DSV Virupaksha vegi

Week2_Lab2

EB25
E21CSEU0697

In [22]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [23]:

```
data = pd.read_csv("/Users/ObesityData.csv") data.head()
```

Out[23]:

	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	NCP	CAEC	SMOKE	CH2O	SCC	FAF	TUE	CALC	
0	Female	21.0	1.62	64.0	yes	no	2.0	3.0	Sometimes	no	2.0	no	0.0	1.0	no	P
1	Female	21.0	1.52	56.0	yes	no	3.0	3.0	Sometimes	yes	3.0	yes	3.0	0.0	Sometimes	P
2	Male	23.0	1.80	77.0	yes	no	2.0	3.0	Sometimes	no	2.0	no	2.0	1.0	Frequently	P
3	Male	27.0	1.80	87.0	no	no	3.0	3.0	Sometimes	no	2.0	no	2.0	0.0	Frequently	
4	Male	22.0	1.78	89.8	no	no	2.0	1.0	Sometimes	no	2.0	no	0.0	0.0	Sometimes	P

In [24]:

```
data.insert(0, "intercept", 1)
```

In [25]:

```
data['Gender'] = data['Gender'].replace(['Female'], 1)
data['Gender'] = data['Gender'].replace(['Male'], 0)
data
```

Out[25]:

	intercept	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	NCP	CAEC	SMOKE	CH2O	SC
0	1	1	21.000000	1.620000	64.000000	yes	no	2.0	3.0	Sometimes	no	2.000000	n
1	1	1	21.000000	1.520000	56.000000	yes	no	3.0	3.0	Sometimes	yes	3.000000	ye
2	1	0	23.000000	1.800000	77.000000	yes	no	2.0	3.0	Sometimes	no	2.000000	n
3	1	0	27.000000	1.800000	87.000000	no	no	3.0	3.0	Sometimes	no	2.000000	n
4	1	0	22.000000	1.780000	89.800000	no	no	2.0	1.0	Sometimes	no	2.000000	n
...	...	...	...	...	...	...	...	...	...	...	...	...	.
2106	1	1	20.976842	1.710730	131.408528	yes	yes	3.0	3.0	Sometimes	no	1.728139	n
2107	1	1	21.982942	1.748584	133.742943	yes	yes	3.0	3.0	Sometimes	no	2.005130	n
2108	1	1	22.524036	1.752206	133.689352	yes	yes	3.0	3.0	Sometimes	no	2.054193	n
2109	1	1	24.361936	1.739450	133.346641	yes	yes	3.0	3.0	Sometimes	no	2.852339	n
2110	1	1	23.664709	1.738836	133.472641	yes	yes	3.0	3.0	Sometimes	no	2.863513	n

2111 rows × 18 columns

In [26]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data["FAVC"] = le.fit_transform(data["FAVC"])
data["SMOKE"] = le.fit_transform(data["SMOKE"])
data["SCC"] = le.fit_transform(data["SCC"])
data["family_history_with_overweight"] = le.fit_transform(data["family_history_with_overweight"])
data["NObeyesdad"] = le.fit_transform(data["NObeyesdad"])
data["MTRANS"] = le.fit_transform(data["MTRANS"])
data["CAEC"] = le.fit_transform(data["CAEC"])
data["CALC"] = le.fit_transform(data["CALC"])
data
```

Out[26]:

	intercept	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	NCP	CAEC	SMOKE	CH2O	SCC	
0	1	1	21.000000	1.620000	64.000000	1	0	2.0	3.0	2	0	2.000000	0	0
1	1	1	21.000000	1.520000	56.000000	1	0	3.0	3.0	2	1	3.000000	1	3
2	1	0	23.000000	1.800000	77.000000	1	0	2.0	3.0	2	0	2.000000	0	2
3	1	0	27.000000	1.800000	87.000000	0	0	3.0	3.0	2	0	2.000000	0	2
4	1	0	22.000000	1.780000	89.800000	0	0	2.0	1.0	2	0	2.000000	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
2106	1	1	20.976842	1.710730	131.408528	1	1	3.0	3.0	2	0	1.728139	0	1
2107	1	1	21.982942	1.748584	133.742943	1	1	3.0	3.0	2	0	2.005130	0	1
2108	1	1	22.524036	1.752206	133.689352	1	1	3.0	3.0	2	0	2.054193	0	1
2109	1	1	24.361936	1.739450	133.346641	1	1	3.0	3.0	2	0	2.852339	0	1
2110	1	1	23.664709	1.738836	133.472641	1	1	3.0	3.0	2	0	2.863513	0	1

2111 rows × 18 columns

In [28]:

```
from sklearn.model_selection import train_test_split
x1 = data.drop(columns=["NObeyesdad"])
y1 = data["NObeyesdad"]
x_train, x_test, y_train, y_test = train_test_split(x1, y1, test_size = 0.4)
```

In [29]:

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression(multi_class = "auto")

model.fit(x_train, y_train)
y_pred = model.predict(x_test)
```

```
/opt/anaconda3/lib/python3.9/site-packages/sklearn/linear_model/_logistic.py:814: ConvergenceWarning: lbf
gs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/p
reprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.o
rg/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

In [31]:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred)
```

Out[31]:

```
0.6603550295857988
```