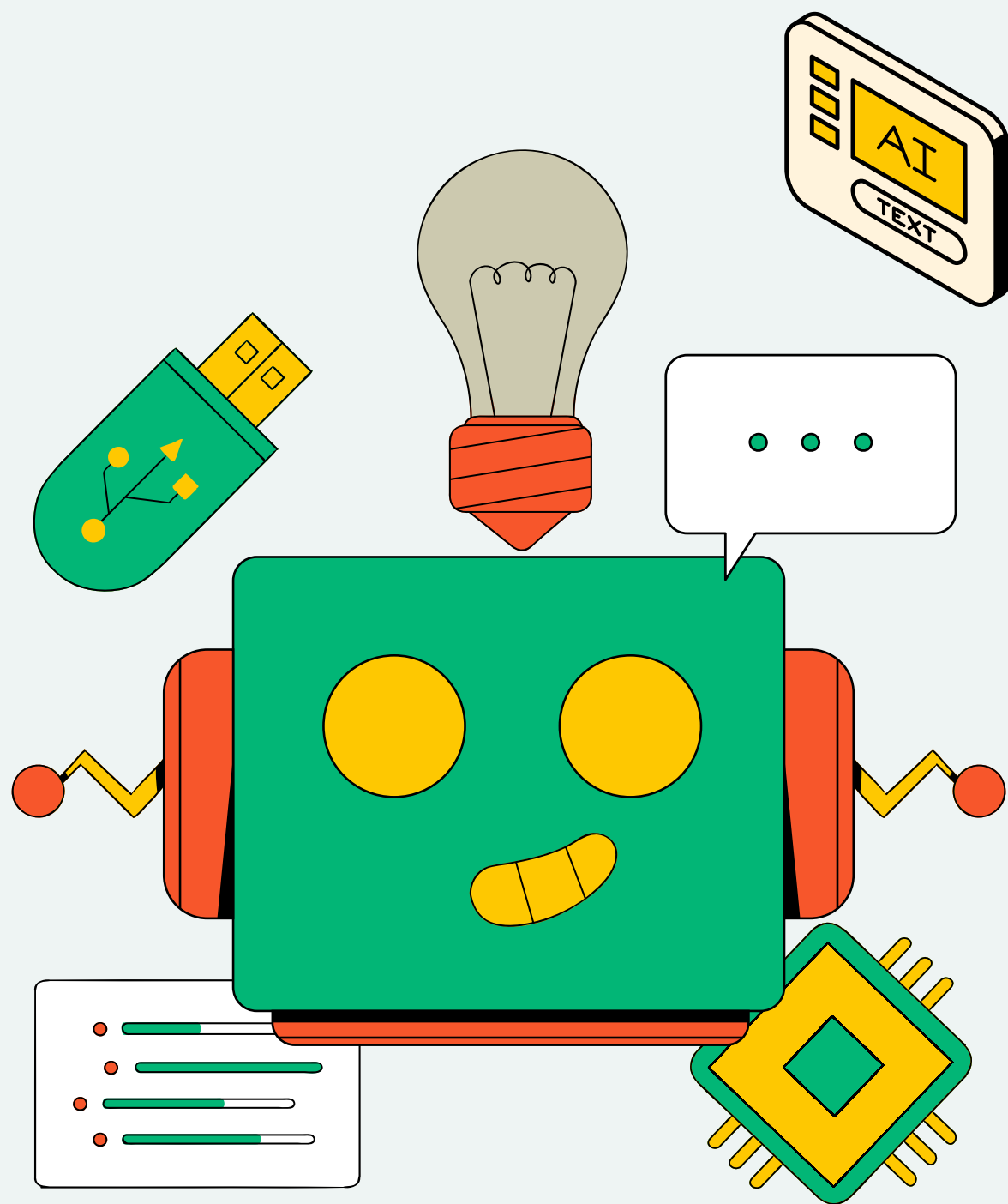


THINK UNLIMITED
WE LEARN FOR THE FUTURE

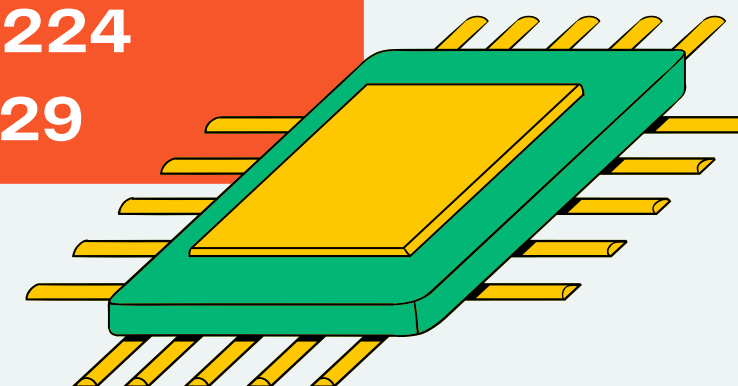


OBESITY RISK PREDICTION

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



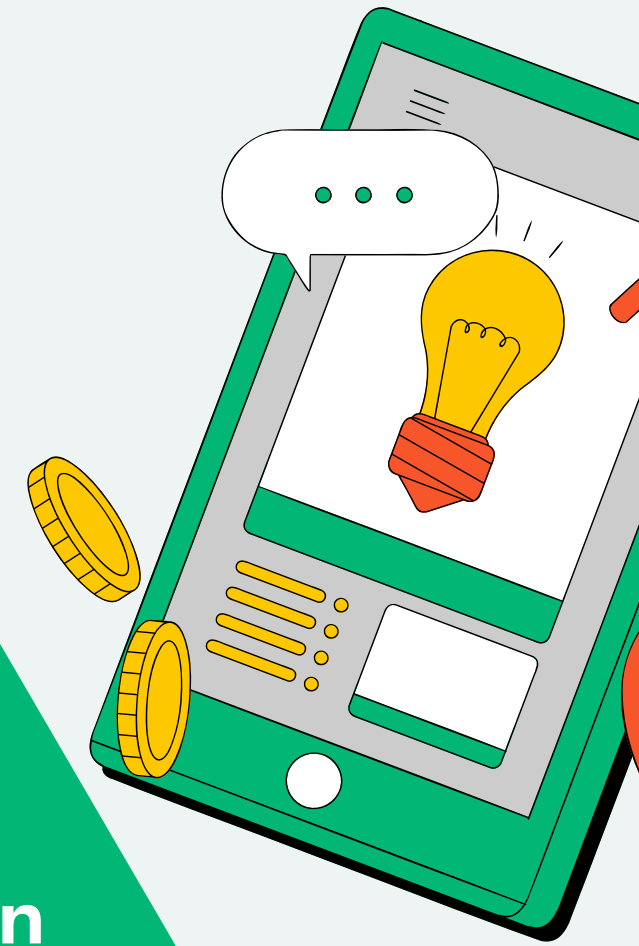
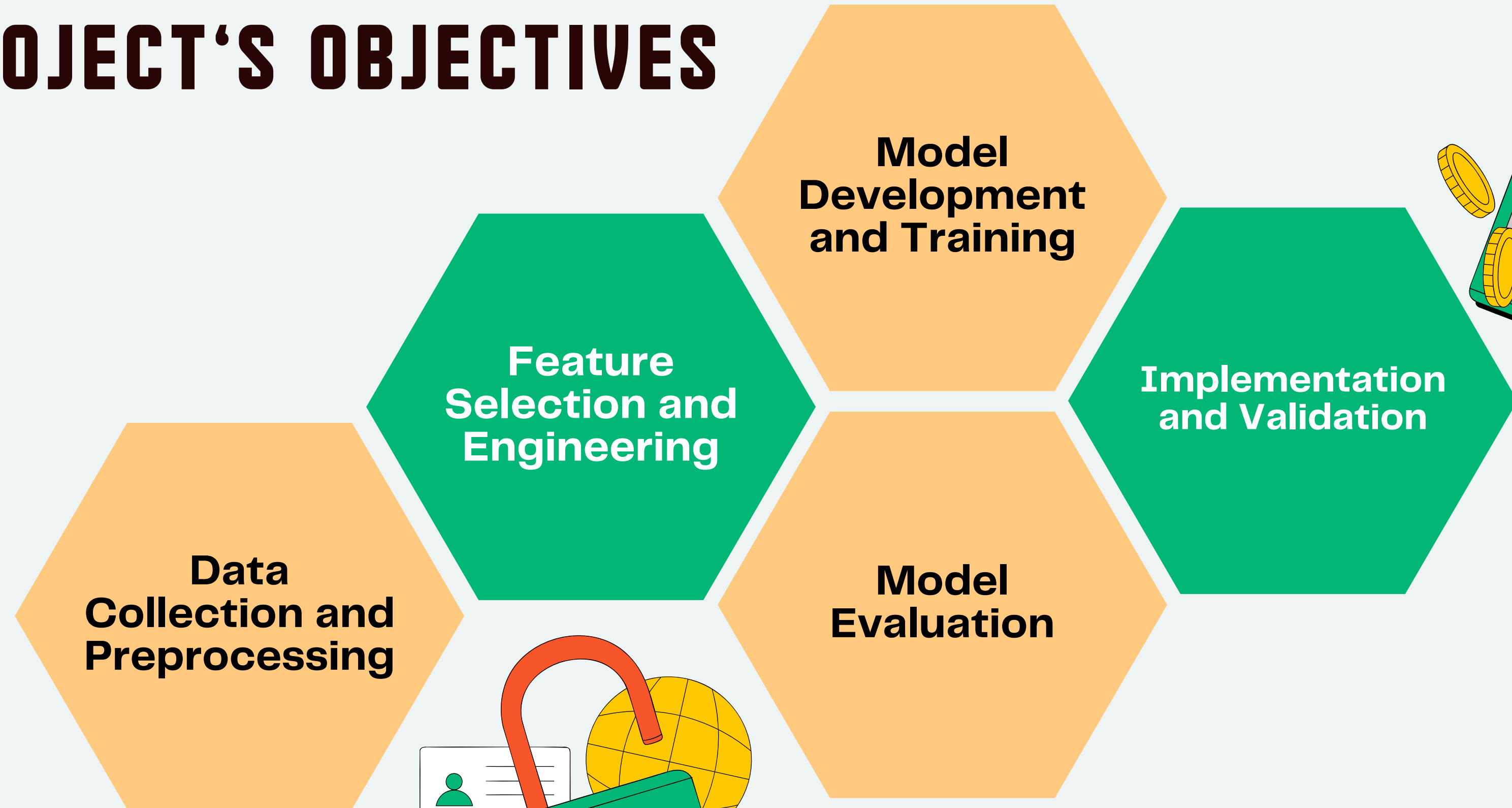
INTRODUCTION

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.

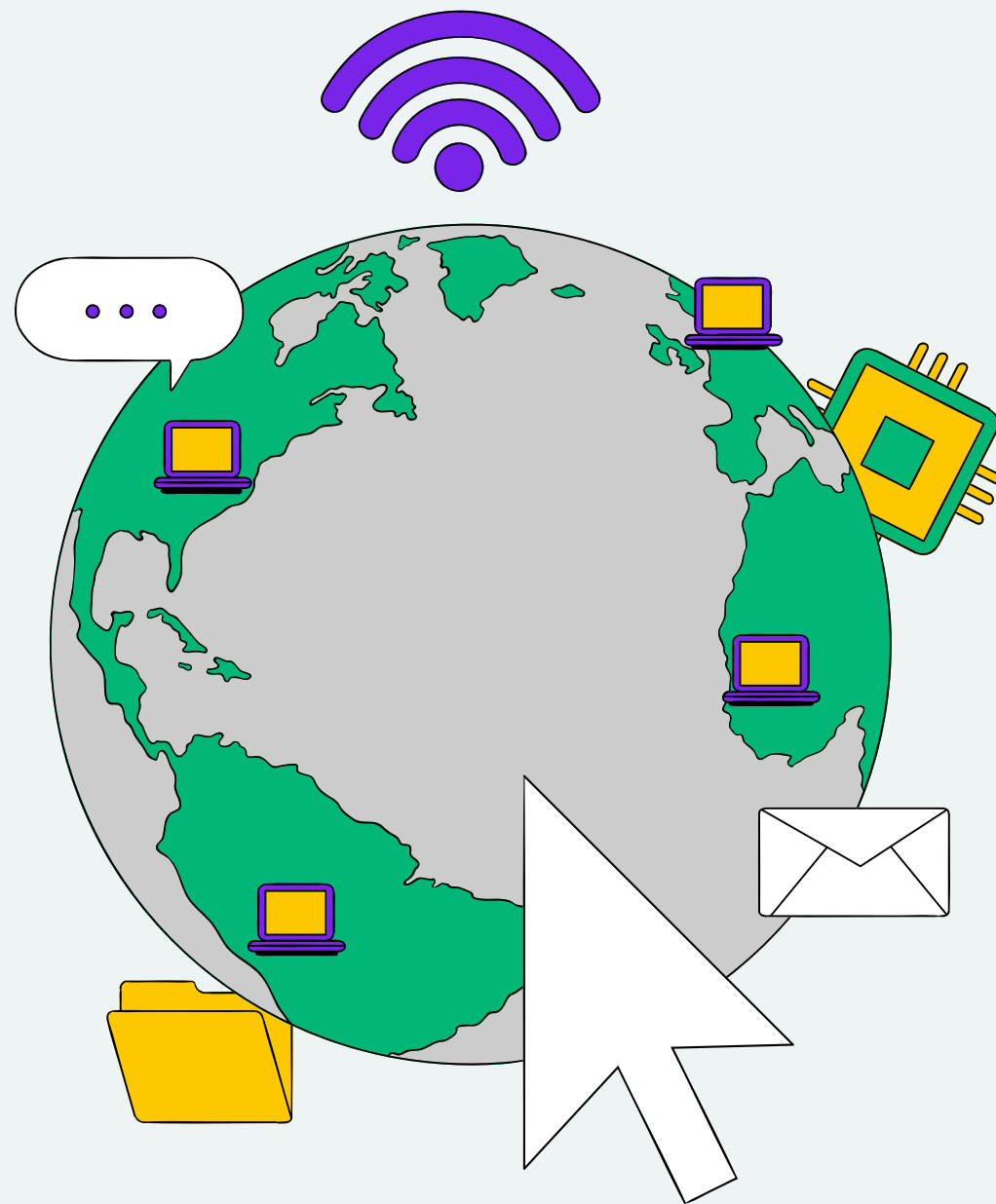


PROJECT'S OBJECTIVES



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



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



The flowchart illustrates a machine learning pipeline for hyperparameter tuning and feature selection. It starts with a **Dataset** input, which is processed through **EDA & Preprocessing**. The data is then split into **Train** and **Test** sets. The **Train** set is used for **Hyperparameter Tuning with GridSearch and Stratified K-fold**, which involves training various models: **Random Forest**, **Decision Tree**, **k-NN**, **SVC**, **Multilayer Perceptron**, **Soft Voting Classifier**, **AdaBoost**, and **GradientBoost**. The **Test** set is also used for training these models. The **Feature Selection (Embedded Methods)** step is applied to the **Train** set, and the resulting models are evaluated. The **Evaluation** step is performed on the **Test** set. The **Hyperparameter Tuning** step is a feedback loop that involves **Retrain** and **Get the hyperparameters** to optimize the models. The **Feature Selection** step is also a feedback loop that involves **Retrain** and **Get the hyperparameters** to optimize the models. The final output is the **Evaluation** of the models, which is used to select the best model for deployment.

```
graph LR
    Dataset[(Dataset)] --> EDA[EDA & Preprocessing]
    EDA --> Train[(Train)]
    EDA --> Test[(Test)]
    Train --> RF[Random Forest]
    Train --> DT[Decision Tree]
    Train --> kNN[k-NN]
    Train --> SVC[SVC]
    Train --> MLP[Multilayer Perceptron]
    Train --> SVC_V[Soft Voting Classifier]
    Train --> AdaBoost[AdaBoost]
    Train --> GradientBoost[GradientBoost]
    Train --> FS[Feature Selection Embedded Methods]
    FS --> RF
    FS --> DT
    FS --> kNN
    FS --> SVC
    FS --> MLP
    FS --> SVC_V
    FS --> AdaBoost
    FS --> GradientBoost
    FS --> FS
    FS --> Retrain1[Retrain]
    Retrain1 --> FS
    FS --> GetHP1[Get the hyperparameters]
    GetHP1 --> FS
    GetHP1 --> FS2[Feature Selection Embedded Methods]
    FS2 --> FS
    FS2 --> Retrain2[Retrain]
    Retrain2 --> FS2
    FS2 --> GetHP2[Get the hyperparameters]
    GetHP2 --> FS2
    GetHP2 --> FS3[Feature Selection Embedded Methods]
    FS3 --> FS2
    FS3 --> Retrain3[Retrain]
    Retrain3 --> FS3
    FS3 --> GetHP3[Get the hyperparameters]
    GetHP3 --> FS3
    GetHP3 --> FS4[Feature Selection Embedded Methods]
    FS4 --> FS3
    FS4 --> Retrain4[Retrain]
    Retrain4 --> FS4
    FS4 --> GetHP4[Get the hyperparameters]
    GetHP4 --> FS4
    GetHP4 --> FS5[Feature Selection Embedded Methods]
    FS5 --> FS4
    FS5 --> Retrain5[Retrain]
    Retrain5 --> FS5
    FS5 --> GetHP5[Get the hyperparameters]
    GetHP5 --> FS5
    GetHP5 --> FS6[Feature Selection Embedded Methods]
    FS6 --> FS5
    FS6 --> Retrain6[Retrain]
    Retrain6 --> FS6
    FS6 --> GetHP6[Get the hyperparameters]
    GetHP6 --> FS6
    GetHP6 --> FS7[Feature Selection Embedded Methods]
    FS7 --> FS6
    FS7 --> Retrain7[Retrain]
    Retrain7 --> FS7
    FS7 --> GetHP7[Get the hyperparameters]
    GetHP7 --> FS7
    GetHP7 --> FS8[Feature Selection Embedded Methods]
    FS8 --> FS7
    FS8 --> Retrain8[Retrain]
    Retrain8 --> FS8
    FS8 --> GetHP8[Get the hyperparameters]
    GetHP8 --> FS8
    GetHP8 --> FS9[Feature Selection Embedded Methods]
    FS9 --> FS8
    FS9 --> Retrain9[Retrain]
    Retrain9 --> FS9
    FS9 --> GetHP9[Get the hyperparameters]
    GetHP9 --> FS9
    GetHP9 --> FS10[Feature Selection Embedded Methods]
    FS10 --> FS9
    FS10 --> Retrain10[Retrain]
    Retrain10 --> FS10
    FS10 --> GetHP10[Get the hyperparameters]
    GetHP10 --> FS10
    GetHP10 --> FS11[Feature Selection Embedded Methods]
    FS11 --> FS10
    FS11 --> Retrain11[Retrain]
    Retrain11 --> FS11
    FS11 --> GetHP11[Get the hyperparameters]
    GetHP11 --> FS11
    GetHP11 --> FS12[Feature Selection Embedded Methods]
    FS12 --> FS11
    FS12 --> Retrain12[Retrain]
    Retrain12 --> FS12
    FS12 --> GetHP12[Get the hyperparameters]
    GetHP12 --> FS12
    GetHP12 --> FS13[Feature Selection Embedded Methods]
    FS13 --> FS12
    FS13 --> Retrain13[Retrain]
    Retrain13 --> FS13
    FS13 --> GetHP13[Get the hyperparameters]
    GetHP13 --> FS13
    GetHP13 --> FS14[Feature Selection Embedded Methods]
    FS14 --> FS13
    FS14 --> Retrain14[Retrain]
    Retrain14 --> FS14
    FS14 --> GetHP14[Get the hyperparameters]
    GetHP14 --> FS14
    GetHP14 --> FS15[Feature Selection Embedded Methods]
    FS15 --> FS14
    FS15 --> Retrain15[Retrain]
    Retrain15 --> FS15
    FS15 --> GetHP15[Get the hyperparameters]
    GetHP15 --> FS15
    GetHP15 --> FS16[Feature Selection Embedded Methods]
    FS16 --> FS15
    FS16 --> Retrain16[Retrain]
    Retrain16 --> FS16
    FS16 --> GetHP16[Get the hyperparameters]
    GetHP16 --> FS16
    GetHP16 --> FS17[Feature Selection Embedded Methods]
    FS17 --> FS16
    FS17 --> Retrain17[Retrain]
    Retrain17 --> FS17
    FS17 --> GetHP17[Get the hyperparameters]
    GetHP17 --> FS17
    GetHP17 --> FS18[Feature Selection Embedded Methods]
    FS18 --> FS17
    FS18 --> Retrain18[Retrain]
    Retrain18 --> FS18
    FS18 --> GetHP18[Get the hyperparameters]
    GetHP18 --> FS18
    GetHP18 --> FS19[Feature Selection Embedded Methods]
    FS19 --> FS18
    FS19 --> Retrain19[Retrain]
    Retrain19 --> FS19
    FS19 --> GetHP19[Get the hyperparameters]
    GetHP19 --> FS19
    GetHP19 --> FS20[Feature Selection Embedded Methods]
    FS20 --> FS19
    FS20 --> Retrain20[Retrain]
    Retrain20 --> FS20
    FS20 --> GetHP20[Get the hyperparameters]
    GetHP20 --> FS20
    GetHP20 --> FS21[Feature Selection Embedded Methods]
    FS21 --> FS20
    FS21 --> Retrain21[Retrain]
    Retrain21 --> FS21
    FS21 --> GetHP21[Get the hyperparameters]
    GetHP21 --> FS21
    GetHP21 --> FS22[Feature Selection Embedded Methods]
    FS22 --> FS21
    FS22 --> Retrain22[Retrain]
    Retrain22 --> FS22
    FS22 --> GetHP22[Get the hyperparameters]
    GetHP22 --> FS22
    GetHP22 --> FS23[Feature Selection Embedded Methods]
    FS23 --> FS22
    FS23 --> Retrain23[Retrain]
    Retrain23 --> FS23
    FS23 --> GetHP23[Get the hyperparameters]
    GetHP23 --> FS23
    GetHP23 --> FS24[Feature Selection Embedded Methods]
    FS24 --> FS23
    FS24 --> Retrain24[Retrain]
    Retrain24 --> FS24
    FS24 --> GetHP24[Get the hyperparameters]
    GetHP24 --> FS24
    GetHP24 --> FS25[Feature Selection Embedded Methods]
    FS25 --> FS24
    FS25 --> Retrain25[Retrain]
    Retrain25 --> FS25
    FS25 --> GetHP25[Get the hyperparameters]
    GetHP25 --> FS25
    GetHP25 --> FS26[Feature Selection Embedded Methods]
    FS26 --> FS25
    FS26 --> Retrain26[Retrain]
    Retrain26 --> FS26
    FS26 --> GetHP26[Get the hyperparameters]
    GetHP26 --> FS26
    GetHP26 --> FS27[Feature Selection Embedded Methods]
    FS27 --> FS26
    FS27 --> Retrain27[Retrain]
    Retrain27 --> FS27
    FS27 --> GetHP27[Get the hyperparameters]
    GetHP27 --> FS27
    GetHP27 --> FS28[Feature Selection Embedded Methods]
    FS28 --> FS27
    FS28 --> Retrain28[Retrain]
    Retrain28 --> FS28
    FS28 --> GetHP28[Get the hyperparameters]
    GetHP28 --> FS28
    GetHP28 --> FS29[Feature Selection Embedded Methods]
    FS29 --> FS28
    FS29 --> Retrain29[Retrain]
    Retrain29 --> FS29
    FS29 --> GetHP29[Get the hyperparameters]
    GetHP29 --> FS29
    GetHP29 --> FS30[Feature Selection Embedded Methods]
    FS30 --> FS29
    FS30 --> Retrain30[Retrain]
    Retrain30 --> FS30
    FS30 --> GetHP30[Get the hyperparameters]
    GetHP30 --> FS30
    GetHP30 --> FS31[Feature Selection Embedded Methods]
    FS31 --> FS30
    FS31 --> Retrain31[Retrain]
    Retrain31 --> FS31
    FS31 --> GetHP31[Get the hyperparameters]
    GetHP31 --> FS31
    GetHP31 --> FS32[Feature Selection Embedded Methods]
    FS32 --> FS31
    FS32 --> Retrain32[Retrain]
    Retrain32 --> FS32
    FS32 --> GetHP32[Get the hyperparameters]
    GetHP32 --> FS32
    GetHP32 --> FS33[Feature Selection Embedded Methods]
    FS33 --> FS32
    FS33 --> Retrain33[Retrain]
    Retrain33 --> FS33
    FS33 --> GetHP33[Get the hyperparameters]
    GetHP33 --> FS33
    GetHP33 --> FS34[Feature Selection Embedded Methods]
    FS34 --> FS33
    FS34 --> Retrain34[Retrain]
    Retrain34 --> FS34
    FS34 --> GetHP34[Get the hyperparameters]
    GetHP34 --> FS34
    GetHP34 --> FS35[Feature Selection Embedded Methods]
    FS35 --> FS34
    FS35 --> Retrain35[Retrain]
    Retrain35 --> FS35
    FS35 --> GetHP35[Get the hyperparameters]
    GetHP35 --> FS35
    GetHP35 --> FS36[Feature Selection Embedded Methods]
    FS36 --> FS35
    FS36 --> Retrain36[Retrain]
    Retrain36 --> FS36
    FS36 --> GetHP36[Get the hyperparameters]
    GetHP36 --> FS36
    GetHP36 --> FS37[Feature Selection Embedded Methods]
    FS37 --> FS36
    FS37 --> Retrain37[Retrain]
    Retrain37 --> FS37
    FS37 --> GetHP37[Get the hyperparameters]
    GetHP37 --> FS37
    GetHP37 --> FS38[Feature Selection Embedded Methods]
    FS38 --> FS37
    FS38 --> Retrain38[Retrain]
    Retrain38 --> FS38
    FS38 --> GetHP38[Get the hyperparameters]
    GetHP38 --> FS38
    GetHP38 --> FS39[Feature Selection Embedded Methods]
    FS39 --> FS38
    FS39 --> Retrain39[Retrain]
    Retrain39 --> FS39
    FS39 --> GetHP39[Get the hyperparameters]
    GetHP3
```



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



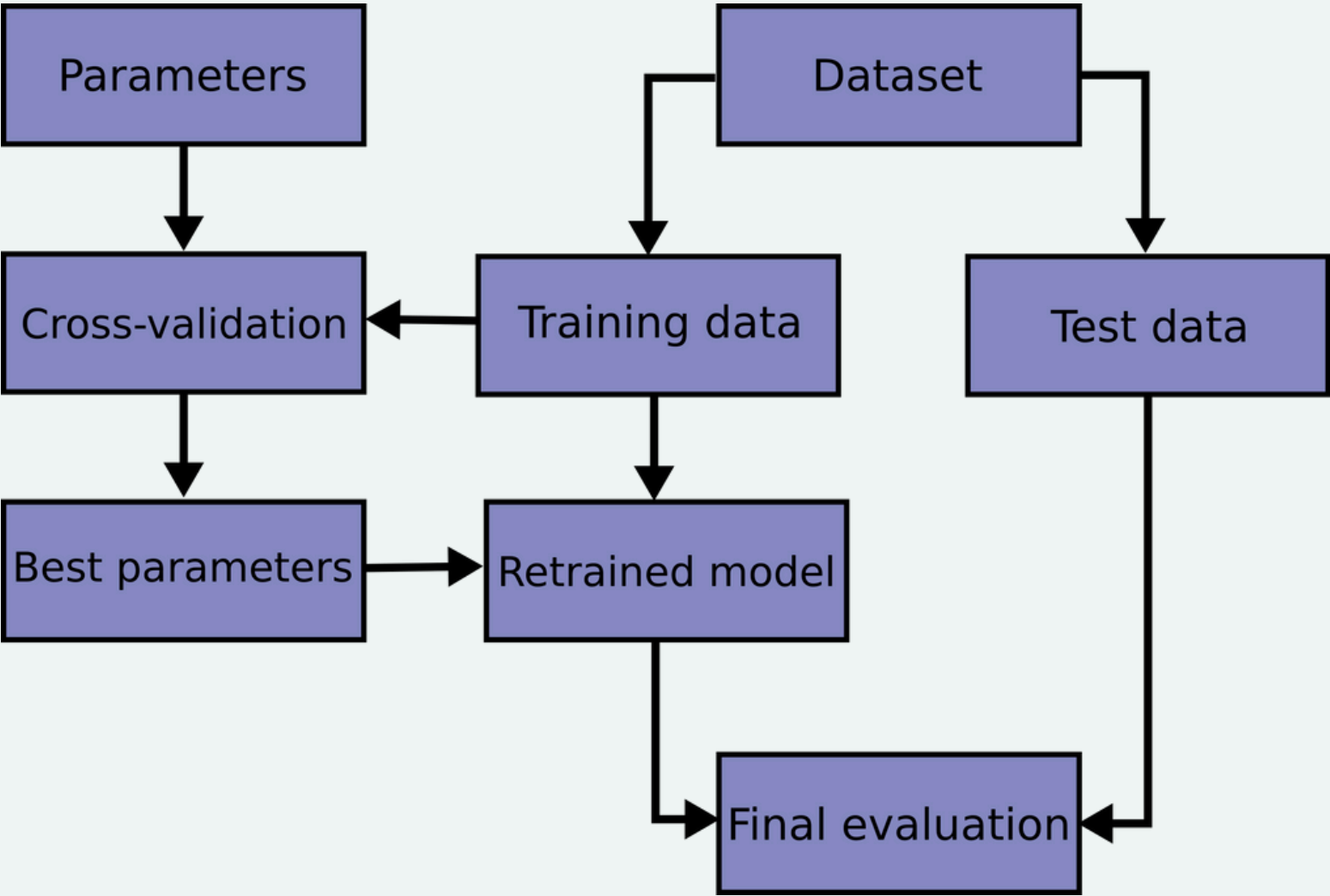
1) Train and tune the hyperparameters of the models: kNN, Decision Tree, Random Forest, and SVC

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

(3) Train multilayer perceptrons

(4) Train with ensemble 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

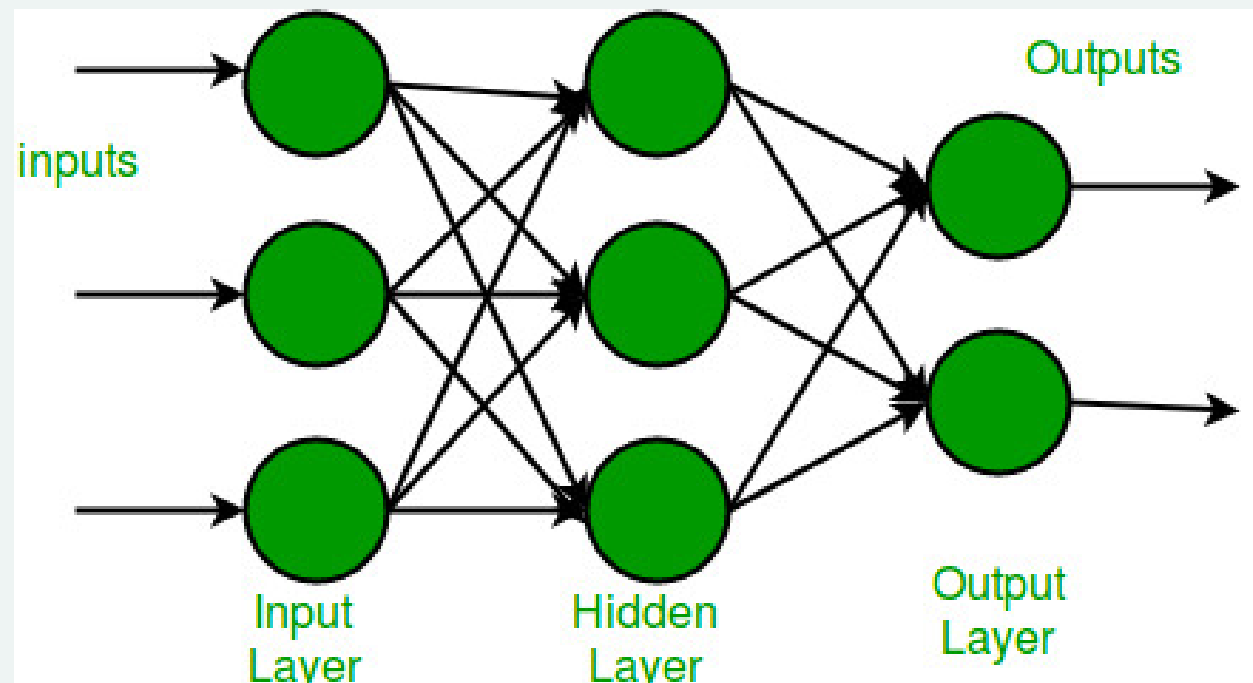
We using 2 main methods.

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.

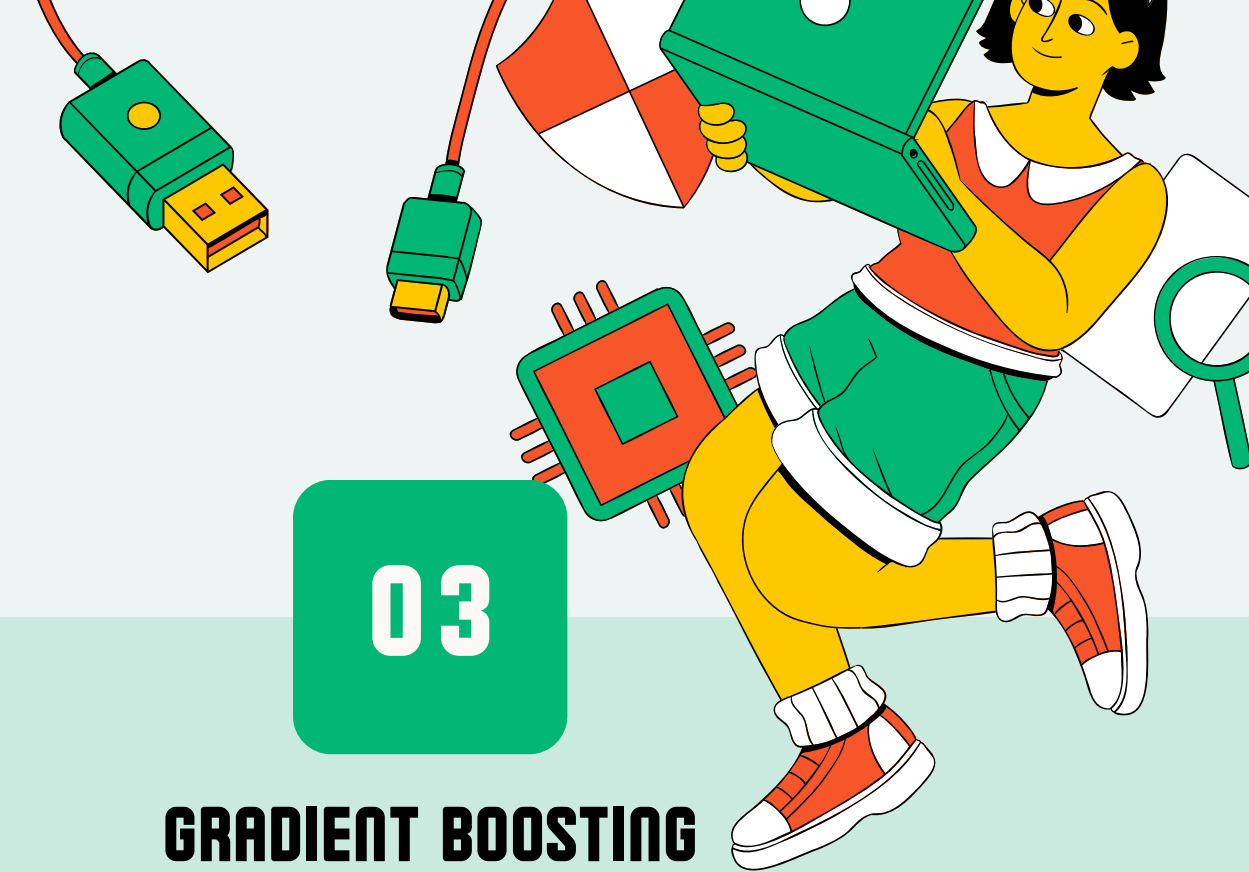
After the hyperparameter tuning process using GridSearch with Stratified K-Fold, we obtain the hyperparameters for AdaBoost as `n_estimators = 500` and `learning_rate = 1`.

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

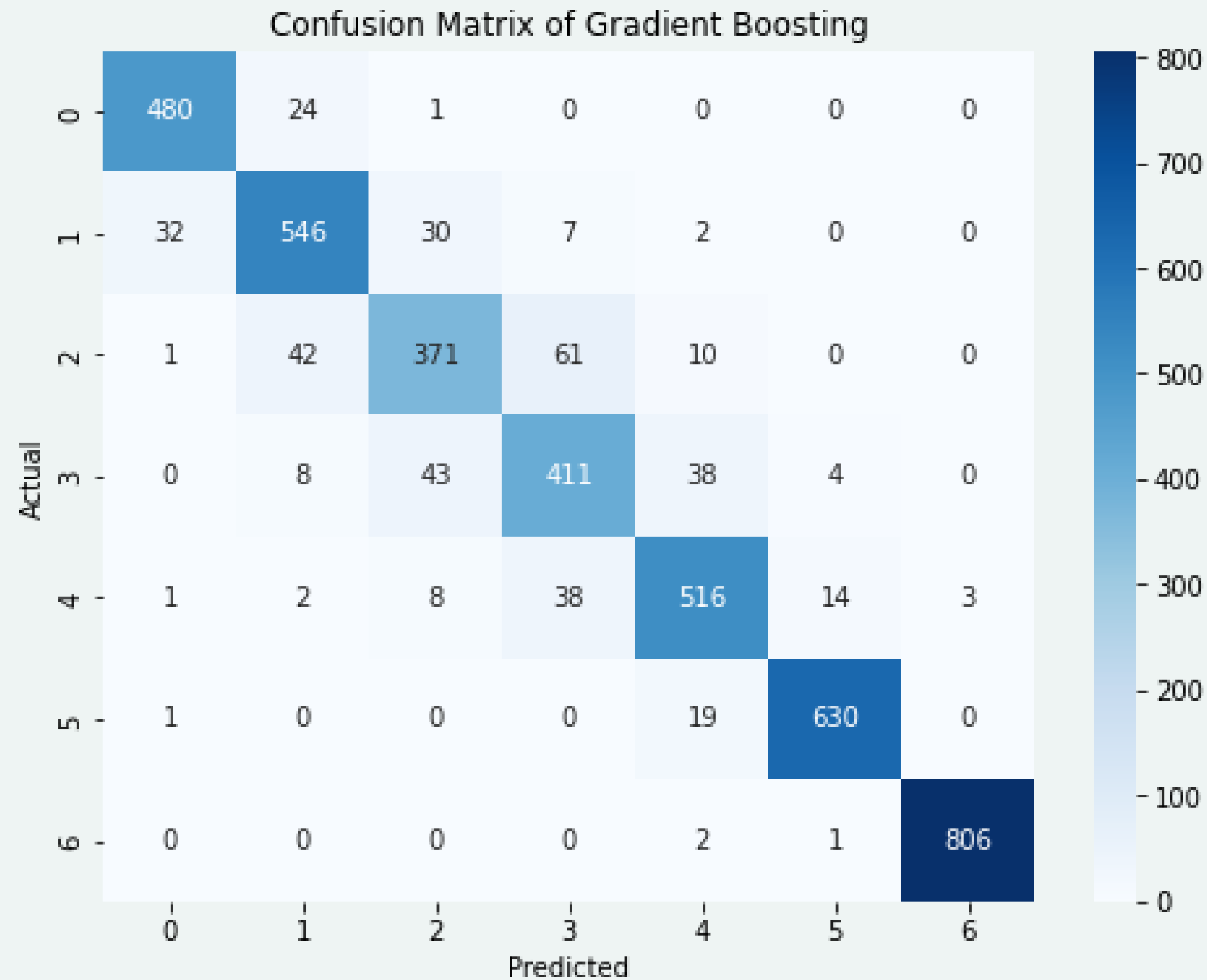
Model	Feature Selection			Macro F1 (%)	Weighted F1 (%)	Accuracy (%)
	Decision Tree	Random Forest	SFS			
Decision Tree	-	-	-	86.78	88.00	88.03
	yes	-	-	86.40	87.66	87.67
	-	yes	-	86.12	87.43	87.48
	-	-	yes	86.14	87.34	87.40
Random Forest	-	-	-	89.28	90.31	90.37
	yes	-	-	88.79	89.88	89.93
	-	yes	-	88.36	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.

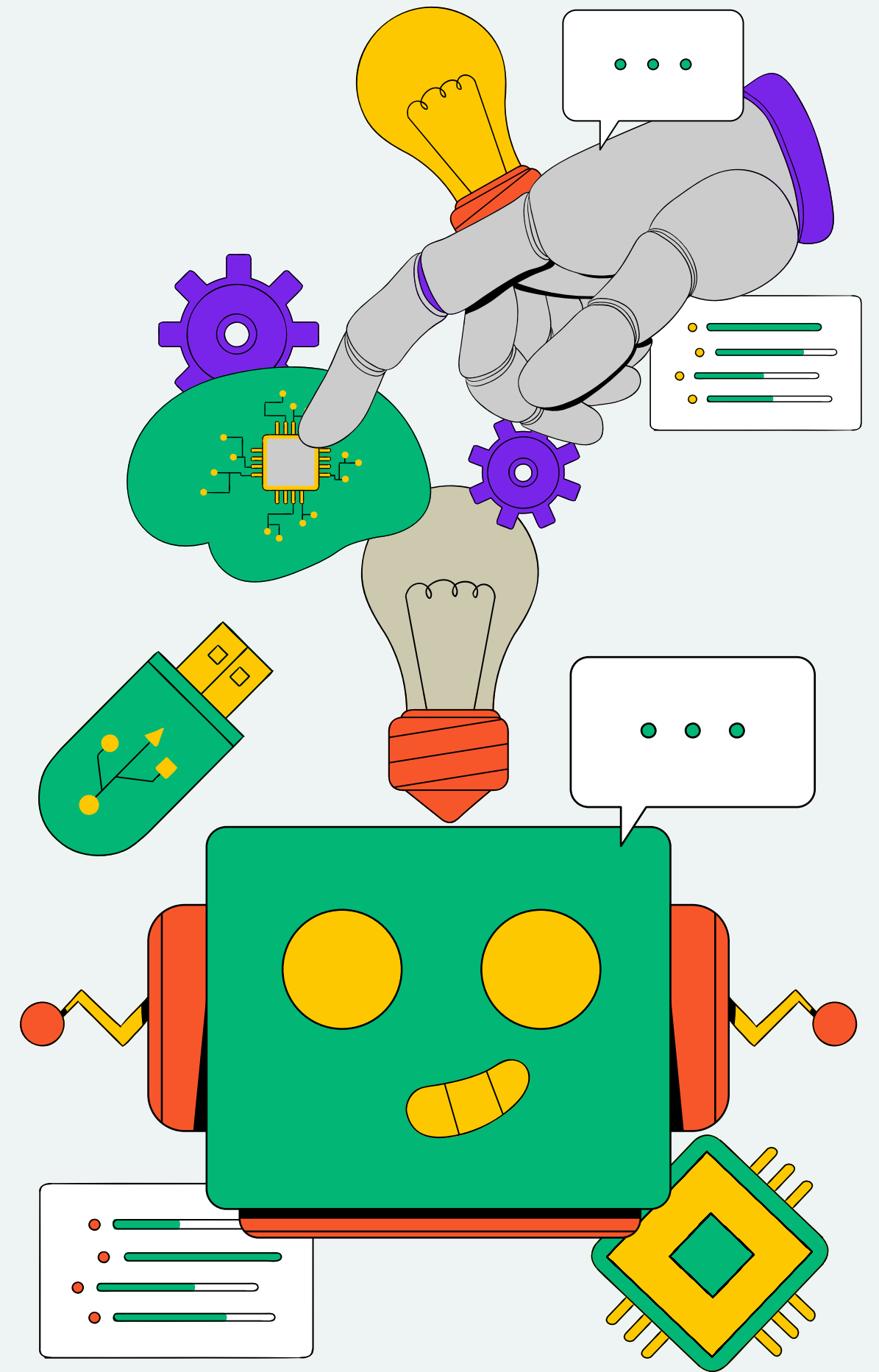


CONFUSION MATRIX



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

