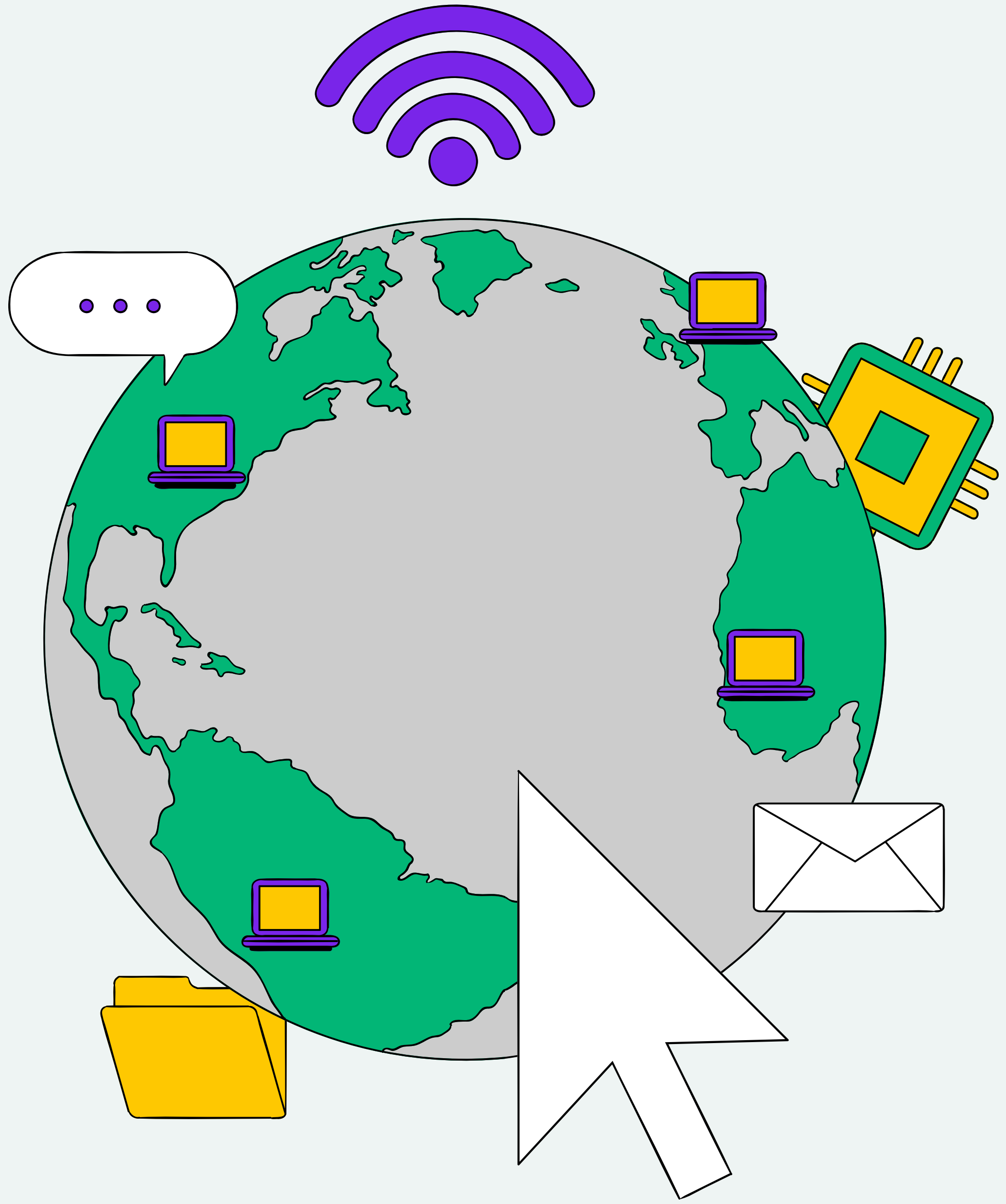
Model Development and Training

Model Evaluation

Implementation and Validation



DATASET



The dataset we use for this problem is driven from Kaggle competition, which is a table-type dataset, including 18 columns and 20758 rows

The data contains 17 attributes (one attribute is for ID) , the records are labeled with the class variable NObesity (Obesity Level), using the labels of

- Insufficient Weight
- Normal Weight
- Overweight Level I
- Overweight Level II
- Obesity Type I
- Obesity Type II
- Obesity Type III

https://www.kaggle.com/competitions/playground-series-s4e2/data

DATASET

Feature	Datatype	Description
ID	Categorical	Unique identifier
Smoke	Categorical	Smoker or not
Weight	Numerical	Weight (Float)
Age	Numerical	Age (Float)
Height	Numerical	Height (Float)
Gender	Categorical	Gender
Family_history_with_overweight	Categorical	Family history with overweight
FAVC	Categorical	Frequent consumption of high-caloric food items
FCVC	Numerical	Frequency of consuming vegetables (Float)
NCP	Numerical	Number of main meals consumed per day (Float)
CAEC	Categorical	Frequency of consuming food between meals
CH20	Numerical	Amount of water consumed daily (Float)
CALC	Categorical	Frequency of alcohol consumption
SCC	Categorical	Monitoring of calorie consumption
FAF	Numerical	Frequency of engaging in physical activity (Float)
TUE	Numerical	Time spent using technology devices (Float)
MTRANS	Categorical	Mode of transportation used

DATA PREPROCESSING

01

DATA SPLITTING

Divide the dataset into training and test using 80% of the dataset for training the model and the remaining 20% for validating the model's accuracy.

The feature on which the dataset is split into training and testing is specified by the function's stratify property

02

EDA

Take it in two main steps:

Univariate Analysis: focus on one feature at a time to understand its distribution and range.

Bivariate Analysis: explore the relationship between each feature and the target variable.

03

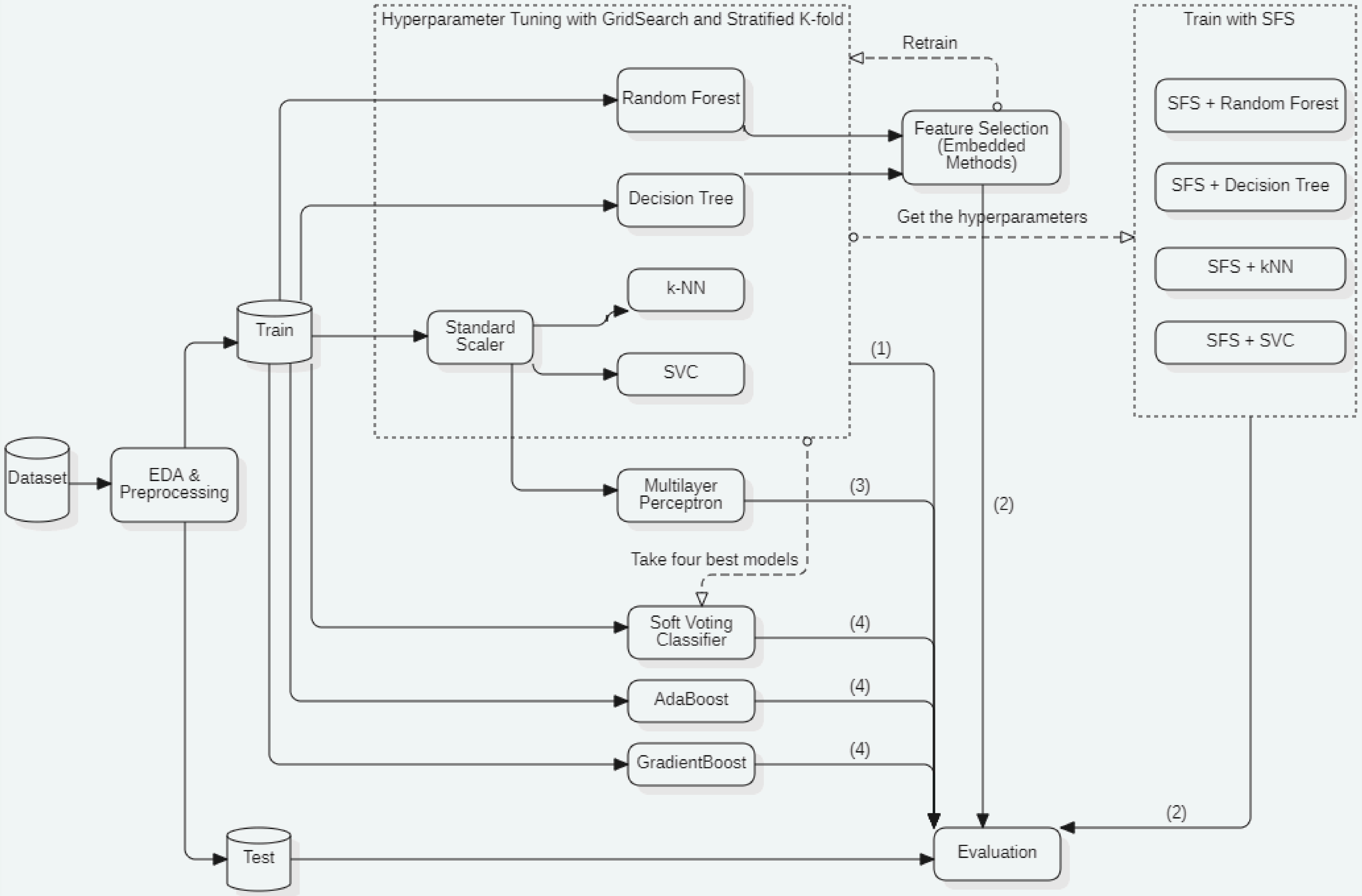
PREPROCESSING

In this part, do the following steps: remove irrelevant attribute, handle missing values, encode the categorical features and scale data.

After exploring the data, we found that the id attribute is irrelevant so we remove it from the data.

About handling missing values, there is no missing value in our dataset, so we skipped this step

FULL PIPELINE



CLASSIFICATION MODEL

BASELINE

- Decision Tree
- k-NN
- SVM
- Multilayer Perceptrons

ENSEMBLE

- Random Forest
- Voting Classifier
- Adaptive Boosting
- Gradient Boosting

MODEL ASSESSMENT

- Accuracy
- F1-score

TRAINING CLASSIFICATION MODELS

1) Train and tune the hyperparameters of the models: kNN, Decision Tree, Random Forest, and SVC

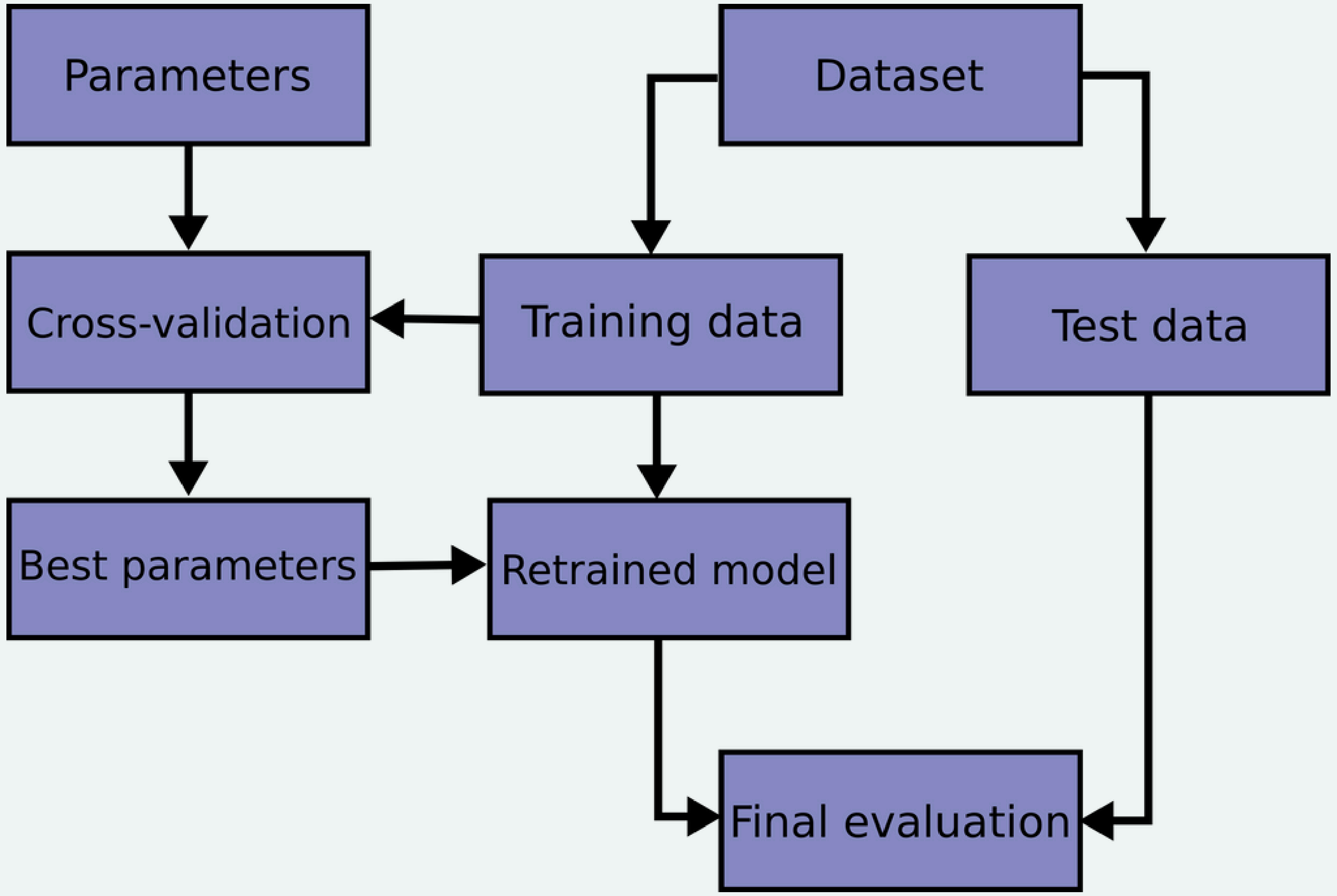
(2) Train the models with the hyperparameters tuned from (1) along with the feature selection method

(3) Train multilayer perceptrons

(4) Train with ensem-ble methods (including Voting Classifier, AdaBoost, and GradientBoost)

TRAINING KNN, DECISION TREE, RANDOM FOREST, SVC

- Tune hyperparameters of each model using Grid Search method combined with Stratified K-Fold method with k = 5.
- Metric used to find best hyperparameters for each model is ‘accuracy’.
- After finding best hyperparameters for each model models will be retrained with original training data to produce best results.



Model	Hyperparameters
k-NN	metric: manhattan, n_neighbors: 17, weights: distance
Decision Tree	criterion: entropy, max_depth: 11, min_samples_leaf: 10, min_samples_split: 2, splitter: best
Random Forest	criterion: entropy, max_depth: None, max_features: sqrt, n_estimators: 900
SVC	C: 5, kernel: linear

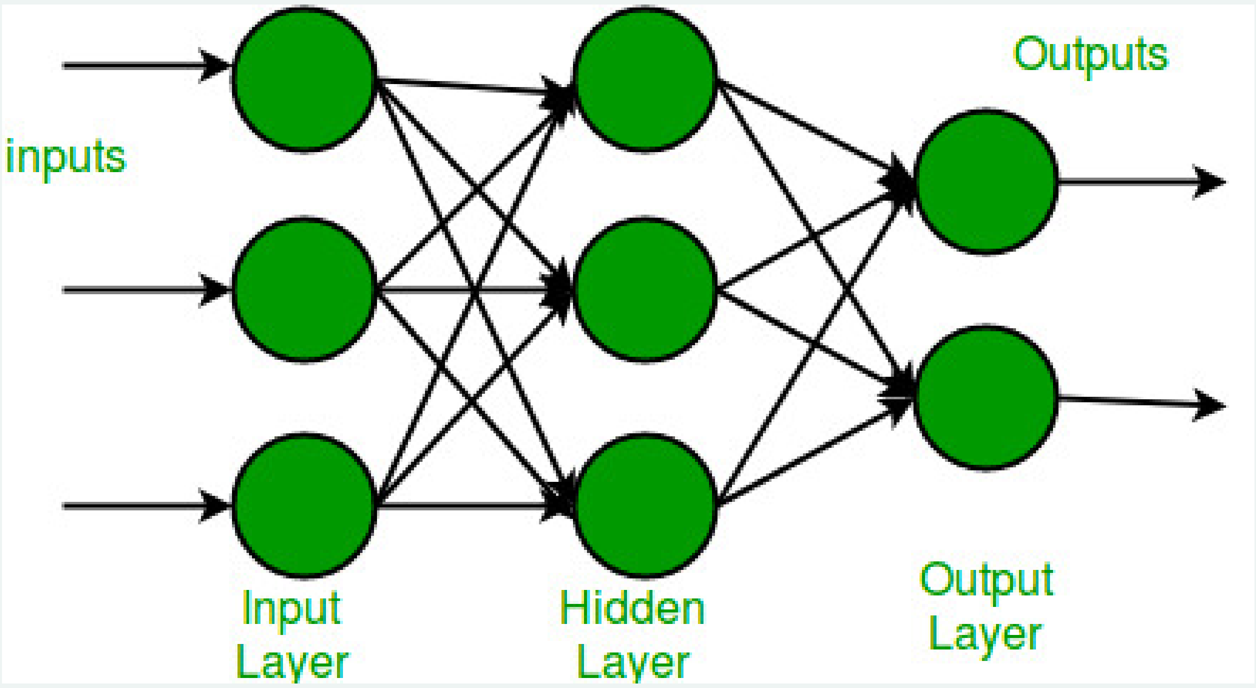
TRAINING WITH FEATURE SELECTION

We using 2 main methods.

Firstly, we use SFS method, which implemented along with the baseline model with selected hyperparameters in a pipeline, then training model with maximum 10 features selected which drive best ‘accuracy’ score.

For the second method, we training Decision Tree and Random Forest to deliver the best selected combination of 10 features. Then we use these two combination of featured to train again with the model with selected hyperparameters.

TRAINING MULTILAYER PERCEPTIONS



Our selected multiple perceptron model is a multilayer perceptron (MLP) implemented using the Keras Sequential API

- Dropout layer: Each hidden layer is connected with a dropout layer of rate 0.01 to avoid overfitting.
- Loss Function: The loss function used for training is sparse categorical cross entropy
- Optimizer: The Adam optimizer with a learning rate of 0.01 is used to optimize the model parameters during training

We train the multi-layer perceptron model for up to 50 epochs, with a validation set ratio of 0.2 and a batch size of 32.

To prevent overfitting on the training set and enhance the model's generalization, we use Early Stopping with the validation loss as the monitored metric, and a patience of 7 epochs.

Additionally, we set the parameter restore_best_weights to True, which saves the model at the checkpoint with the best evaluation score.

TRAINING ENSEMBLE MODELS

01

VOTING CLASSIFIER

Combine four base models: kNN, Decision Tree, Random Forest, and SVC with the hyperparameters selected from the previous section.

The voting method used is soft voting.

02

ADAPTIVE BOOSTING

Use the Decision Tree as the base estimator with the hyperparameters chosen from the previous section.

Then, we tune the hyperparameters of AdaBoost, including the number of trees and the learning rate.

After the hyperparameter tuning process using GridSearch with Stratified K-Fold, we obtain the hyperparameters for AdaBoost as n_estimators = 500 and learning_rate = 1.

03

GRADIENT BOOSTING

Perform hyperparameter tuning with the parameters: n_estimators, learning_rate, and max_depth.

After the tuning process, the values of the hyperparameters are 200, 0.1, and 3, respectively

EXPERIMENTAL RESULT

Model	Feature Selection			Macro F1 (%)	Weighted F1 (%)	Accuracy (%)
	Decision Tree	Random Forest	SFS			
Decision Tree	-	-	-	86.78	88.00	88.03
	yes	-	-	86.40	87.66	87.67
	-	yes	-	86.12	87.43	87.48
	-	-	yes	86.14	87.34	87.40
Random Forest	-	-	-	89.28	90.31	90.37
	yes	-	-	88.79	89.88	89.93
	-	yes	-	88.36	89.50	89.55
	-	-	yes	89.11	90.13	90.17
k-NN	-	-	-	77.17	79.12	79.36
	yes	-	-	78.15	80.13	80.32
	-	yes	-	77.44	79.63	80.08
	-	-	yes	83.78	85.20	85.24
SVC	-	-	-	85.50	86.86	86.95
	yes	-	-	84.69	86.17	86.27
	-	yes	-	84.93	86.35	86.42
	-	-	yes	84.99	86.37	86.46
Multilayer Perceptron	-	-	-	87.42	88.61	88.66
Voting Classifier	-	-	-	89.22	90.21	90.29
AdaBoost	-	-	-	89.19	90.21	90.20
GradientBoosting	-	-	-	**89.54**	**90.53**	**90.56**

Gradient Boosting method gives the best results across all three metrics.

For feature selection methods, SFS provides features with higher importance compared to the methods using Decision Tree and Random Forest.

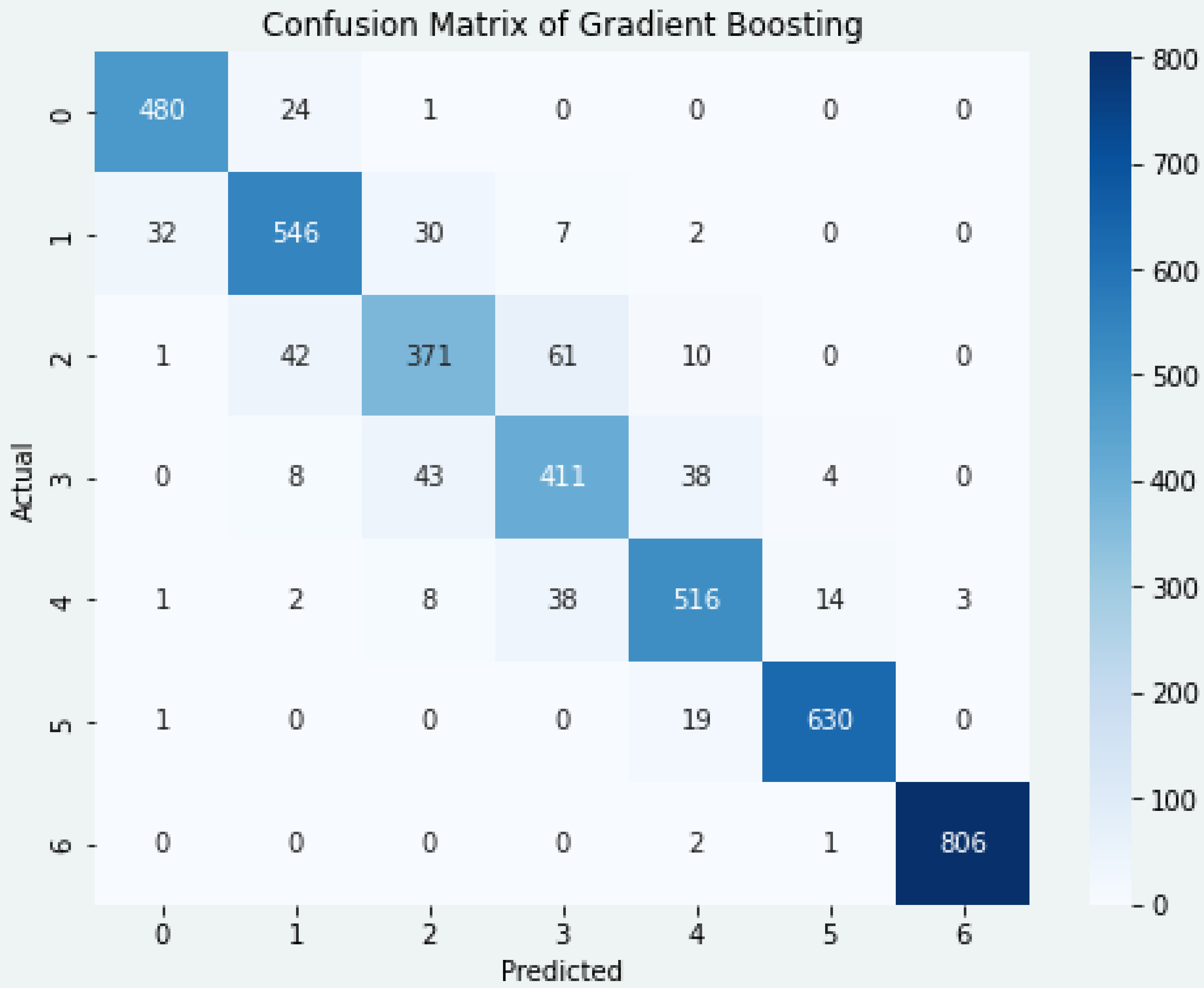
EXPERIMENTAL RESULT

Model	Feature Selection			Macro F1 (%)	Weighted F1 (%)	Accuracy (%)
	Decision Tree	Random Forest	SFS			
Decision Tree	-	-	-	86.78	88.00	88.03
	yes	-	-	86.40	87.66	87.67
	-	yes	-	86.12	87.43	87.48
	-	-	yes	86.14	87.34	87.40
Random Forest	-	-	-	89.28	90.31	90.37
	yes	-	-	88.79	89.88	89.93
	-	yes	-	88.36	89.50	89.55
	-	-	yes	89.11	90.13	90.17
k-NN	-	-	-	77.17	79.12	79.36
	yes	-	-	78.15	80.13	80.32
	-	yes	-	77.44	79.63	80.08
	-	-	yes	83.78	85.20	85.24
SVC	-	-	-	85.50	86.86	86.95
	yes	-	-	84.69	86.17	86.27
	-	yes	-	84.93	86.35	86.42
	-	-	yes	84.99	86.37	86.46
Multilayer Perceptron	-	-	-	87.42	88.61	88.66
Voting Classifier	-	-	-	89.22	90.21	90.29
AdaBoost	-	-	-	89.19	90.21	90.20
GradientBoosting	-	-	-	**89.54**	**90.53**	**90.56**

Gradient Boosting method gives the best results across all three metrics.

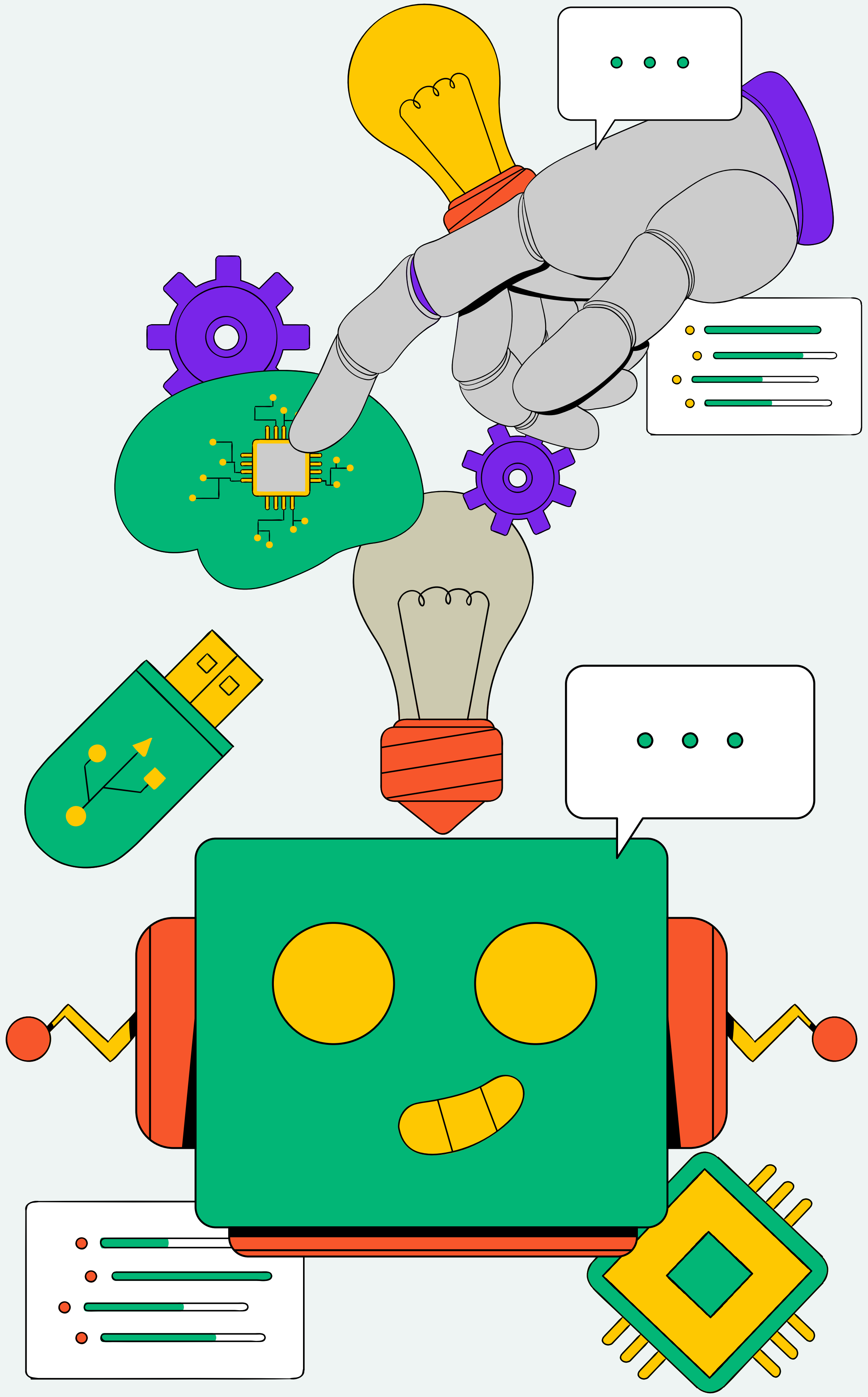
For feature selection methods, SFS provides features with higher importance compared to the methods using Decision Tree and Random Forest.

CONFUSION MATRIX

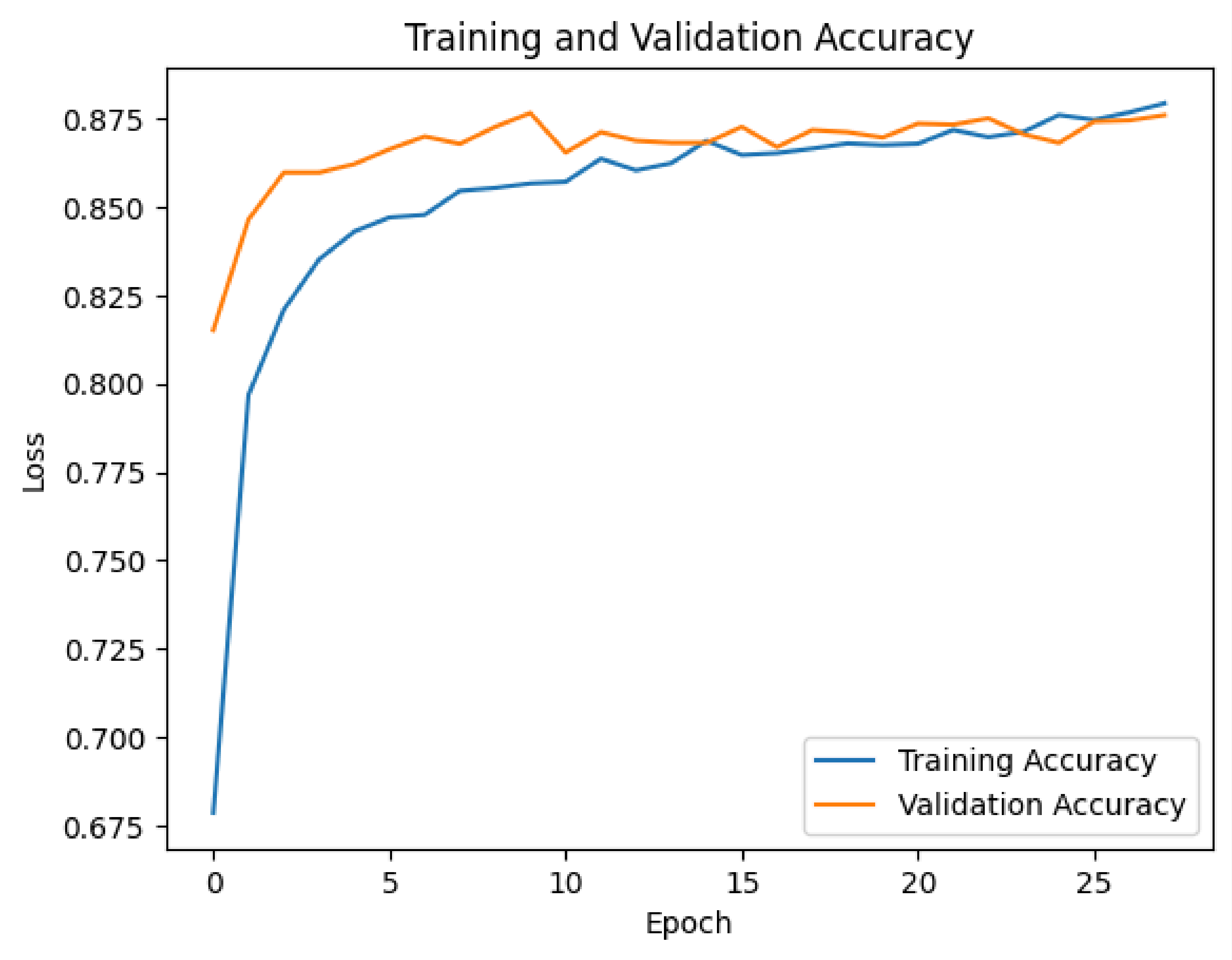


QUESTIONS AND ANSWERS

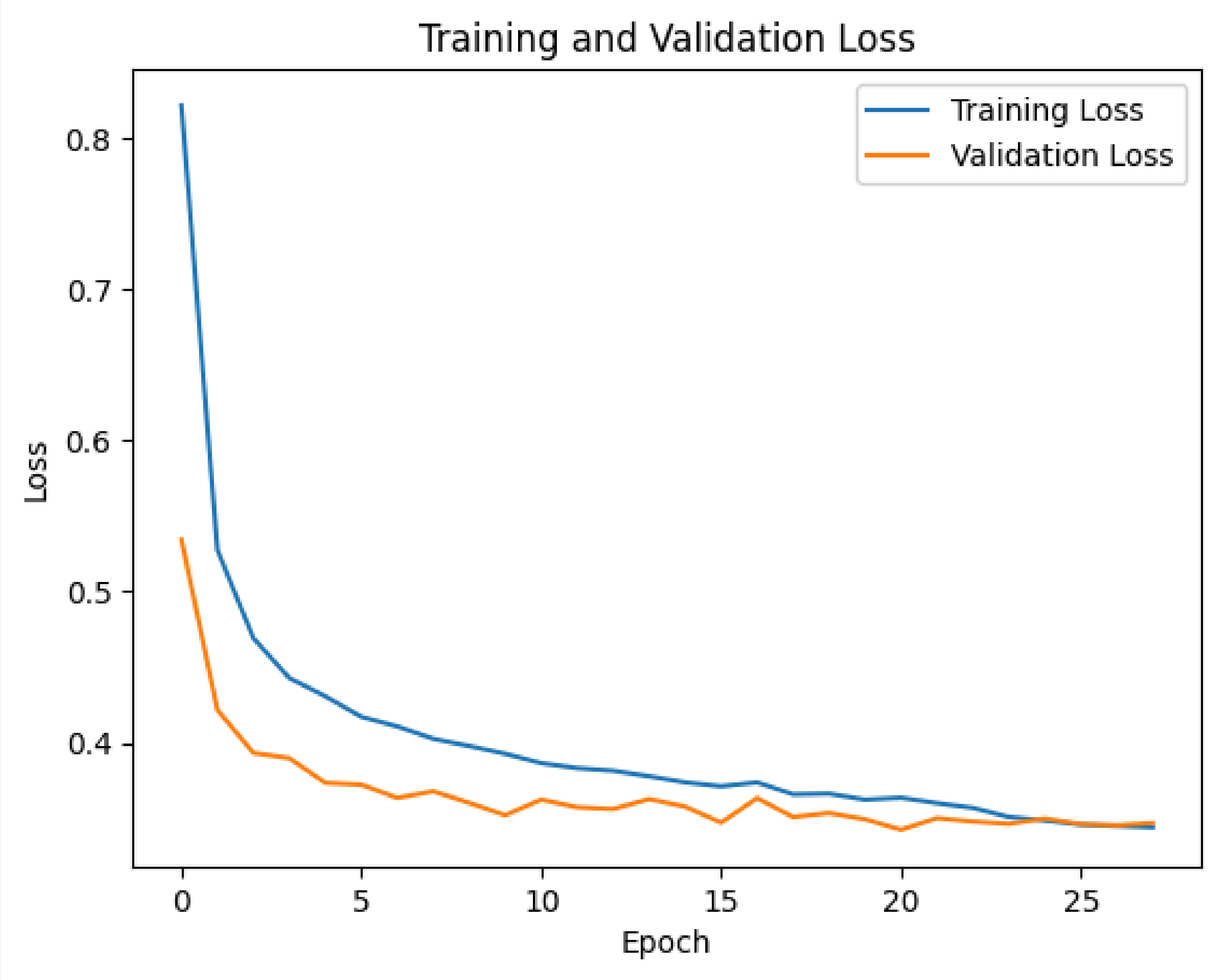
Your insights and questions are highly valuable to us, and we want to create an engaging and interactive session. Please feel free to send us your questions and concerns for clarifications.



ANN



Training and Validation Accuracy
0.875
0.850
0.825
0.800
0.775
0.750
0.725
0.700
0.675
Loss
0
5
10
15
20
25
Epoch
Training Accuracy
Validation Accuracy



Training and Validation Loss
0.8
0.7
0.6
0.5
0.4
Loss
0
5
10
15
20
25
Epoch
Training Loss
Validation Loss