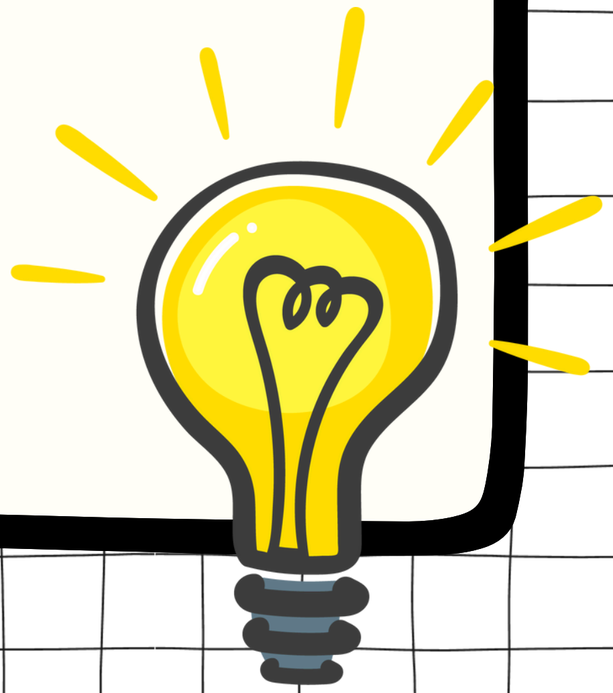
Machine Learning and Data Mining

Capstone Project

Obesity Level Classification



Instructor: Assoc. Prof. Than Quang Khoat

Class: 147828

Group: 9

Team member

Nguyen Chi Long	20210553
Ngo Xuan Bach	20215181
Le Xuan Hieu	20215201
Dinh Viet Quang	20215235
Nguyen Viet Thuan	20210826

Contents

01

Introduction

Introduction

- What is our project about?
- What is the date of today?
- How are you?
- What is your daily routine?
- How do you know about obesity?
- What is problem formulation?

obesity

noun [U]

UK /əʊˈbiː.sə.ti/ US /oʊˈbiː.sə.ţi/

Add to word list

C1

the fact of being extremely fat, in a way that is dangerous for health:

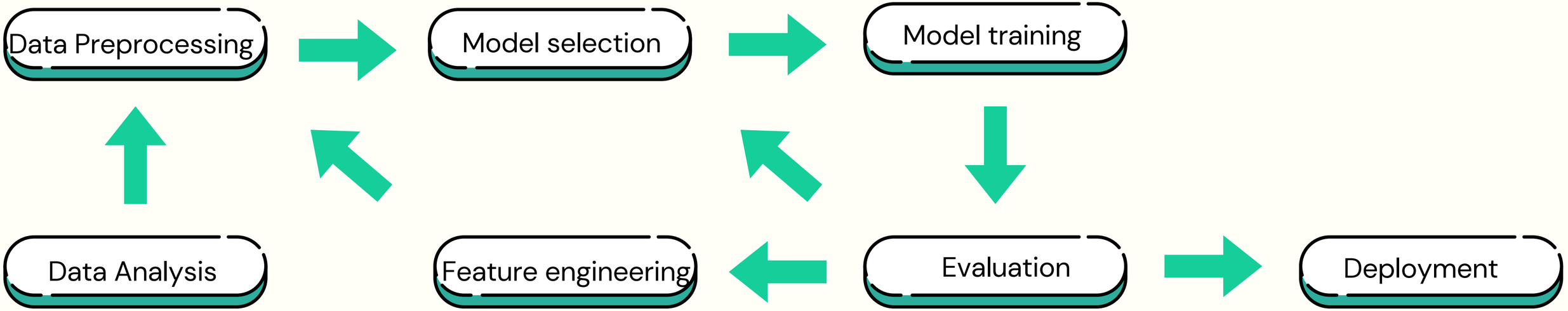
- *The National Institute of Health is discussing ways of tackling the problem of **childhood obesity.***
- *A diet that is high in fat and sugar can lead to obesity.*

- Cambridge dictionary -

"Rates of overweight and obesity continue to grow in adults and children. From 1990 to 2022, the percentage of children and adolescents aged 5–19 years living with obesity increased four-fold from 2% to 8% globally, while the percentage of adults 18 years of age and older living with obesity more than doubled from 7% to 16%."

- World Health Organization (WHO) -

Strategy



The main workflow of our project

02

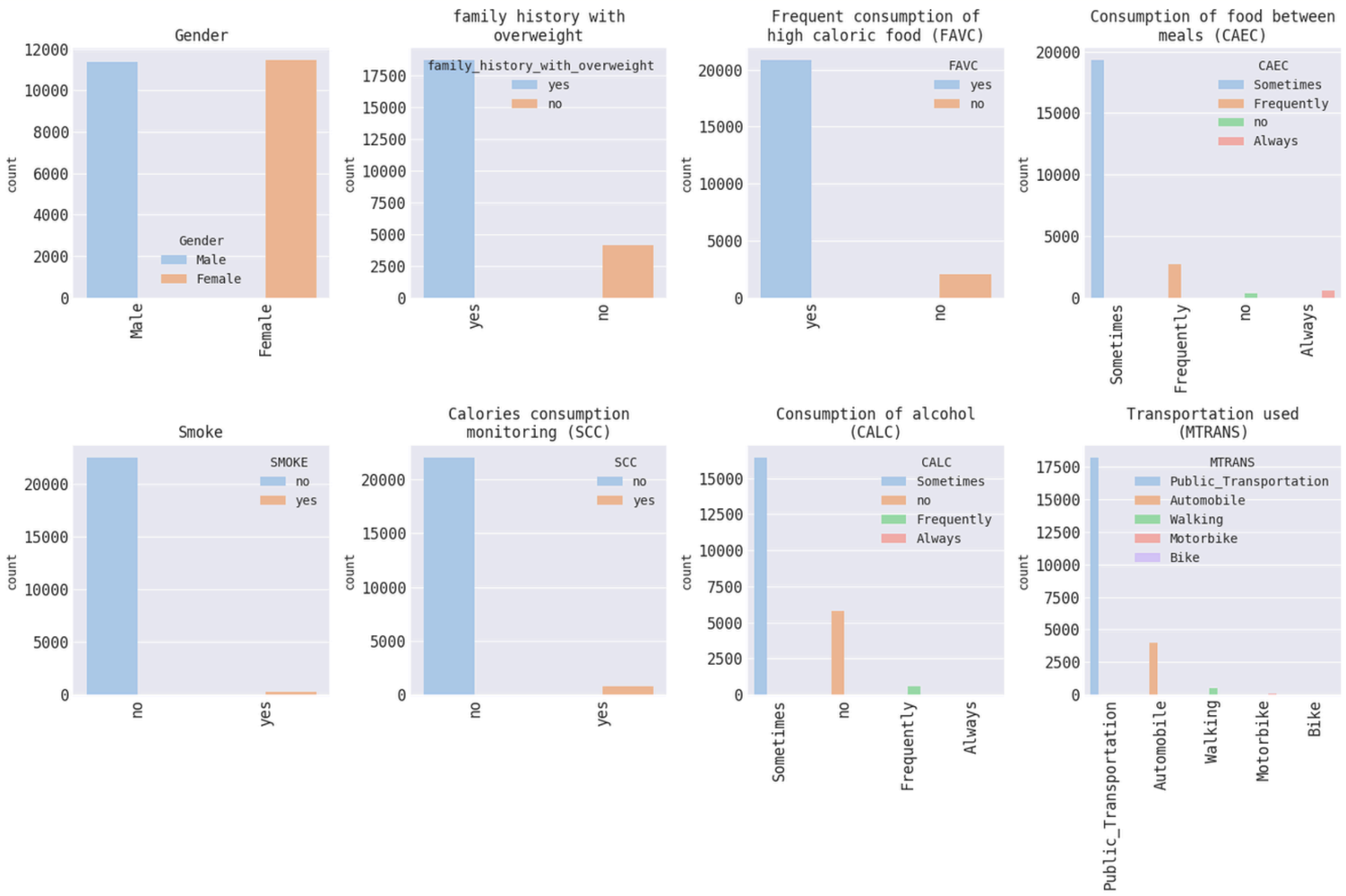
Dataset

Dataset

Column Name	Description
Gender	Person's Gender
Age	Years old
Height	Height (meters)
Weight	Weight (Kilograms)
family_history_with_overweight	Yes/no question, to see if person has family history with overweight
FAVC	'Frequent consumption of high caloric food' - yes/no question, to see if person eats high caloric food frequently
FCVC	'Frequency of consumption of vegetables' Similar to FAVC but with vegetables (frequency)
NCP	'Number of main meals' NCP's value is 1, 2, 3, 4
CAEC	'Consumption of food between meals' takes 4 values (Always, Frequently, Sometimes, no)
SMOKE	"Do you smoke?", yes/no question
CH2O	'Consumption of water daily' CH2O's values should be 1,2 or 3
SCC	'Calories consumption monitoring': "Do you monitor your calories consumption?", yes/no question
FAF	'Physical activity frequency', ranged from 0 to 3 (float), where 0 means no physical activity and 3 means frequent activity
TUE	'Time using technology devices', such as smartphone or laptop - ranged from 0 to 2, where 0 means no devices used while 2 means high frequency of devices use
CALC	'Consumption of alcohol' takes 4 values: Always, Frequently, Sometimes, no
MTRANS	'Transportation used' MTRANS takes 5 values Public_Transportation, Automobile, Walking, Motorbike, & Bike
NObeyesdad	This is our target value, takes 7 values

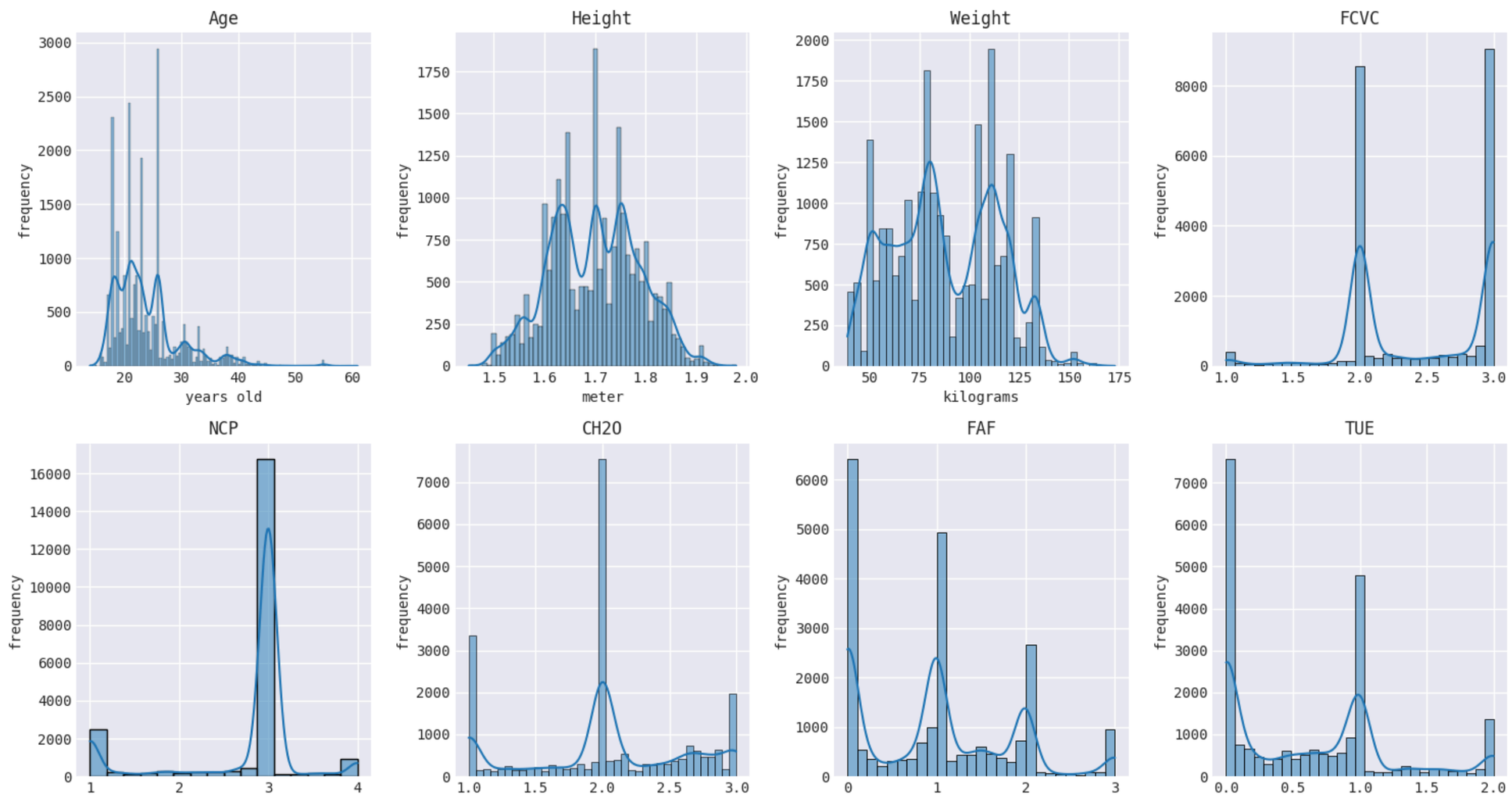
- 22869 records.
- Consist of 17 categorical and numerical features.

Some Data Exploration



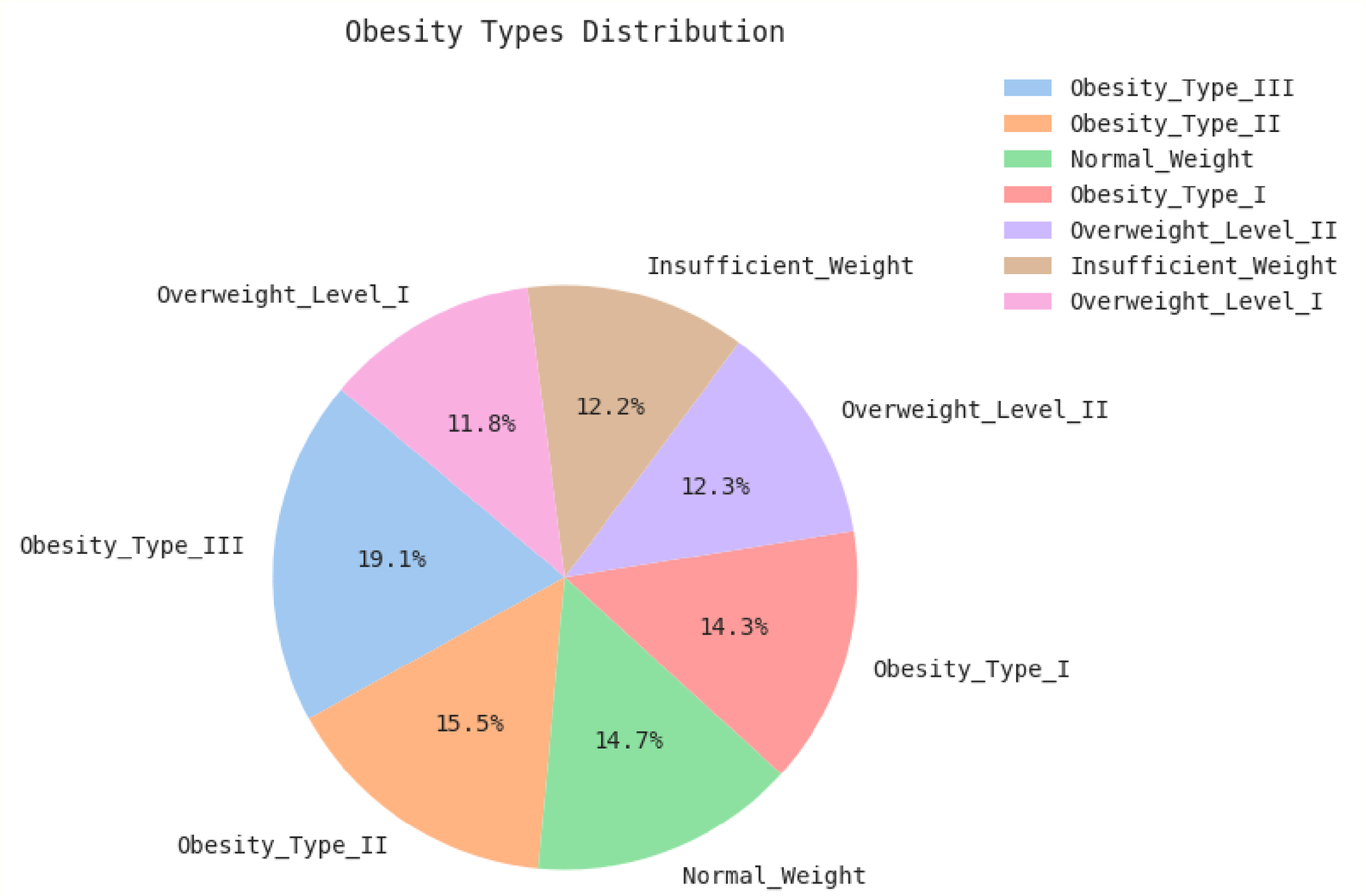
Distribution of categorical features

Some Data Exploration



Distribution of numerical features

Some Data Exploration



Some Data Exploration

03

Experiments

Data preprocessing

Preprocessing

One-hot encoder

['Gender', 'family_history_with_overweight', 'FAVC', 'SMOKE', 'SCC', 'MTRANS']

Ordinal encoder

['CAEC', 'CALC']

Standard Scaler

Numerical column: ['Age', 'Height', 'Weight"]

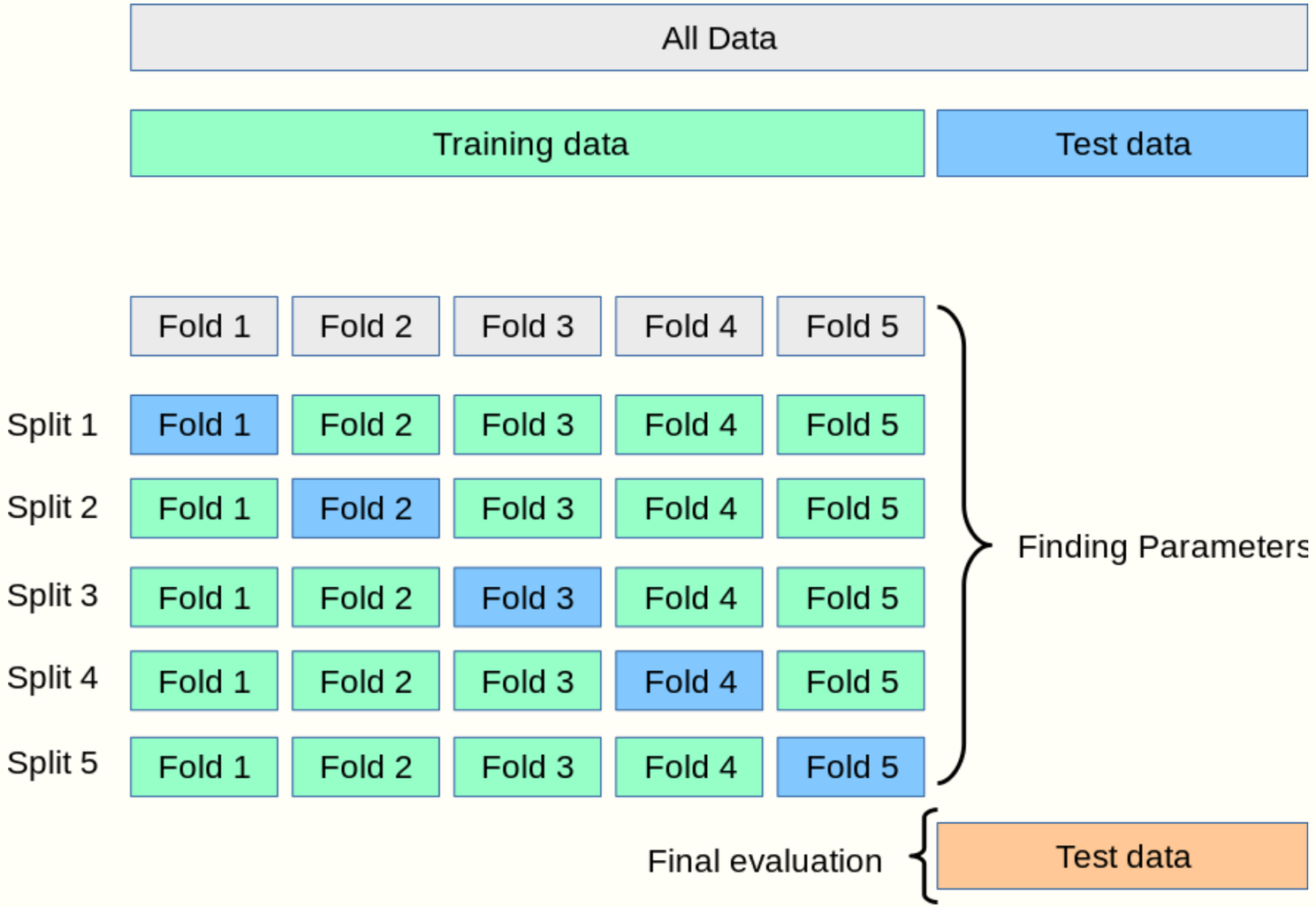
Boxcox

['Age']

Discrete values transform

FCVC, NCP, CH2O, FAF, TUE

Model evaluation



Model selection

K-nearest neighbors

- **n_neighbors (K)**: [2, 5, 8, 10, 20]
- **p (parameter of Minkowski distance):** [1, 2]

params	mean_fit_time	mean_score_time	mean_test_score	mean_train_score
{'p': 1, 'n_neighbors': 20}	0.0117	3.4113	0.8297	0.8484
{'p': 1, 'n_neighbors': 10}	0.0115	3.3076	0.8280	0.8627
{'p': 1, 'n_neighbors': 8}	0.0114	3.3052	0.8272	0.8660
{'p': 1, 'n_neighbors': 5}	0.0127	3.3420	0.8193	0.8773
{'p': 2, 'n_neighbors': 10}	0.0114	0.8605	0.8025	0.8390
{'p': 2, 'n_neighbors': 8}	0.0137	0.9487	0.7998	0.8466
{'p': 2, 'n_neighbors': 20}	0.0116	0.8811	0.7968	0.8197
{'p': 2, 'n_neighbors': 5}	0.0117	0.8607	0.7956	0.8613
{'p': 1, 'n_neighbors': 2}	0.0161	3.4240	0.7790	0.8966
{'p': 2, 'n_neighbors': 2}	0.0113	0.8641	0.7642	0.8891

Model selection

Logistic regression

- **C**: the inverse of regularization parameter. C(C): [1e-1, 1, 10, 100, 1000]

params	mean_fit_time	mean_score_time	mean_test_score	mean_train_score
{'penalty': 'l2', 'C': 10}	15.6687	0.0051	0.8700	0.8720
{'penalty': 'l2', 'C': 100}	31.1332	0.0053	0.8700	0.8725
{'penalty': 'l2', 'C': 1000}	35.6433	0.0054	0.8699	0.8725
{'penalty': None}	37.5202	0.0043	0.8697	0.8725
{'penalty': 'l2', 'C': 1}	7.0759	0.0053	0.8656	0.8677
{'penalty': 'l2', 'C': 0.1}	4.4298	0.0052	0.8350	0.8369

Model selection

Decision tree

- **max_depth**: [5, 10, 15, 20]
- **min_samples_leaf** : [5, 10, 20, 30, 50]
- **criterion**: [’gini’, ‘entropy’]

params	mean_fit_time	mean_score_time	mean_test_score	mean_train_score
{'min_samples_leaf': 10, 'max_depth': 10, 'criterion': 'entropy'}	0.1437	0.0051	0.8831	0.9071
{'min_samples_leaf': 20, 'max_depth': 10, 'criterion': 'entropy'}	0.1486	0.0055	0.8823	0.8976
{'min_samples_leaf': 5, 'max_depth': 10, 'criterion': 'entropy'}	0.1459	0.0051	0.8820	0.9144
{'min_samples_leaf': 30, 'max_depth': 15, 'criterion': 'entropy'}	0.1273	0.0045	0.8819	0.8909
{'min_samples_leaf': 20, 'max_depth': 20, 'criterion': 'entropy'}	0.1650	0.0082	0.8817	0.8986

Top 5 best parameter setting

{'min_samples_leaf': 50, 'max_depth': 10, 'criterion': 'gini'}	0.1010	0.0049	0.8728	0.8800
{'min_samples_leaf': 50, 'max_depth': 20, 'criterion': 'gini'}	0.1588	0.0074	0.8728	0.8800
{'min_samples_leaf': 5, 'max_depth': 5, 'criterion': 'gini'}	0.0691	0.0051	0.8435	0.8460
{'min_samples_leaf': 50, 'max_depth': 5, 'criterion': 'gini'}	0.0674	0.0049	0.8417	0.8436
{'min_samples_leaf': 50, 'max_depth': 5, 'criterion': 'entropy'}	0.0864	0.0050	0.8307	0.8345

Top 5 worst parameter setting

Model selection

Random forest

- **n_estimators:** [100, 300, 500, 1000, 2000]
- **max_depth:** [10, 20, 30]
- **min_samples_leaf':** [10, 20, 30]
- **criterion:** ['gini', 'entropy']

params	mean_fit_time	mean_score_time	mean_test_score	mean_train_score
{'n_estimators': 500, 'min_samples_leaf': 10, 'max_depth': 20, 'criterion': 'entropy'}	14.4901	0.4004	0.8965	0.9222
{'n_estimators': 1000, 'min_samples_leaf': 10, 'max_depth': 20, 'criterion': 'gini'}	24.5074	0.8482	0.8956	0.9216
{'n_estimators': 2000, 'min_samples_leaf': 10, 'max_depth': 10, 'criterion': 'entropy'}	56.5501	1.4682	0.8917	0.9141
{'n_estimators': 2000, 'min_samples_leaf': 10, 'max_depth': 10, 'criterion': 'gini'}	44.7706	1.5665	0.8904	0.9117
{'n_estimators': 2000, 'min_samples_leaf': 20, 'max_depth': 30, 'criterion': 'gini'}	44.7748	1.5418	0.8882	0.9064

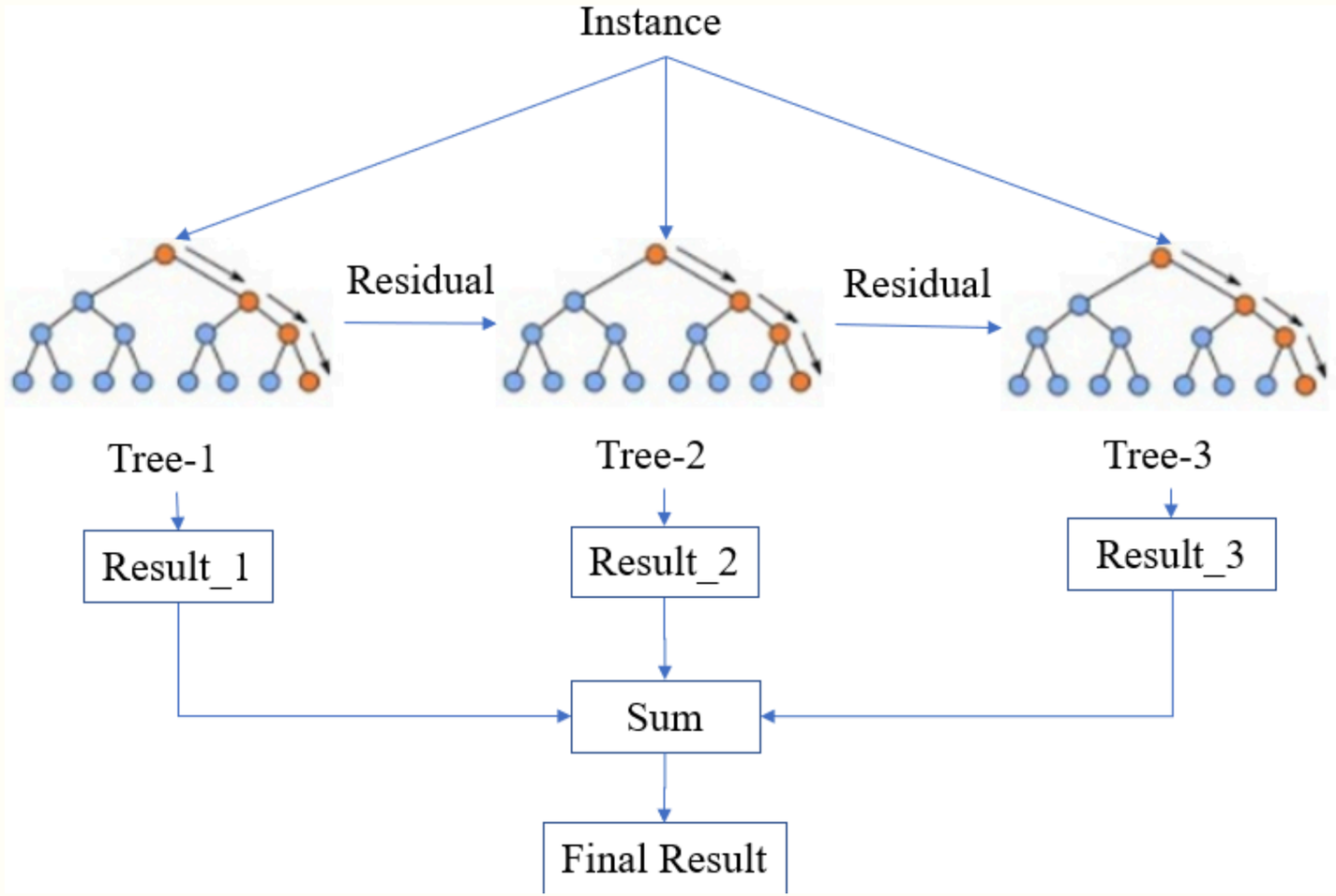
Top 5 best parameter setting

{'n_estimators': 500, 'min_samples_leaf': 30, 'max_depth': 10, 'criterion': 'gini'}	10.4965	0.3669	0.8795	0.8924
{'n_estimators': 300, 'min_samples_leaf': 30, 'max_depth': 10, 'criterion': 'entropy'}	8.0571	0.2142	0.8794	0.8935
{'n_estimators': 100, 'min_samples_leaf': 30, 'max_depth': 10, 'criterion': 'gini'}	2.0512	0.0782	0.8783	0.8904
{'n_estimators': 100, 'min_samples_leaf': 30, 'max_depth': 20, 'criterion': 'entropy'}	2.6312	0.0965	0.8777	0.8933
{'n_estimators': 100, 'min_samples_leaf': 30, 'max_depth': 10, 'criterion': 'entropy'}	2.6146	0.0756	0.8769	0.8914

Top 5 worst parameter setting

Model selection

XGBoost



Model selection

XGBoost

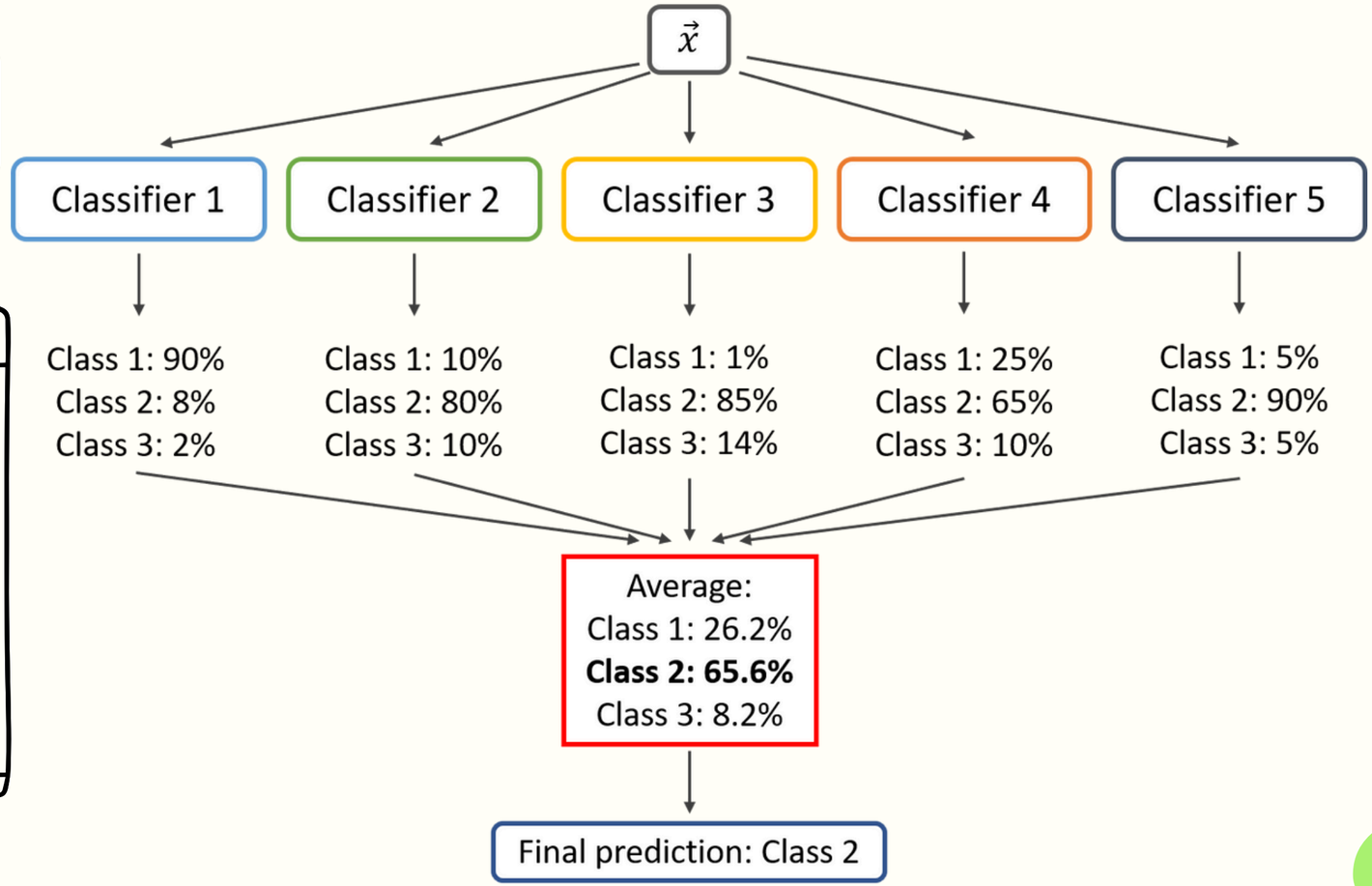
- **n_estimators:** [100, 300, 500, 1000, 2000]
- **max_depth:** [10, 20, 30]

params	mean_fit_time	mean_score_time	mean_test_score	mean_train_score
{'n_estimators': 500, 'max_depth': 15}	24.1672	0.7437	0.9057	1.0000
{'n_estimators': 300, 'max_depth': 15}	16.4752	0.5553	0.9055	1.0000
{'n_estimators': 100, 'max_depth': 10}	6.3175	0.1774	0.9051	0.9999
{'n_estimators': 200, 'max_depth': 15}	12.2354	0.3745	0.9051	1.0000
{'n_estimators': 100, 'max_depth': 15}	7.1653	0.2139	0.9048	1.0000
{'n_estimators': 500, 'max_depth': 20}	22.8552	0.6701	0.9047	1.0000
{'n_estimators': 300, 'max_depth': 20}	16.3073	0.5154	0.9046	1.0000
{'n_estimators': 300, 'max_depth': 10}	15.9453	0.4860	0.9043	1.0000
{'n_estimators': 200, 'max_depth': 20}	12.4350	0.3785	0.9041	1.0000
{'n_estimators': 100, 'max_depth': 20}	7.5500	0.2092	0.9037	1.0000
{'n_estimators': 200, 'max_depth': 10}	11.2365	0.3470	0.9035	1.0000
{'n_estimators': 500, 'max_depth': 10}	24.2957	0.7151	0.9029	1.0000

Model selection

Voting classifier

- **XGBClassifier**(n_estimators = 500, max_depth = 15)
- **RandomForestClassifier(**n_estimators = 500, min_samples_leaf = 10, max_depth = 20, criterion = entropy);
- **XGBClassifier**(n_estimators = 100, max_depth = 10)



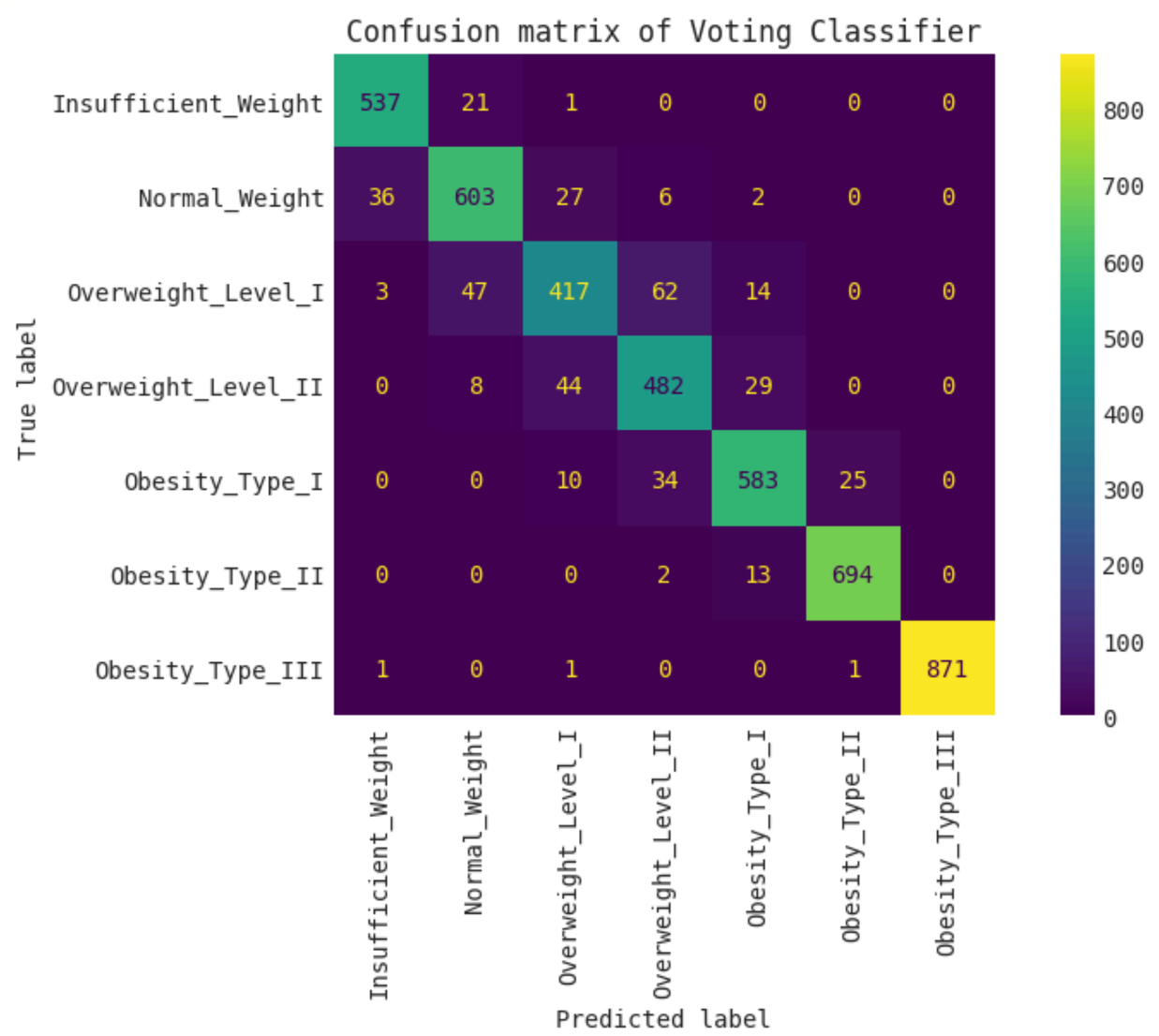
04

Results Evaluation

Cross-validation result

params	mean_fit_time	mean_score_time	mean_test_score	mean_train_score	model_name
{'p': 1, 'n_neighbors': 20}	0.0117	3.4113	0.8297	0.8484	knn
{'penalty': 'l2', 'C': 10}	15.6687	0.0051	0.8700	0.8720	logistic_regression
{'min_samples_leaf': 10, 'max_depth': 10, 'criterion': 'entropy'}	0.1437	0.0051	0.8831	0.9071	decision_tree
{'n_estimators': 500, 'min_samples_leaf': 10, 'max_depth': 20, 'criterion': 'entropy'}	14.4901	0.4004	0.8965	0.9222	random_forest
{'n_estimators': 500, 'max_depth': 15}	24.1672	0.7437	0.9057	1.0000	xg_boost
XGBClassifier(n_estimators = 500, max_depth = 15); RandomForestClassifier(n_estimators = 500, min_samples_leaf = 10, max_depth = 20, criterion = entropy); XGBClassifier(n_estimators = 100, max_depth = 10)	43.9071	1.2480	0.9072	1.0000	voting_classifier
{'penalty': None}	75.1169	0.0103	0.8754	0.8782	logistic_regression_featenge

Evaluation on test set



Voting Classifier

- Accuracy: 0.9154
- F1_score: 0.9149
- Precision_score: 0.9150
- Recall_score: 0.9154

05 Conclusions

01

Difficulties

- The effective way to preprocessing the data is not an easy task.
- Limited understanding of model relating to theoretical properties.

02

Summary

- By using various methods and tools, we have trained five different models on our dataset.
- **The best model among all is Voting Classifier.**

03

Possible extensions

- Finding better training strategy for each models.
- Research and implement new models such as DNN or LightGBM.
- Application further deployment for better user experience.

Thank you

Machine Learning and Data Mining
Group 9