Importing relevant Libraries

In [91]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Display whole dataset
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)

# Import data from local files
obesity_dt = pd.read_excel(r"C:\Users\pavilion14\Downloads\ObesityDataSet_raw_and_d

# Display the first 10 rows of the dataset

obesity_dt.head(10)
```

Out[91]:

	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	NCP	CAEC
0	Female	21.0	1.62	64.0	yes	no	2.0	3.0	Sometimes
1	Female	21.0	1.52	56.0	yes	no	3.0	3.0	Sometimes
2	Male	23.0	1.80	77.0	yes	no	2.0	3.0	Sometimes
3	Male	27.0	1.80	87.0	no	no	3.0	3.0	Sometimes
4	Male	22.0	1.78	89.8	no	no	2.0	1.0	Sometimes
5	Male	29.0	1.62	53.0	no	yes	2.0	3.0	Sometimes
6	Female	23.0	1.50	55.0	yes	yes	3.0	3.0	Sometimes
7	Male	22.0	1.64	53.0	no	no	2.0	3.0	Sometimes
8	Male	24.0	1.78	64.0	yes	yes	3.0	3.0	Sometimes
9	Male	22.0	1.72	68.0	yes	yes	2.0	3.0	Sometimes

Sanity Check On Data

In [92]:
```
obesity_dt.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2111 entries, 0 to 2110
Data columns (total 17 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   Gender                          2111 non-null   object 
 1   Age                             2111 non-null   float64
 2   Height                          2111 non-null   float64
 3   Weight                          2111 non-null   float64
 4   family_history_with_overweight  2111 non-null   object 
 5   FAVC                            2111 non-null   object 
 6   FCVC                            2111 non-null   float64
 7   NCP                             2111 non-null   float64
 8   CAEC                            2111 non-null   object 
 9   SMOKE                           2111 non-null   object 
 10  CH2O                            2111 non-null   float64
 11  SCC                             2111 non-null   object 
 12  FAF                             2111 non-null   float64
 13  TUE                             2111 non-null   float64
 14  CALC                            2111 non-null   object 
 15  MTRANS                          2111 non-null   object 
 16  NObeyesdad                      2111 non-null   object 
dtypes: float64(8), object(9)
memory usage: 280.5+ KB
```

In [93]:

```
obesity_dt.shape
```

Out[93]:

```
(2111, 17)
```

In [94]:

```
# Checking for missing values in each variable
Missing_values = obesity_dt.isna().any()

# Checking for duplicate data
num_duplicates = obesity_dt.duplicated().sum()


print(Missing_values)
print(f'Duplicates total {num_duplicates}')
```

```
Gender                            False
Age                               False
Height                            False
Weight                            False
family_history_with_overweight    False
FAVC                              False
FCVC                              False
NCP                               False
CAEC                              False
SMOKE                             False
CH2O                              False
SCC                               False
FAF                               False
TUE                               False
CALC                              False
MTRANS                            False
NObeyesdad                        False
dtype: bool
Duplicates total 24
```

In [95]:

```
# Display duplicate rows
Duplicated_rows  = obesity_dt[obesity_dt.duplicated()]

# Drop duplicate values
obesity_dt.drop_duplicates(inplace = True)
```

```
duplicate = obesity_dt.duplicated().sum()
print(f"The Total duplicate is:{duplicate}")
```

```
The Total duplicate is:0
```

In [96]:

```
#Checking for garbage values
for i in obesity_dt.select_dtypes(include = 'object').columns:
    print(obesity_dt[i].value_counts())
    print('***'*10)
```

```
Gender
Male      1052
Female    1035
Name: count, dtype: int64
******************************
family_history_with_overweight
yes    1722
no      365
Name: count, dtype: int64
******************************
FAVC
yes    1844
no      243
Name: count, dtype: int64
******************************
CAEC
Sometimes     1761
Frequently     236
Always          53
no              37
Name: count, dtype: int64
******************************
SMOKE
no     2043
yes      44
Name: count, dtype: int64
******************************
SCC
no     1991
yes      96
Name: count, dtype: int64
******************************
CALC
Sometimes     1380
no             636
Frequently      70
Always           1
Name: count, dtype: int64
******************************
MTRANS
Public_Transportation    1558
Automobile                456
Walking                    55
Motorbike                  11
Bike                        7
Name: count, dtype: int64
******************************
NObeyesdad
Obesity_Type_I         351
Obesity_Type_III       324
Obesity_Type_II        297
Overweight_Level_II    290
Normal_Weight          282
Overweight_Level_I     276
Insufficient_Weight    267
Name: count, dtype: int64
******************************
```

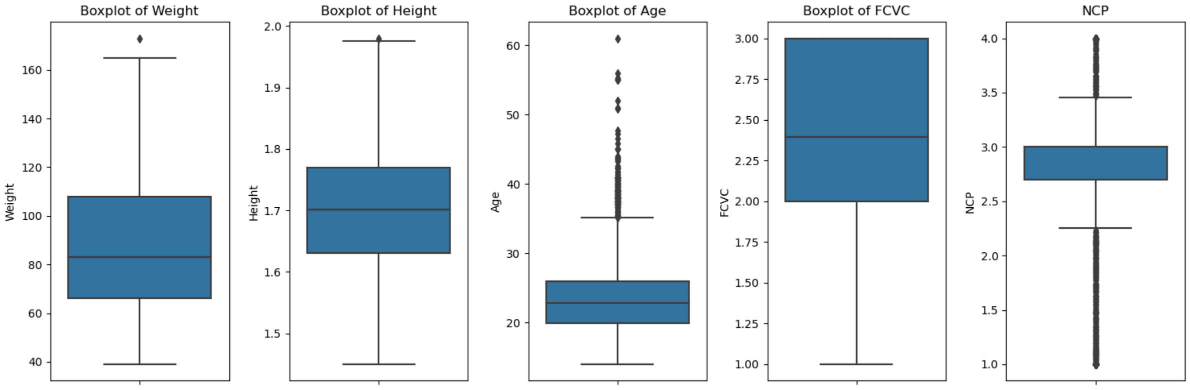
```
In [97]: obesitydf_encoded.shape
```

```
Out[97]: (2087, 20)
```

Checking and Handling Outliers

In [98]:

```
#Checking For outliners
plt.figure(figsize = (15,5))
# Plot the boxplot for Weight
plt.subplot(1, 5, 1)
sns.boxplot(y=obesity_dt['Weight'])
plt.title('Boxplot of Weight')

# Plot the boxplot for Height
plt.subplot(1, 5, 2)
sns.boxplot(y=obesity_dt['Height'])
plt.title('Boxplot of Height')

plt.subplot(1, 5, 3)
sns.boxplot(y=obesity_dt['Age'])
plt.title('Boxplot of Age')


# Plot the boxplot for Weight
plt.subplot(1, 5, 4)
sns.boxplot(y=obesity_dt['FCVC'])
plt.title('Boxplot of FCVC')

# Plot the boxplot for Height
plt.subplot(1, 5, 5)
sns.boxplot(y=obesity_dt['NCP'])
plt.title('NCP')





# Show the plot
plt.tight_layout()
plt.show()
```



In [99]:

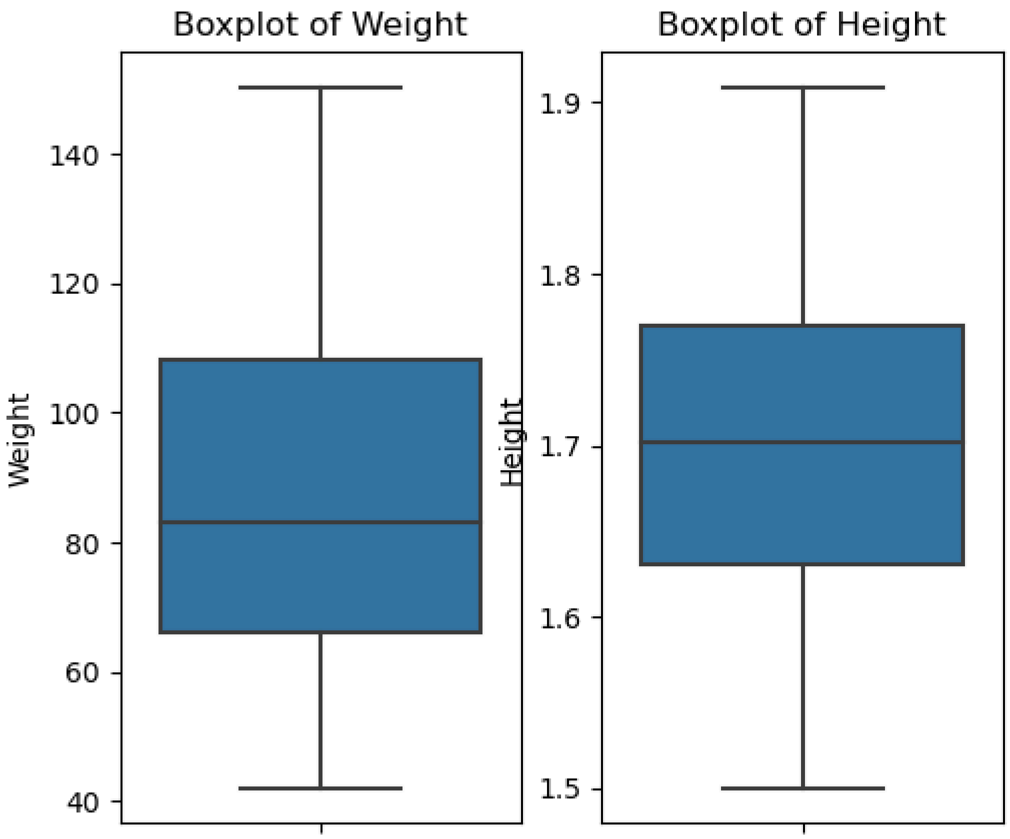
```
# Define a function to cap outliers
def cap_outliers(df, lower_quantile=0.01, upper_quantile=0.99):
    lower_bound = df.quantile(lower_quantile)
    upper_bound = df.quantile(upper_quantile)
    return df.clip(lower_bound, upper_bound)

# Apply capping to Weight and Height
obesity_dt['Weight'] = cap_outliers(obesity_dt['Weight'])
obesity_dt['Height'] = cap_outliers(obesity_dt['Height'])

obesity_dt.describe()
```

Out[99]:

	Age	Height	Weight	FCVC	NCP	CH2O	FAF
count	2087.000000	2087.000000	2087.000000	2087.000000	2087.000000	2087.000000	2087.000000
mean	24.353090	1.702594	86.822725	2.421466	2.701179	2.004749	1.012812
std	6.368801	0.092570	26.046264	0.534737	0.764614	0.608284	0.853475
min	14.000000	1.500000	41.995135	1.000000	1.000000	1.000000	0.000000
25%	19.915937	1.630178	66.000000	2.000000	2.697467	1.590922	0.124505
50%	22.847618	1.701584	83.101100	2.396265	3.000000	2.000000	1.000000
75%	26.000000	1.769491	108.015907	3.000000	3.000000	2.466193	1.678102
max	61.000000	1.909117	150.397017	3.000000	4.000000	3.000000	3.000000

In [100…

```
#Checking  if outliners are removed.
plt.figure(figsize = (15,5))
# Plot the boxplot for Weight
plt.subplot(1, 5, 1)
sns.boxplot(y=obesity_dt['Weight'])
plt.title('Boxplot of Weight')

# Plot the boxplot for Height
plt.subplot(1, 5, 2)
sns.boxplot(y=obesity_dt['Height'])
plt.title('Boxplot of Height')

plt.show()
```



Exploratory Data Analysis (EDA)

Summarizing the data

In [101…

```
obesity_dt.describe(include = 'number').T
```

Out[101]:

	count	mean	std	min	25%	50%	75%	max
Age	2087.0	24.353090	6.368801	14.000000	19.915937	22.847618	26.000000	61.000000
Height	2087.0	1.702594	0.092570	1.500000	1.630178	1.701584	1.769491	1.909117
Weight	2087.0	86.822725	26.046264	41.995135	66.000000	83.101100	108.015907	150.397017
FCVC	2087.0	2.421466	0.534737	1.000000	2.000000	2.396265	3.000000	3.000000
NCP	2087.0	2.701179	0.764614	1.000000	2.697467	3.000000	3.000000	4.000000
CH2O	2087.0	2.004749	0.608284	1.000000	1.590922	2.000000	2.466193	3.000000
FAF	2087.0	1.012812	0.853475	0.000000	0.124505	1.000000	1.678102	3.000000
TUE	2087.0	0.663035	0.608153	0.000000	0.000000	0.630866	1.000000	2.000000

In [102…

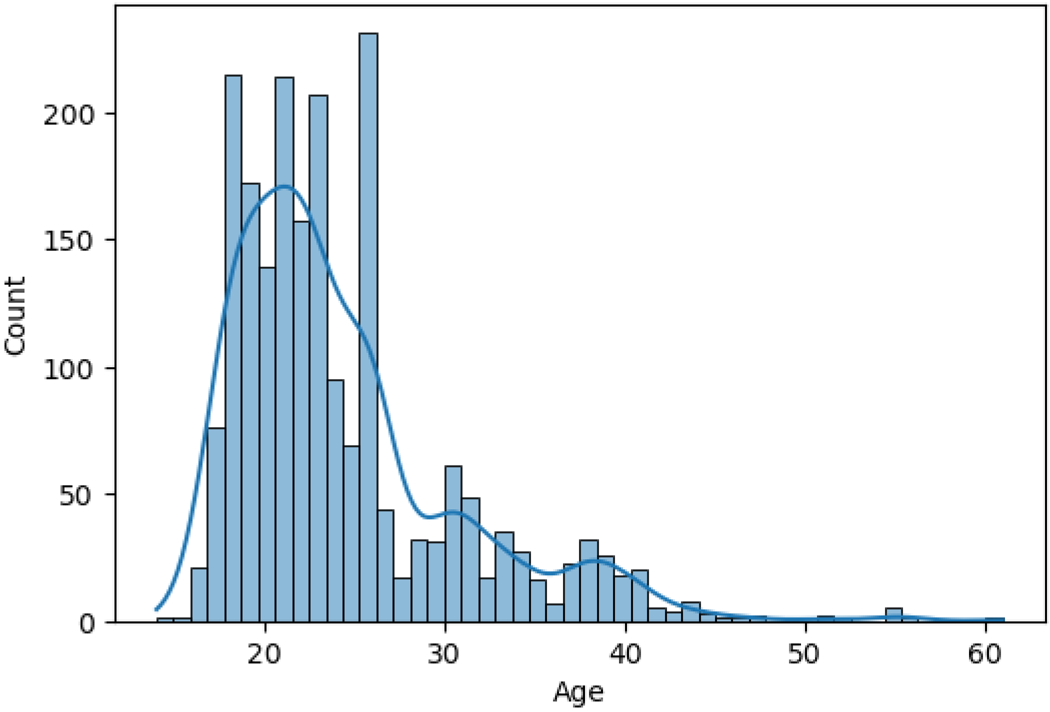
```
obesity_dt.describe(include = 'object')
```

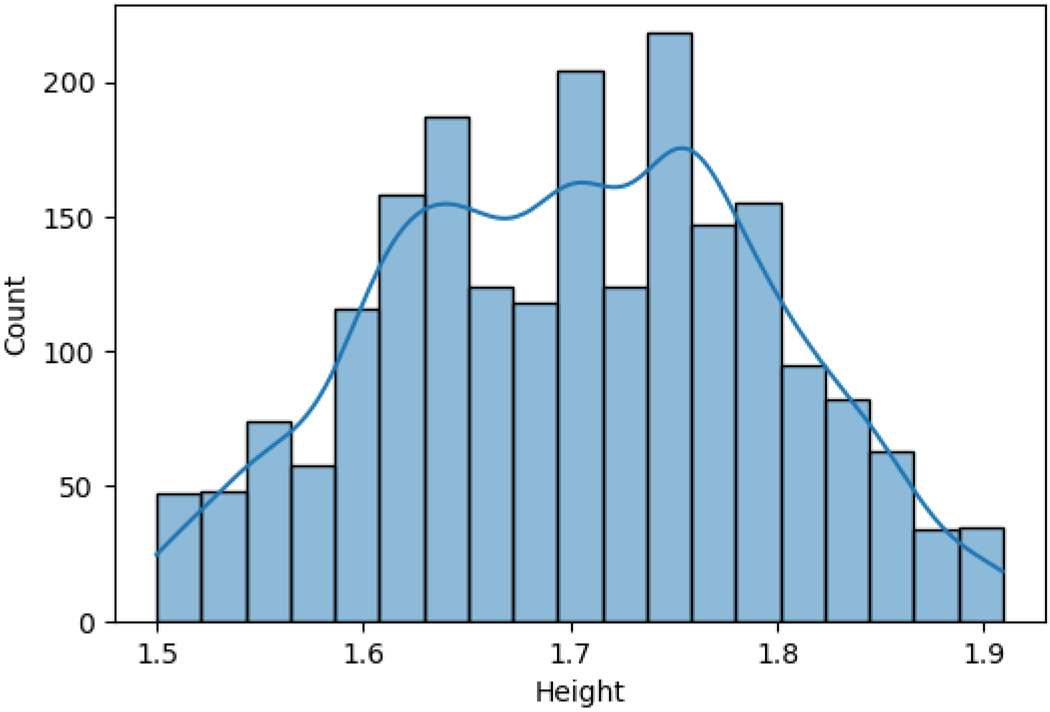
Out[102]:

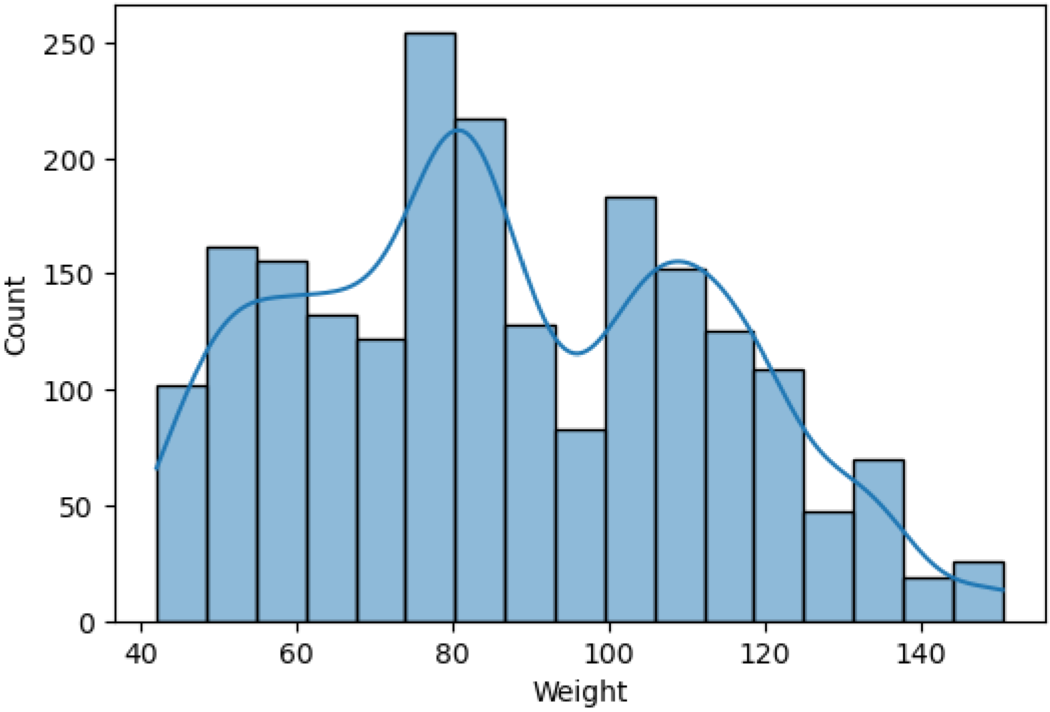
	Gender	family_history_with_overweight	FAVC	CAEC	SMOKE	SCC	CALC	
count	2087	2087	2087	2087	2087	2087	2087	
unique	2	2	2	4	2	2	4	
top	Male	yes	yes	Sometimes	no	no	Sometimes	Publ
freq	1052	1722	1844	1761	2043	1991	1380	

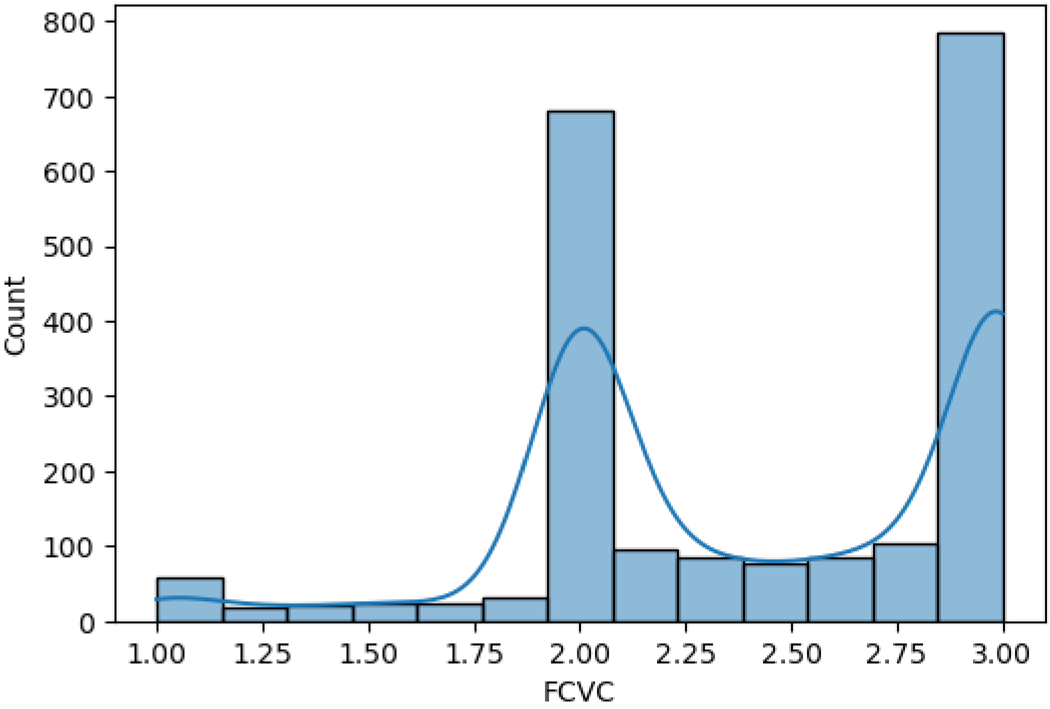
In [103…

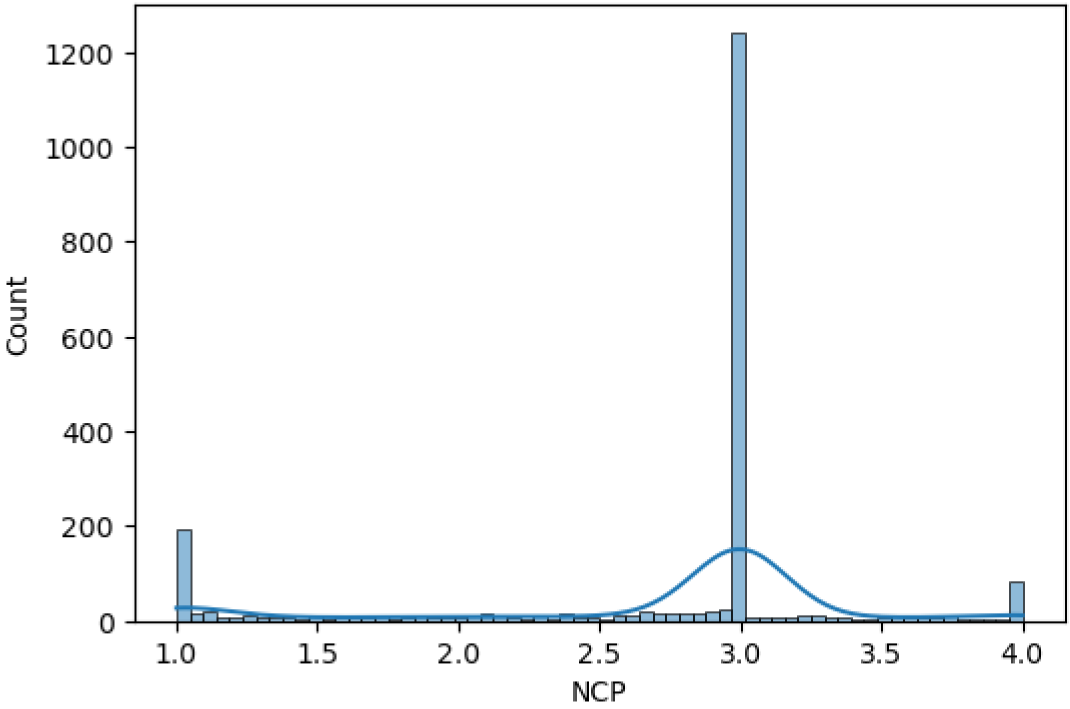
```
#Ploting Histogram to understand distribution
for i in obesity_dt.select_dtypes(include = 'number').columns:
    plt.figure(figsize= (6,4))
    sns.histplot(data = obesity_dt, kde=True, x = i)
    plt.show()
```

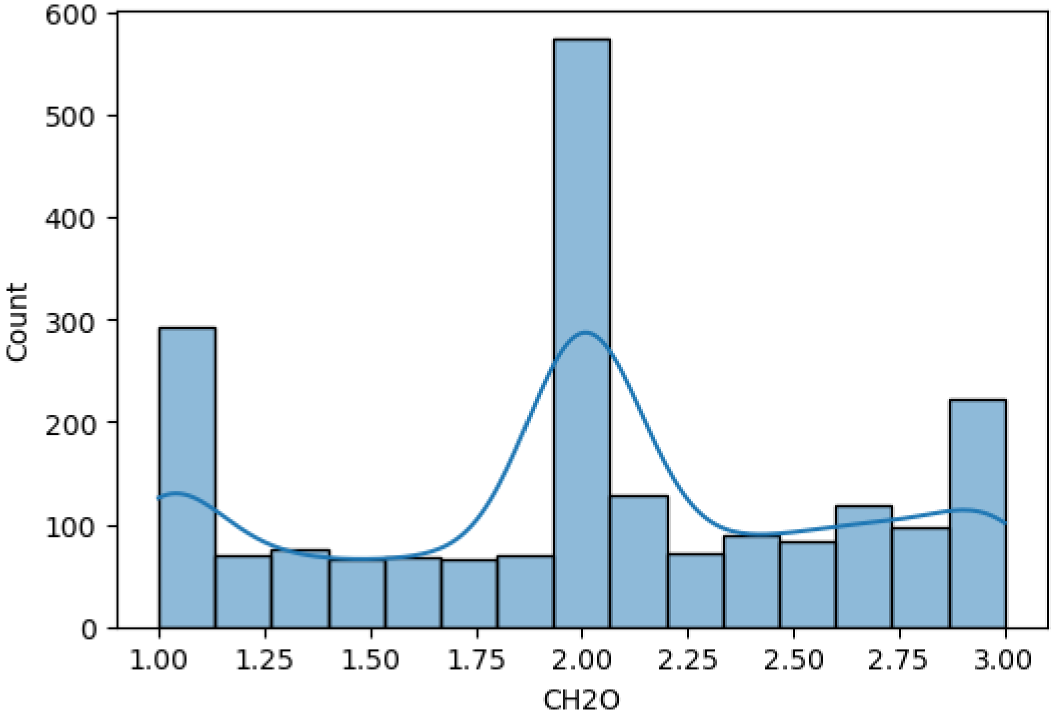

Count
0
50
100
150
200
20
30
40
50
60
Age

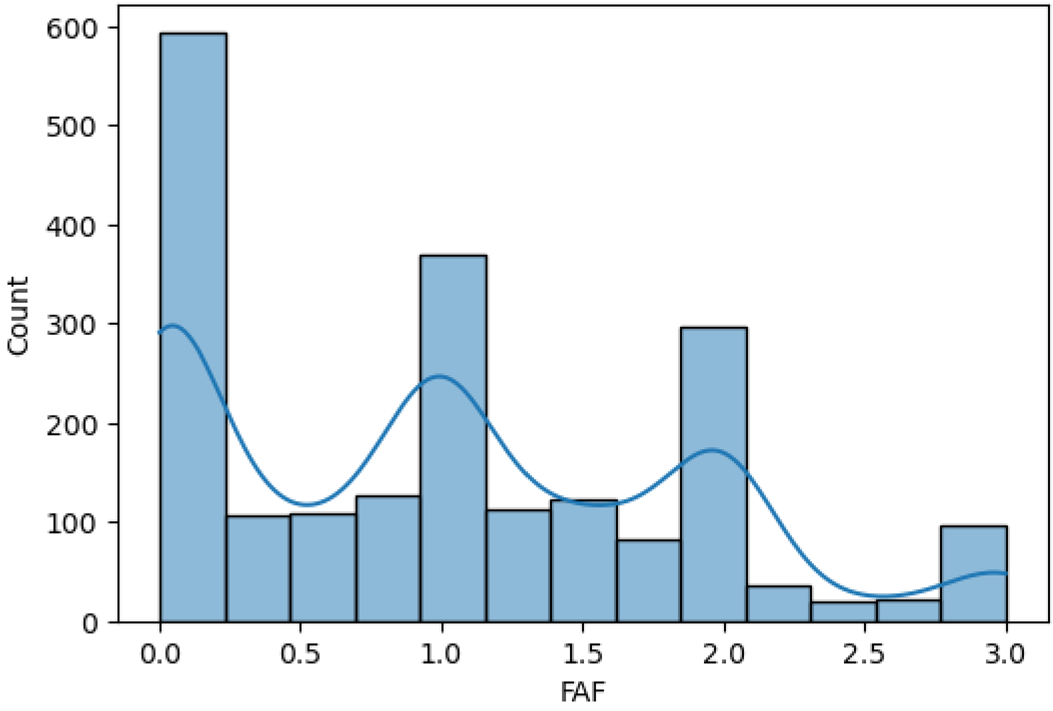

Count
0
50
100
150
200
1.5
1.6
1.7
1.8
1.9
Height

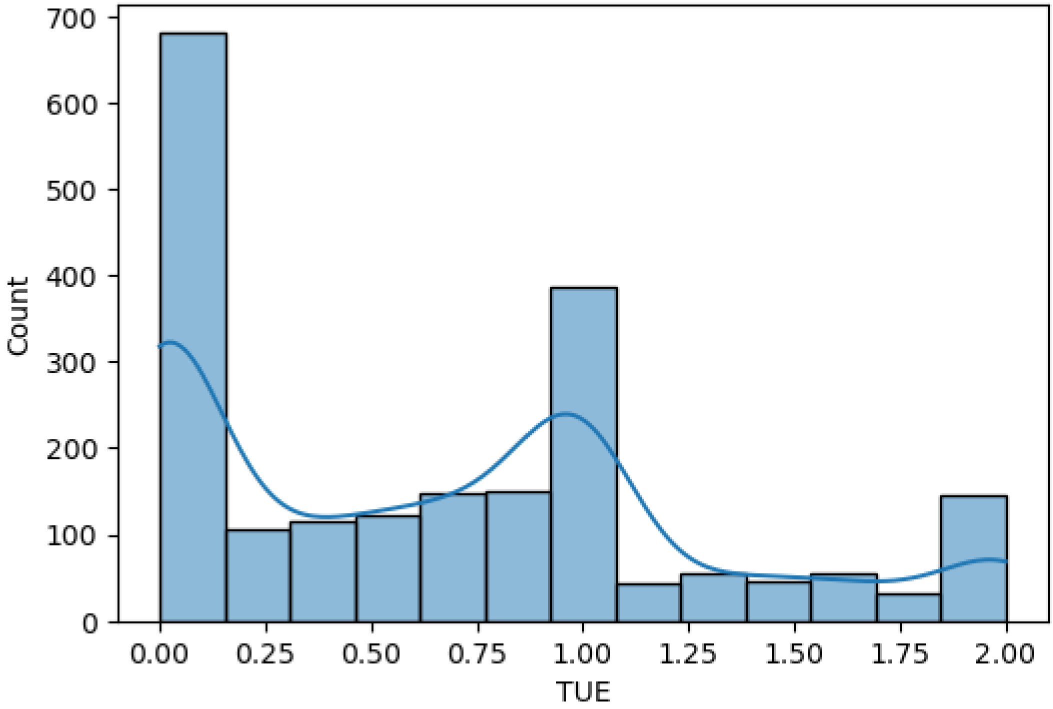

Count
0
50
100
150
200
250
40
60
80
100
120
140
Weight


Count
0
100
200
300
400
500
600
700
800
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
FCVC


Count
0
200
400
600
800
1000
1200
1.0
1.5
2.0
2.5
3.0
3.5
4.0
NCP


Count
0
100
200
300
400
500
600
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
CH2O



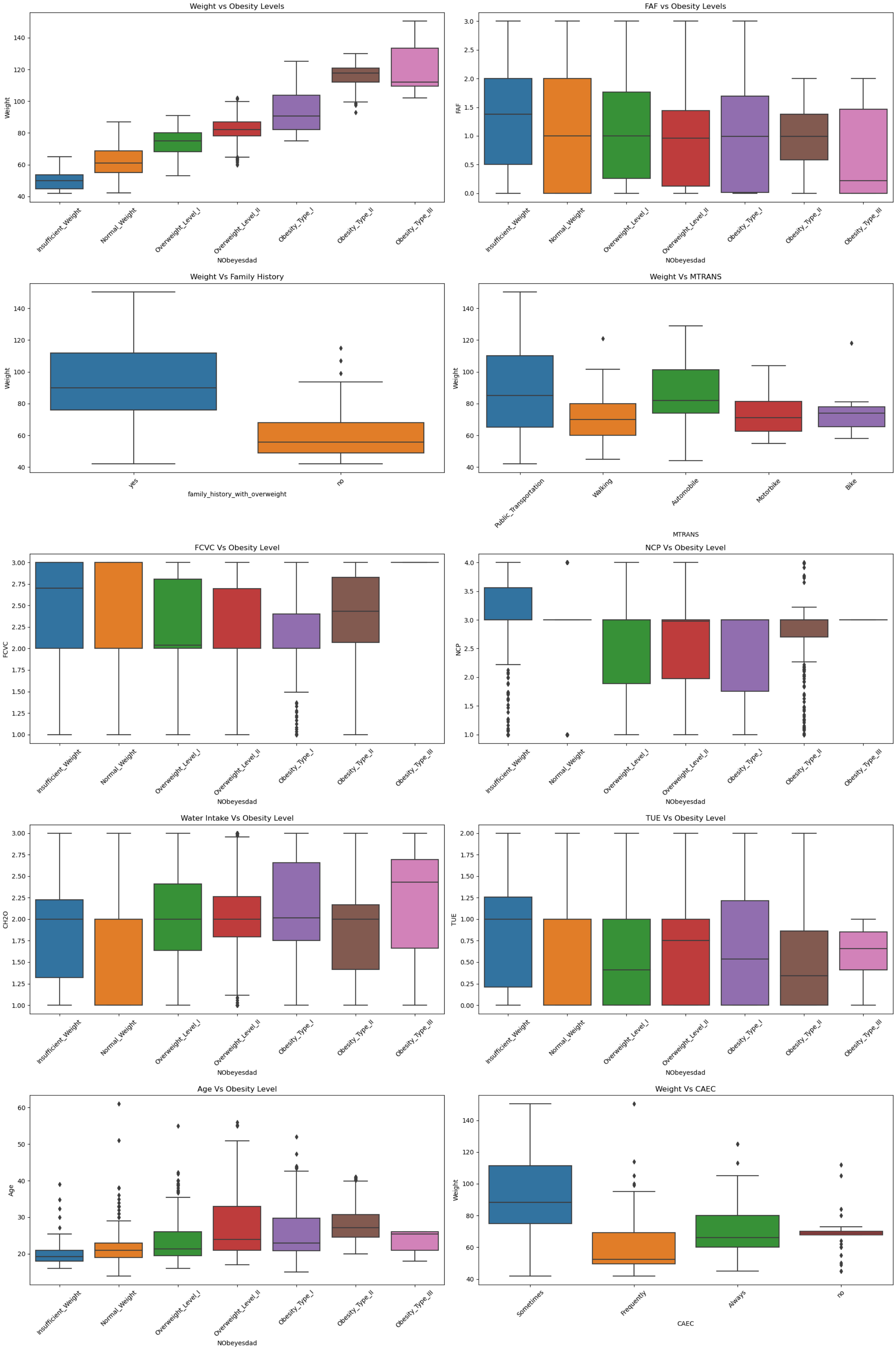


# Exploring Relationships between different attributes

In [104…

```
obesity_order = [
    "Insufficient_Weight",
    "Normal_Weight",
    "Overweight_Level_I",
    "Overweight_Level_II",
    "Obesity_Type_I",
    "Obesity_Type_II",
    "Obesity_Type_III"
]

fig, axes = plt.subplots(5, 2, figsize=(20, 30))
sns.boxplot(x='NObeyesdad', y='Weight',order =obesity_order, data=obesity_dt, ax=ax
axes[0,0].set_title('Weight vs Obesity Levels')
axes[0,0].tick_params(axis='x', rotation=45)
```

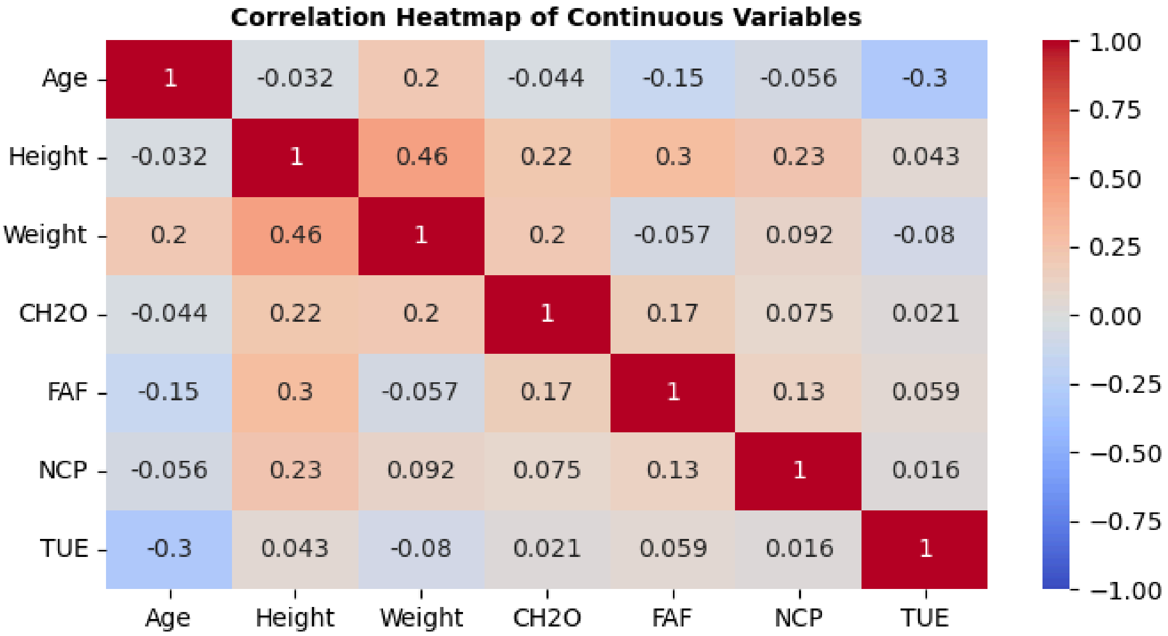
```
sns.boxplot(x='NObeyesdad', y='FAF', data=obesity_dt,order =obesity_order, ax=axes[
axes[0,1].set_title('FAF vs Obesity Levels')
axes[0,1].tick_params(axis='x', rotation=45)


sns.boxplot(x='family_history_with_overweight',y = 'Weight', data=obesity_dt, ax=ax
axes[1,0].set_title('Weight Vs Family History')
axes[1,0].tick_params(axis = 'x', rotation = 45)

sns.boxplot(x='MTRANS',y = 'Weight', data=obesity_dt, ax=axes[1,1])
axes[1,1].set_title('Weight Vs MTRANS')
axes[1,1].tick_params(axis = 'x', rotation = 45)

sns.boxplot(x='NObeyesdad',y = 'FCVC', order =obesity_order, data=obesity_dt, ax=ax
axes[2,0].set_title('FCVC Vs Obesity Level')
axes[2,0].tick_params(axis = 'x', rotation = 45)

sns.boxplot(x='NObeyesdad',y = 'NCP',order =obesity_order, data=obesity_dt, ax=axes
axes[2,1].set_title('NCP Vs Obesity Level')
axes[2,1].tick_params(axis = 'x', rotation = 45)

sns.boxplot(x='NObeyesdad',y = 'CH2O',order =obesity_order, data=obesity_dt, ax=axe
axes[3,0].set_title('Water Intake Vs Obesity Level')
axes[3,0].tick_params(axis = 'x', rotation = 45)

sns.boxplot(x='NObeyesdad',y = 'TUE',order =obesity_order, data=obesity_dt, ax=axes
axes[3,1].set_title('TUE Vs Obesity Level')
axes[3,1].tick_params(axis = 'x', rotation = 45)

sns.boxplot(x='NObeyesdad',y = 'Age',order =obesity_order, data=obesity_dt, ax=axes
axes[4,0].set_title('Age Vs Obesity Level')
axes[4,0].tick_params(axis = 'x', rotation = 45)

sns.boxplot(x='CAEC',y = 'Weight', data=obesity_dt, ax=axes[4,1])
axes[4,1].set_title('Weight Vs CAEC')
axes[4,1].tick_params(axis = 'x', rotation = 45)




plt.tight_layout()
plt.show()
```



In [105…

```
#Checking for relationship between Variables
cols = obesity_dt[['Age', 'Height', 'Weight','CH2O', 'FAF','NCP', 'TUE']]

# Calculating the correlation matrix
correlation_matrix = cols.corr()

# Plotting the heatmap
plt.figure(figsize=(8, 4))
```

```
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title("Correlation Heatmap of Continuous Variables", weight = 'bold', fontsize
plt.show()
```

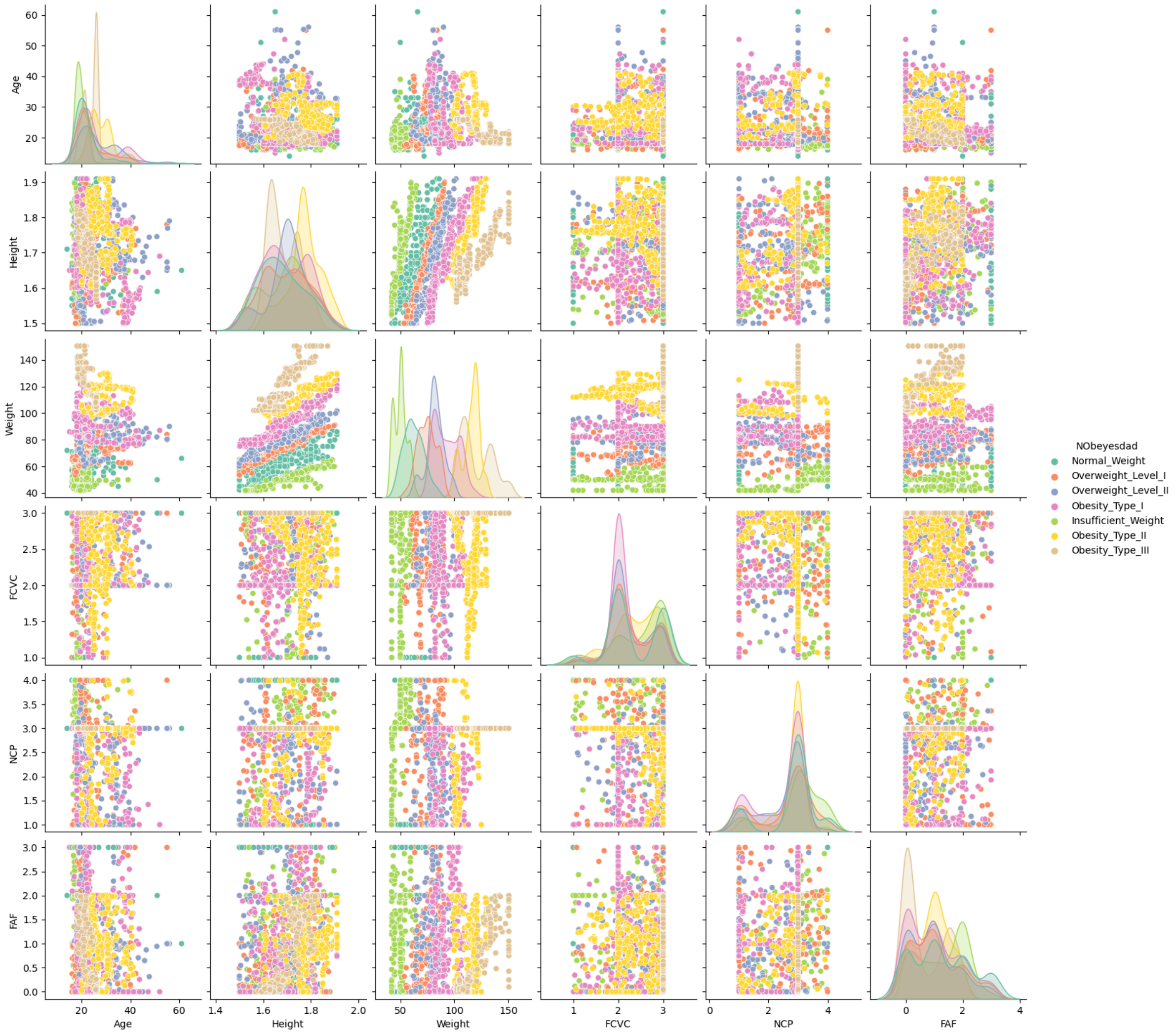


Preparing Data For Machine Learning

In [106…

```
# Pair plots
selected_features = ['Age', 'Height', 'Weight', 'FCVC', 'NCP', 'FAF']
pairplot_data = obesity_dt[selected_features + ['NObeyesdad']]

# Pair plot colored by the obesity levels
sns.pairplot(pairplot_data, hue='NObeyesdad', diag_kind='kde', palette='Set2')
plt.suptitle('Pair Plot of continuous variables', y= 1.02)
plt.show()
```

Pair Plot of continuous variables



In [107…

```
from sklearn.preprocessing  import LabelEncoder,OneHotEncoder


obesity_dt.dtypes

# Create a copy of the data for encoding
obesitydf_encoded = obesity_dt.copy()

# Columns for label encoding (binary)
binary_columns = ['Gender', 'SMOKE', 'family_history_with_overweight', 'FAVC', 'SCC
label_encoder = LabelEncoder()

# Apply label encoding to binary columns
for col in binary_columns:
    obesitydf_encoded[col] = label_encoder.fit_transform(obesitydf_encoded[col])

# One-hot encode multi-class columns
multi_class_columns = ['MTRANS']

# drop='first' to avoid multicollinearity and convert to a dataframe
OHE = OneHotEncoder(handle_unknown = 'ignore',sparse_output=False, drop='first').se

# Apply one-hot encoding to the selected columns
encoded_features = OHE.fit_transform(obesitydf_encoded[multi_class_columns])


# Drop the original multi-class columns and concatenate the new one-hot encoded col
obesitydf_encoded = obesitydf_encoded.drop(columns=multi_class_columns)
obesitydf_encoded = pd.concat([obesitydf_encoded, encoded_features], axis=1)
```

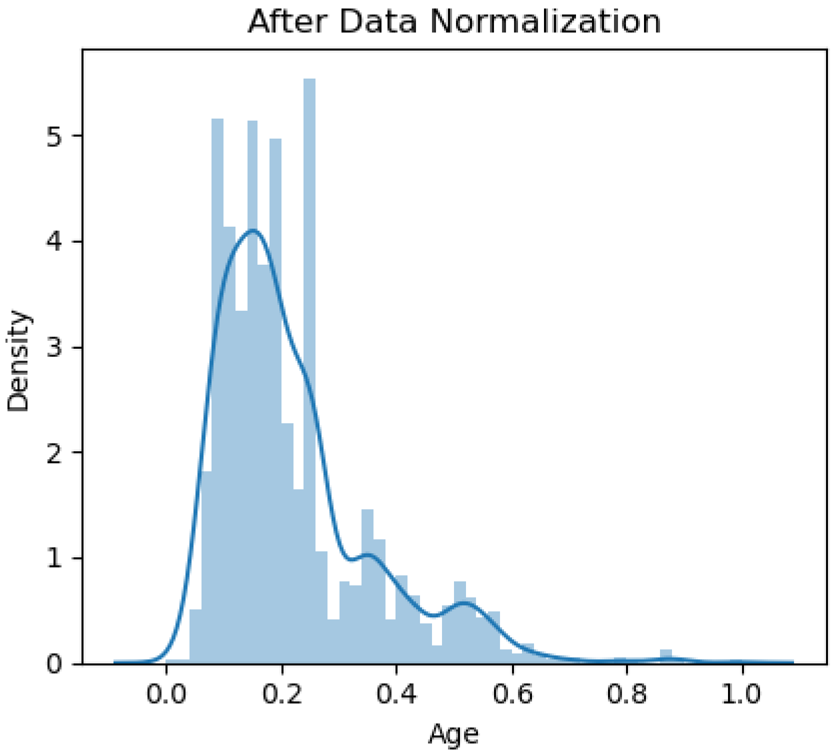
```
# Display the first few rows of the copied dataset
obesitydf_encoded.head()
```

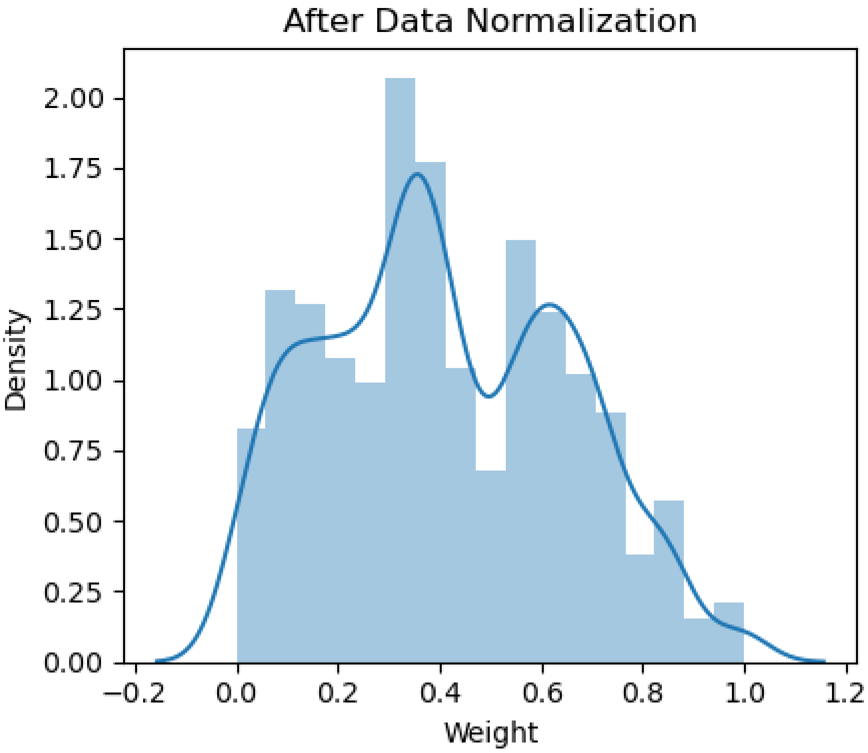
Out[107]:

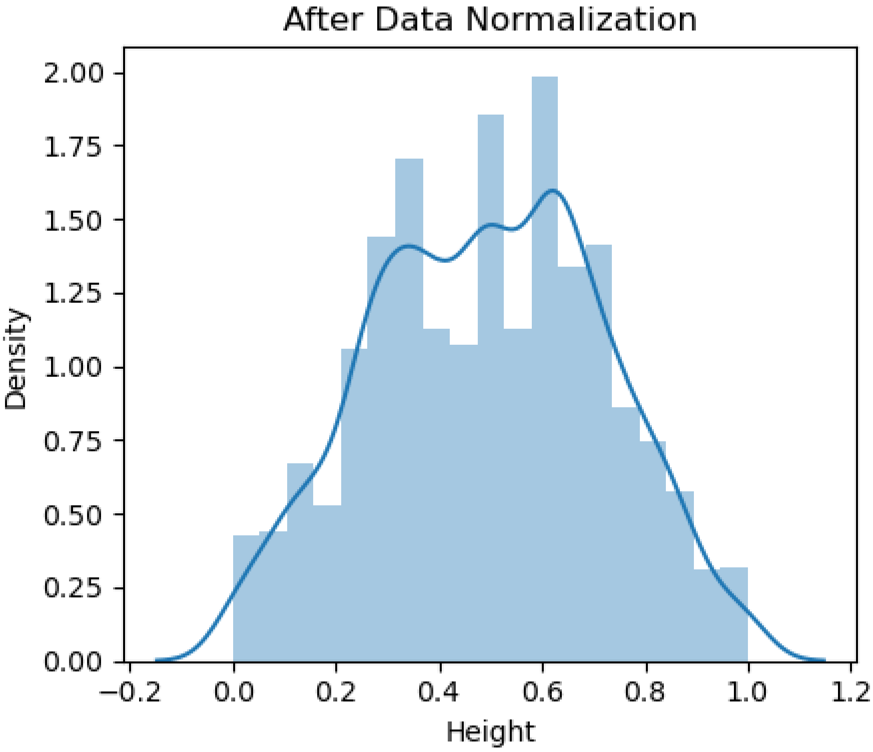
	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	NCP	CAEC	SMOI
0	0	21.0	1.62	64.0	1	0	2.0	3.0	2	
1	0	21.0	1.52	56.0	1	0	3.0	3.0	2	
2	1	23.0	1.80	77.0	1	0	2.0	3.0	2	
3	1	27.0	1.80	87.0	0	0	3.0	3.0	2	
4	1	22.0	1.78	89.8	0	0	2.0	1.0	2	

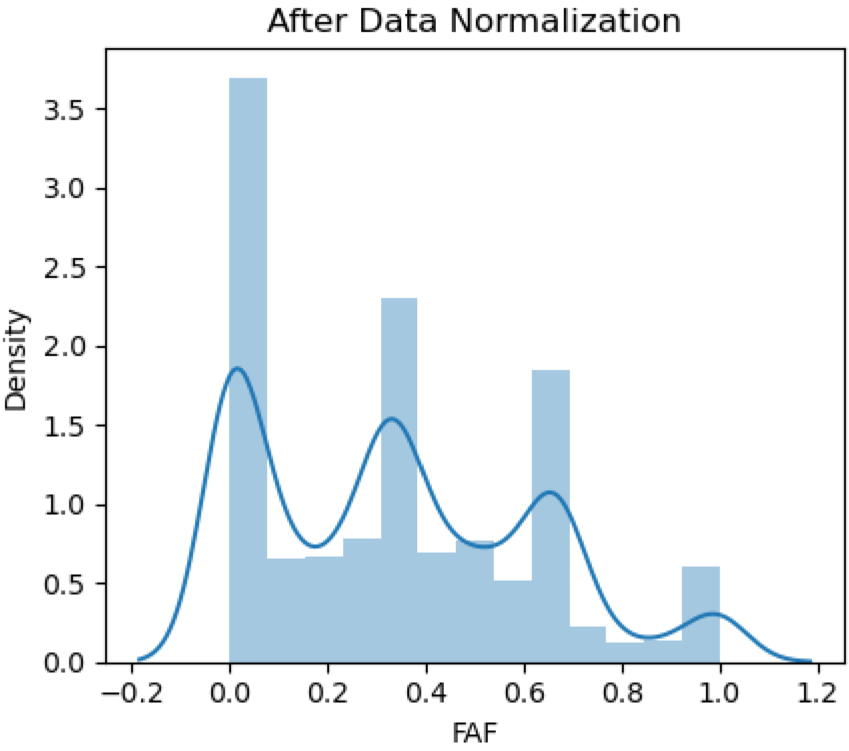
In [108…

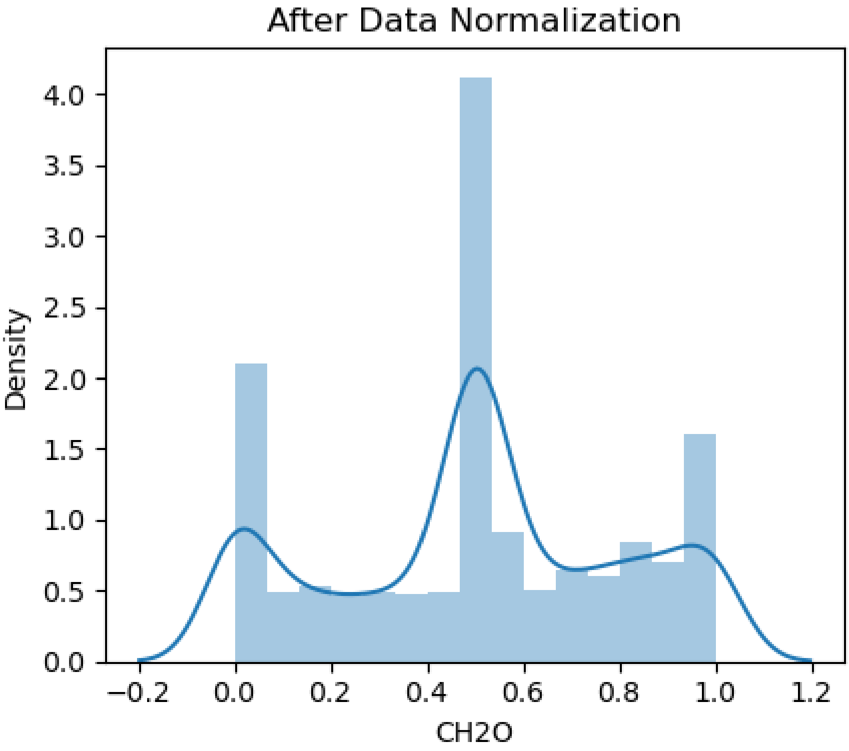
```
import warnings
warnings.filterwarnings("ignore")
col = ['Age', 'Weight', 'Height','FAF','CH2O','NCP', 'FCVC']

from sklearn.preprocessing import MinMaxScaler
# Data Normalize Continuous Variables using Min-Max Scaling
scaler = MinMaxScaler()
obesitydf_encoded[col] = scaler.fit_transform(obesitydf_encoded[col])


for i in col:
    plt.figure(figsize = (16,4))
    plt.subplot(141)
    sns.distplot(obesitydf_encoded[i],label = 'skew:' + str(np.round(obesitydf_enco
    plt.title('After Data Normalization')
    plt.tight_layout()
    plt.show()
```

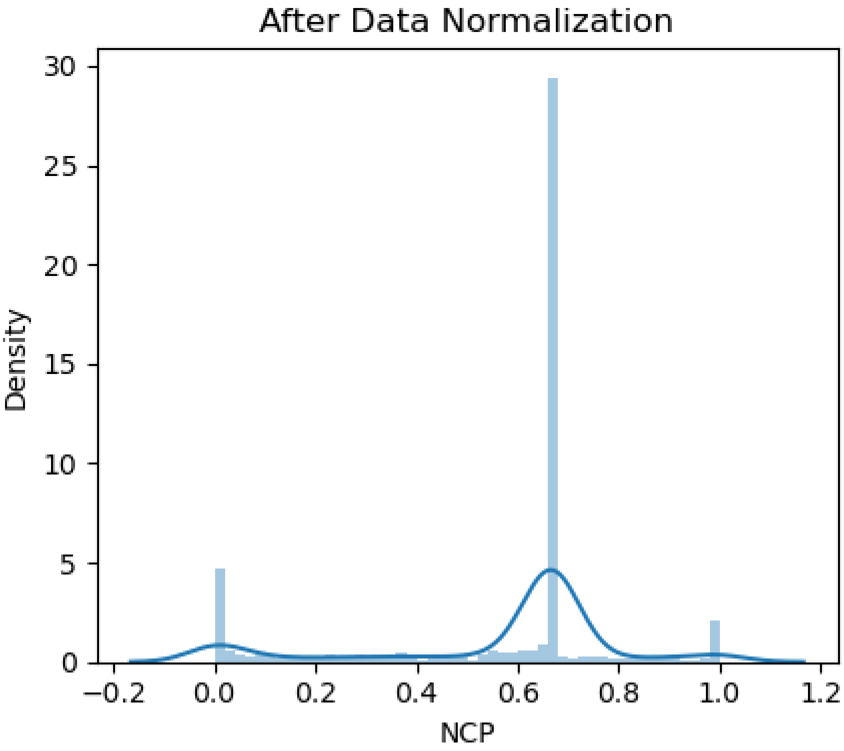


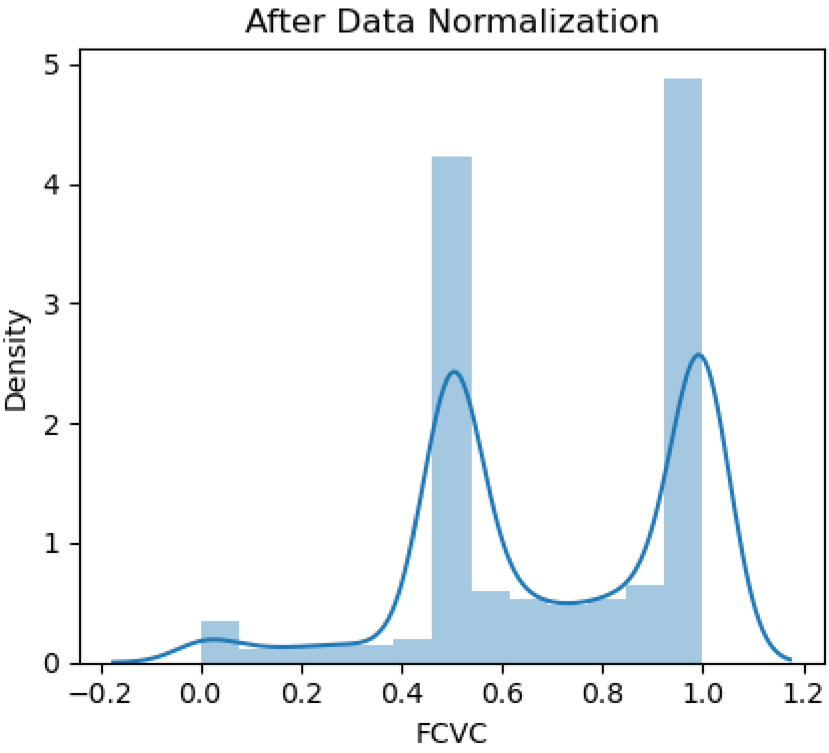

After Data Normalization
Density
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Weight


After Data Normalization
Density
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Height


After Data Normalization
Density
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
FAF


After Data Normalization
Density
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
CH2O





In [109…

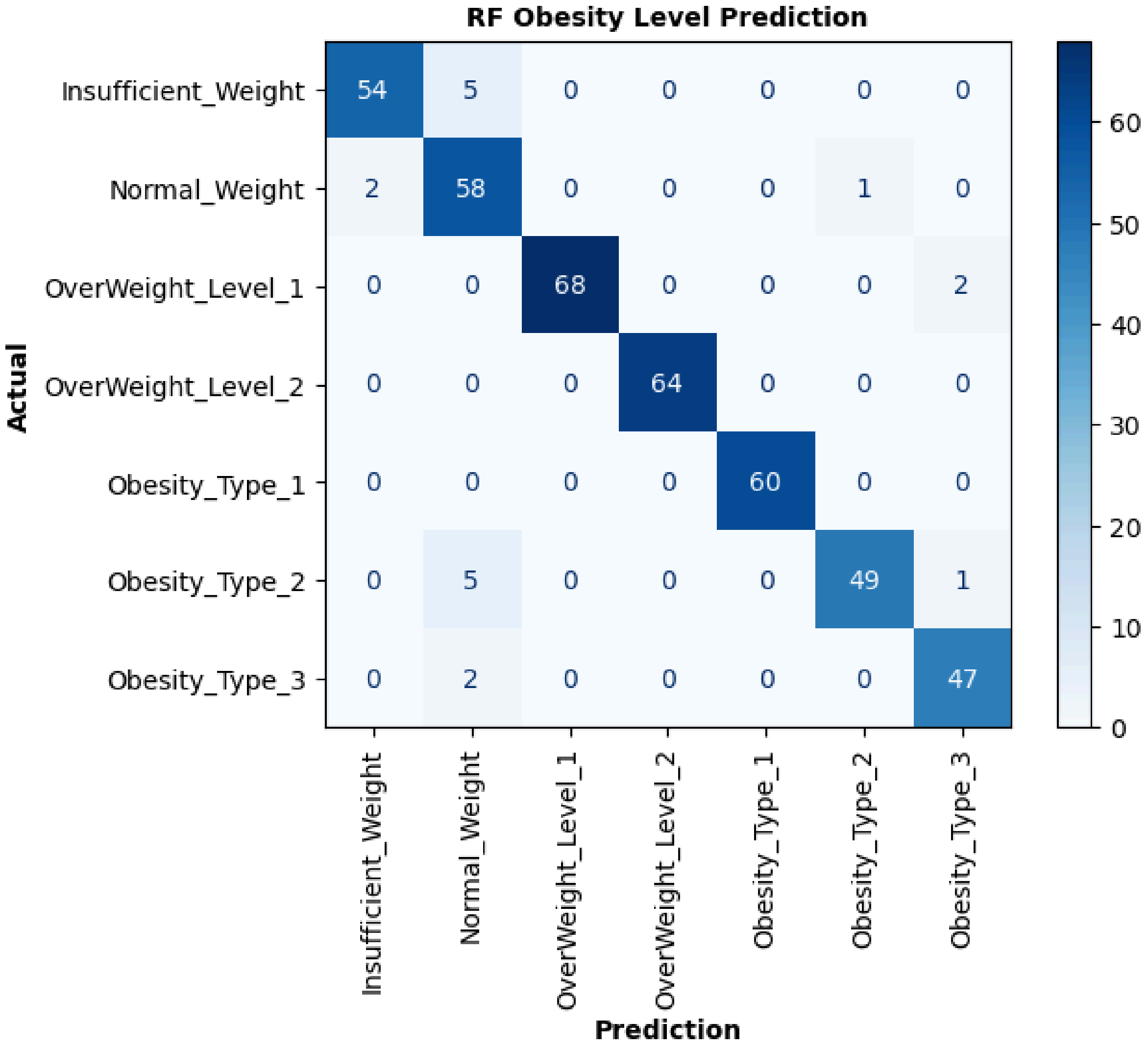
```
obesitydf_encoded.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 2087 entries, 0 to 2110
Data columns (total 20 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   Gender                          2087 non-null   int32  
 1   Age                             2087 non-null   float64
 2   Height                          2087 non-null   float64
 3   Weight                          2087 non-null   float64
 4   family_history_with_overweight  2087 non-null   int32  
 5   FAVC                            2087 non-null   int32  
 6   FCVC                            2087 non-null   float64
 7   NCP                             2087 non-null   float64
 8   CAEC                            2087 non-null   int32  
 9   SMOKE                           2087 non-null   int32  
 10  CH2O                            2087 non-null   float64
 11  SCC                             2087 non-null   int32  
 12  FAF                             2087 non-null   float64
 13  TUE                             2087 non-null   float64
 14  CALC                            2087 non-null   int32  
 15  NObeyesdad                      2087 non-null   int32  
 16  MTRANS_Bike                     2087 non-null   float64
 17  MTRANS_Motorbike                2087 non-null   float64
 18  MTRANS_Public_Transportation    2087 non-null   float64
 19  MTRANS_Walking                  2087 non-null   float64
dtypes: float64(12), int32(8)
memory usage: 277.2 KB
```

In [110…

```python
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay, classificatio


X = obesitydf_encoded.drop(columns=['NObeyesdad'])
y = obesitydf_encoded['NObeyesdad']

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_st

label_mapping = {
    0: 'Insufficient_Weight',
    1: 'Normal_Weight',
    2: 'OverWeight_Level_1',
    3: 'OverWeight_Level_2',
    4: 'Obesity_Type_1',
    5: 'Obesity_Type_2',
    6: 'Obesity_Type_3'
}

# Apply the mapping
y_train = y_train.map(label_mapping)
y_test = y_test.map(label_mapping)

# Train the RandomForestClassifier
rf_model = RandomForestClassifier(n_estimators=1000, random_state=42)
rf_model.fit(X_train, y_train)
print(rf_model.score(X_test, y_test))

# Make predictions
y_pred = rf_model.predict(X_test)

#  Generate and display the confusion matrix
cm = confusion_matrix(y_test, y_pred, labels = ['Insufficient_Weight','Normal_Weigh
# Plot the confusion matrix
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels= ['Insufficient_W
plt.figure(figsize=(10, 8))
disp.plot(cmap='Blues', colorbar=True)
plt.xticks(rotation = 90)
plt.xlabel('Prediction', weight = 'bold')
plt.ylabel('Actual', weight = 'bold')
plt.title("RF Obesity Level Prediction", weight = 'bold', fontsize = 10)
plt.show()

print(y_pred)
print(classification_report(y_test,y_pred))
```

```
0.9569377990430622
<Figure size 1000x800 with 0 Axes>
```



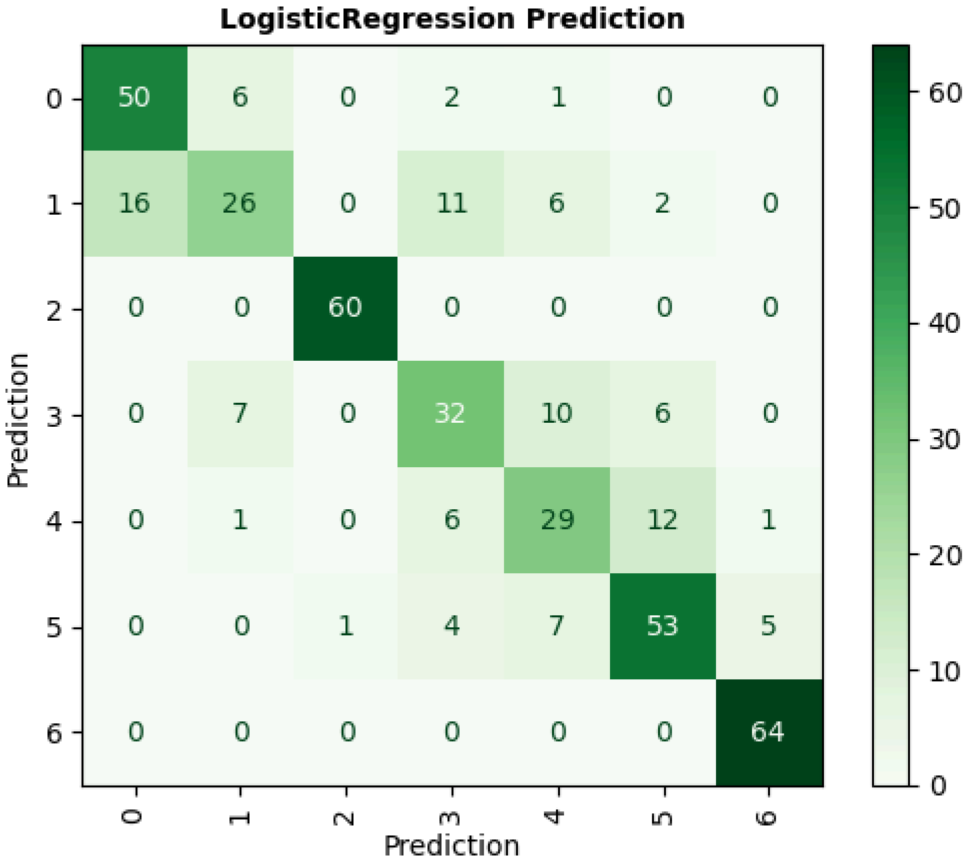
```
['OverWeight_Level_1' 'OverWeight_Level_2' 'Obesity_Type_2'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Obesity_Type_2'
 'Obesity_Type_3' 'Obesity_Type_1' 'Insufficient_Weight' 'Normal_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_1' 'Obesity_Type_1'
 'Normal_Weight' 'Obesity_Type_3' 'Obesity_Type_2' 'Obesity_Type_2'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Obesity_Type_2' 'Obesity_Type_1'
 'Obesity_Type_2' 'Normal_Weight' 'Insufficient_Weight'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Insufficient_Weight'
 'Insufficient_Weight' 'Insufficient_Weight' 'OverWeight_Level_1'
 'Normal_Weight' 'Normal_Weight' 'OverWeight_Level_1' 'OverWeight_Level_2'
 'Obesity_Type_3' 'Obesity_Type_1' 'Normal_Weight' 'Normal_Weight'
 'Insufficient_Weight' 'Normal_Weight' 'Normal_Weight'
 'OverWeight_Level_1' 'Obesity_Type_2' 'Obesity_Type_1' 'Normal_Weight'
 'Obesity_Type_2' 'OverWeight_Level_2' 'Obesity_Type_2' 'Normal_Weight'
 'Obesity_Type_1' 'Obesity_Type_3' 'Insufficient_Weight'
 'OverWeight_Level_1' 'Obesity_Type_1' 'Obesity_Type_1' 'Obesity_Type_1'
 'Obesity_Type_2' 'OverWeight_Level_1' 'Obesity_Type_3' 'Obesity_Type_3'
 'Obesity_Type_3' 'OverWeight_Level_1' 'Normal_Weight'
 'OverWeight_Level_2' 'Normal_Weight' 'OverWeight_Level_2'
 'OverWeight_Level_1' 'Obesity_Type_2' 'Obesity_Type_3'
 'OverWeight_Level_1' 'Obesity_Type_3' 'Normal_Weight'
 'OverWeight_Level_1' 'Normal_Weight' 'OverWeight_Level_1'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Normal_Weight'
 'Insufficient_Weight' 'Insufficient_Weight' 'Obesity_Type_2'
 'Obesity_Type_2' 'Obesity_Type_1' 'OverWeight_Level_1'
 'Insufficient_Weight' 'Insufficient_Weight' 'OverWeight_Level_2'
 'Obesity_Type_3' 'Normal_Weight' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'OverWeight_Level_1' 'OverWeight_Level_2'
 'OverWeight_Level_1' 'Obesity_Type_1' 'Obesity_Type_3' 'Obesity_Type_1'
 'Normal_Weight' 'Obesity_Type_3' 'OverWeight_Level_1' 'Obesity_Type_3'
 'Normal_Weight' 'Normal_Weight' 'Normal_Weight' 'Normal_Weight'
 'Obesity_Type_1' 'OverWeight_Level_1' 'Normal_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_1' 'Insufficient_Weight'
 'Obesity_Type_1' 'OverWeight_Level_2' 'Obesity_Type_2'
 'OverWeight_Level_2' 'Normal_Weight' 'OverWeight_Level_2'
 'Insufficient_Weight' 'Obesity_Type_2' 'OverWeight_Level_1'
 'Insufficient_Weight' 'Obesity_Type_2' 'Obesity_Type_1' 'Obesity_Type_2'
 'Normal_Weight' 'Obesity_Type_2' 'Normal_Weight' 'OverWeight_Level_2'
 'Insufficient_Weight' 'Obesity_Type_3' 'OverWeight_Level_1'
 'Obesity_Type_2' 'OverWeight_Level_1' 'Obesity_Type_3' 'Normal_Weight'
 'Obesity_Type_2' 'Obesity_Type_2' 'Obesity_Type_1' 'Obesity_Type_2'
 'Normal_Weight' 'Normal_Weight' 'Obesity_Type_1' 'Normal_Weight'
 'Obesity_Type_2' 'Obesity_Type_1' 'Obesity_Type_1' 'Obesity_Type_3'
 'Insufficient_Weight' 'Obesity_Type_1' 'Obesity_Type_1'
 'OverWeight_Level_1' 'Insufficient_Weight' 'OverWeight_Level_2'
 'Insufficient_Weight' 'Obesity_Type_1' 'Obesity_Type_1'
 'OverWeight_Level_1' 'Normal_Weight' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'Insufficient_Weight' 'OverWeight_Level_2'
 'Normal_Weight' 'Insufficient_Weight' 'Obesity_Type_1' 'Obesity_Type_3'
 'Normal_Weight' 'OverWeight_Level_1' 'Insufficient_Weight'
 'OverWeight_Level_2' 'Obesity_Type_2' 'Insufficient_Weight'
 'Obesity_Type_3' 'Obesity_Type_3' 'Obesity_Type_3' 'Obesity_Type_3'
 'Normal_Weight' 'Insufficient_Weight' 'Normal_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Insufficient_Weight'
 'Obesity_Type_3' 'Normal_Weight' 'OverWeight_Level_2' 'Obesity_Type_3'
 'OverWeight_Level_1' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'OverWeight_Level_1' 'Normal_Weight' 'Obesity_Type_1' 'Normal_Weight'
 'OverWeight_Level_1' 'Obesity_Type_2' 'Obesity_Type_3' 'Normal_Weight'
 'OverWeight_Level_1' 'Obesity_Type_1' 'OverWeight_Level_1'
 'Obesity_Type_2' 'Obesity_Type_3' 'OverWeight_Level_1'
 'OverWeight_Level_1' 'Obesity_Type_3' 'Obesity_Type_2'
 'OverWeight_Level_1' 'Normal_Weight' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'Obesity_Type_3' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'Obesity_Type_1' 'Obesity_Type_1' 'Obesity_Type_3'
 'Obesity_Type_1' 'OverWeight_Level_1' 'Insufficient_Weight'
 'Obesity_Type_2' 'Obesity_Type_1' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'Obesity_Type_2' 'Obesity_Type_1' 'Obesity_Type_3'
 'Obesity_Type_2' 'OverWeight_Level_1' 'Obesity_Type_3' 'Normal_Weight'
 'Obesity_Type_2' 'Insufficient_Weight' 'OverWeight_Level_2'
 'Obesity_Type_3' 'Obesity_Type_1' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'Insufficient_Weight' 'Normal_Weight'
 'Obesity_Type_1' 'Obesity_Type_2' 'OverWeight_Level_2' 'Normal_Weight'
 'Normal_Weight' 'Obesity_Type_2' 'OverWeight_Level_2' 'Obesity_Type_1'
 'Obesity_Type_3' 'Obesity_Type_1' 'Normal_Weight' 'Normal_Weight'
 'OverWeight_Level_1' 'Insufficient_Weight' 'Obesity_Type_1'
 'Normal_Weight' 'Obesity_Type_1' 'OverWeight_Level_1'
 'OverWeight_Level_1' 'Obesity_Type_2' 'OverWeight_Level_2'
 'Obesity_Type_1' 'OverWeight_Level_1' 'Obesity_Type_2' 'Normal_Weight'
 'Insufficient_Weight' 'OverWeight_Level_1' 'Obesity_Type_1'
 'Insufficient_Weight' 'Obesity_Type_1' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'Normal_Weight' 'OverWeight_Level_1'
 'Obesity_Type_3' 'OverWeight_Level_2' 'Normal_Weight' 'Normal_Weight'
 'Obesity_Type_3' 'Obesity_Type_3' 'OverWeight_Level_1' 'Obesity_Type_2'
 'OverWeight_Level_1' 'Obesity_Type_2' 'OverWeight_Level_2'
 'Obesity_Type_1' 'OverWeight_Level_2' 'Insufficient_Weight'
 'Obesity_Type_2' 'OverWeight_Level_1' 'OverWeight_Level_1'
 'Obesity_Type_3' 'OverWeight_Level_1' 'Normal_Weight' 'Normal_Weight'
 'Obesity_Type_1' 'Insufficient_Weight' 'Obesity_Type_2'
 'OverWeight_Level_1' 'Insufficient_Weight' 'Insufficient_Weight'
 'Normal_Weight' 'Insufficient_Weight' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Insufficient_Weight'
 'Obesity_Type_2' 'Obesity_Type_1' 'Insufficient_Weight' 'Obesity_Type_1'
 'Obesity_Type_3' 'Obesity_Type_1' 'Normal_Weight' 'Normal_Weight'
 'Insufficient_Weight' 'Obesity_Type_2' 'OverWeight_Level_2'
 'Obesity_Type_3' 'Normal_Weight' 'OverWeight_Level_1'
 'OverWeight_Level_1' 'Insufficient_Weight' 'Obesity_Type_1'
 'OverWeight_Level_1' 'Insufficient_Weight' 'Insufficient_Weight'
 'Obesity_Type_1' 'Insufficient_Weight' 'Obesity_Type_1'
 'OverWeight_Level_2' 'Normal_Weight' 'Insufficient_Weight'
 'Insufficient_Weight' 'Obesity_Type_1' 'Normal_Weight'
 'OverWeight_Level_1' 'Normal_Weight' 'OverWeight_Level_1'
 'OverWeight_Level_1' 'Obesity_Type_2' 'Normal_Weight'
 'Insufficient_Weight' 'Obesity_Type_1' 'Obesity_Type_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Normal_Weight'
 'Obesity_Type_2' 'Obesity_Type_2' 'Obesity_Type_3' 'OverWeight_Level_2'
 'Normal_Weight' 'Obesity_Type_3' 'Obesity_Type_2' 'Obesity_Type_1'
 'OverWeight_Level_2' 'Obesity_Type_1' 'OverWeight_Level_1'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Obesity_Type_1'
 'Obesity_Type_1' 'OverWeight_Level_2' 'OverWeight_Level_1'
 'OverWeight_Level_1' 'Obesity_Type_1' 'Obesity_Type_2' 'Obesity_Type_3'
 'Insufficient_Weight' 'Insufficient_Weight' 'Obesity_Type_3'
 'Obesity_Type_1' 'Normal_Weight' 'Obesity_Type_3' 'Normal_Weight'
 'Obesity_Type_1' 'Insufficient_Weight' 'Obesity_Type_3'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Obesity_Type_2'
 'Obesity_Type_1' 'Obesity_Type_3' 'Insufficient_Weight'
 'OverWeight_Level_1' 'Obesity_Type_3' 'Obesity_Type_3'
 'OverWeight_Level_1' 'Normal_Weight' 'Insufficient_Weight'
 'Normal_Weight' 'Obesity_Type_1' 'Obesity_Type_1' 'Insufficient_Weight'
 'OverWeight_Level_1' 'Obesity_Type_3' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'OverWeight_Level_1' 'Normal_Weight'
 'Obesity_Type_2' 'Normal_Weight' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Insufficient_Weight'
 'OverWeight_Level_2' 'Obesity_Type_2' 'Insufficient_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Normal_Weight']
                     precision    recall  f1-score   support
```

```
Insufficient_Weight       0.96      0.92      0.94        59
      Normal_Weight       0.83      0.95      0.89        61
     Obesity_Type_1       1.00      1.00      1.00        60
     Obesity_Type_2       0.98      0.89      0.93        55
     Obesity_Type_3       0.94      0.96      0.95        49
 OverWeight_Level_1       1.00      0.97      0.99        70
 OverWeight_Level_2       1.00      1.00      1.00        64

           accuracy                           0.96       418
          macro avg       0.96      0.96      0.96       418
       weighted avg       0.96      0.96      0.96       418
```

In [111…

```
from sklearn.linear_model import LogisticRegression

lr = LogisticRegression(random_state = 42,)
lr.fit(X_train,y_train)
y_pred = lr.predict(X_test)

print(lr.score(X_test,y_test))

cm = confusion_matrix(y_test,y_pred)


disp = ConfusionMatrixDisplay(confusion_matrix = cm,display_labels = label_mapping)
disp.plot(cmap = 'Greens',colorbar = True)
plt.title('LogisticRegression Prediction', weight = 'bold', fontsize = 10)
plt.xlabel('Prediction')
plt.ylabel('Prediction')
plt.xticks(rotation = 90)
plt.show()
print(y_pred)
print(classification_report(y_test,y_pred))
```

```
0.7511961722488039
```



```
['OverWeight_Level_1' 'OverWeight_Level_2' 'OverWeight_Level_1'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Obesity_Type_2'
 'OverWeight_Level_1' 'Obesity_Type_1' 'Insufficient_Weight'
 'Insufficient_Weight' 'OverWeight_Level_2' 'OverWeight_Level_1'
 'Obesity_Type_1' 'Obesity_Type_3' 'OverWeight_Level_1' 'Obesity_Type_2'
 'Obesity_Type_2' 'OverWeight_Level_2' 'Obesity_Type_2' 'Obesity_Type_3'
 'Obesity_Type_1' 'Obesity_Type_2' 'Insufficient_Weight'
 'Insufficient_Weight' 'OverWeight_Level_1' 'OverWeight_Level_1'
 'Insufficient_Weight' 'Insufficient_Weight' 'Insufficient_Weight'
 'OverWeight_Level_1' 'Normal_Weight' 'Normal_Weight' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Obesity_Type_1' 'Obesity_Type_2'
 'Obesity_Type_3' 'Insufficient_Weight' 'Insufficient_Weight'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Obesity_Type_2'
 'Obesity_Type_1' 'Obesity_Type_2' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'Normal_Weight' 'Insufficient_Weight'
 'Obesity_Type_1' 'OverWeight_Level_1' 'Insufficient_Weight'
 'OverWeight_Level_1' 'Obesity_Type_1' 'Obesity_Type_1' 'Obesity_Type_1'
 'Obesity_Type_2' 'OverWeight_Level_1' 'Obesity_Type_3' 'Obesity_Type_3'
 'Obesity_Type_3' 'OverWeight_Level_1' 'Insufficient_Weight'
 'OverWeight_Level_2' 'Normal_Weight' 'OverWeight_Level_2'
 'OverWeight_Level_1' 'Obesity_Type_2' 'OverWeight_Level_1'
 'OverWeight_Level_1' 'OverWeight_Level_2' 'Obesity_Type_2'
 'OverWeight_Level_1' 'Obesity_Type_3' 'Obesity_Type_2' 'Normal_Weight'
 'OverWeight_Level_2' 'Normal_Weight' 'Insufficient_Weight'
 'Insufficient_Weight' 'Obesity_Type_2' 'Obesity_Type_2' 'Obesity_Type_1'
 'OverWeight_Level_1' 'Normal_Weight' 'Insufficient_Weight'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Normal_Weight' 'Obesity_Type_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'OverWeight_Level_1' 'OverWeight_Level_2'
 'OverWeight_Level_1' 'Obesity_Type_1' 'OverWeight_Level_1'
 'Obesity_Type_1' 'Insufficient_Weight' 'Obesity_Type_3' 'Obesity_Type_3'
 'Obesity_Type_3' 'Obesity_Type_3' 'OverWeight_Level_1' 'Normal_Weight'
 'Normal_Weight' 'Obesity_Type_1' 'Obesity_Type_3' 'Normal_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_1' 'Insufficient_Weight'
 'Obesity_Type_1' 'OverWeight_Level_2' 'Obesity_Type_2'
 'OverWeight_Level_2' 'Normal_Weight' 'OverWeight_Level_2'
 'Insufficient_Weight' 'Obesity_Type_2' 'OverWeight_Level_1'
 'Insufficient_Weight' 'Obesity_Type_2' 'Obesity_Type_1' 'Obesity_Type_2'
 'Normal_Weight' 'Normal_Weight' 'Normal_Weight' 'OverWeight_Level_2'
 'Insufficient_Weight' 'Obesity_Type_3' 'Obesity_Type_1' 'Obesity_Type_2'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Normal_Weight' 'Obesity_Type_2'
 'Obesity_Type_2' 'Obesity_Type_1' 'Obesity_Type_3' 'OverWeight_Level_1'
 'Normal_Weight' 'Obesity_Type_1' 'Insufficient_Weight' 'Obesity_Type_2'
 'Obesity_Type_1' 'Obesity_Type_1' 'Obesity_Type_2' 'Normal_Weight'
 'Obesity_Type_1' 'Obesity_Type_1' 'OverWeight_Level_1'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Insufficient_Weight'
 'Obesity_Type_1' 'Obesity_Type_1' 'OverWeight_Level_1'
 'Insufficient_Weight' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Insufficient_Weight'
 'Insufficient_Weight' 'Obesity_Type_1' 'Obesity_Type_3' 'Normal_Weight'
 'OverWeight_Level_2' 'Insufficient_Weight' 'OverWeight_Level_2'
 'Obesity_Type_2' 'Insufficient_Weight' 'Obesity_Type_2' 'Obesity_Type_3'
 'Obesity_Type_3' 'Obesity_Type_3' 'Obesity_Type_2' 'Insufficient_Weight'
 'Obesity_Type_2' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'Insufficient_Weight' 'Obesity_Type_3' 'Normal_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_1' 'Obesity_Type_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'OverWeight_Level_1'
 'Obesity_Type_2' 'Obesity_Type_1' 'Normal_Weight' 'OverWeight_Level_1'
 'Obesity_Type_2' 'OverWeight_Level_1' 'Insufficient_Weight'
 'OverWeight_Level_1' 'Obesity_Type_1' 'Obesity_Type_3' 'Obesity_Type_2'
 'Obesity_Type_3' 'OverWeight_Level_1' 'OverWeight_Level_1'
 'Obesity_Type_3' 'Obesity_Type_2' 'OverWeight_Level_1' 'Normal_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Obesity_Type_1'
 'Obesity_Type_1' 'Obesity_Type_2' 'Obesity_Type_1' 'OverWeight_Level_1'
 'Insufficient_Weight' 'Obesity_Type_3' 'Obesity_Type_1'
 'OverWeight_Level_1' 'OverWeight_Level_2' 'Obesity_Type_3'
 'Obesity_Type_1' 'Obesity_Type_3' 'OverWeight_Level_1'
 'OverWeight_Level_1' 'Obesity_Type_3' 'Obesity_Type_3' 'Obesity_Type_2'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Obesity_Type_3'
 'Obesity_Type_1' 'OverWeight_Level_1' 'OverWeight_Level_2'
 'Insufficient_Weight' 'Obesity_Type_3' 'Obesity_Type_1' 'Obesity_Type_2'
 'OverWeight_Level_2' 'Normal_Weight' 'Normal_Weight' 'Obesity_Type_3'
 'OverWeight_Level_2' 'Obesity_Type_1' 'Obesity_Type_2' 'Obesity_Type_1'
 'Normal_Weight' 'Normal_Weight' 'OverWeight_Level_1'
 'Insufficient_Weight' 'Obesity_Type_1' 'Normal_Weight' 'Obesity_Type_1'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Obesity_Type_2'
 'OverWeight_Level_2' 'Obesity_Type_1' 'OverWeight_Level_1'
 'Obesity_Type_2' 'OverWeight_Level_1' 'Insufficient_Weight'
 'OverWeight_Level_1' 'Obesity_Type_1' 'Insufficient_Weight'
 'Obesity_Type_1' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'Normal_Weight' 'OverWeight_Level_2' 'Obesity_Type_3'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Obesity_Type_2' 'Obesity_Type_3'
 'OverWeight_Level_1' 'Obesity_Type_3' 'Normal_Weight'
 'OverWeight_Level_1' 'Obesity_Type_2' 'OverWeight_Level_2'
 'Obesity_Type_1' 'OverWeight_Level_2' 'Insufficient_Weight'
 'Obesity_Type_2' 'Obesity_Type_3' 'OverWeight_Level_1' 'Obesity_Type_2'
 'Obesity_Type_3' 'Insufficient_Weight' 'Normal_Weight' 'Obesity_Type_1'
 'Insufficient_Weight' 'Normal_Weight' 'OverWeight_Level_2'
 'Insufficient_Weight' 'Insufficient_Weight' 'Obesity_Type_2'
 'Insufficient_Weight' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'Insufficient_Weight' 'Obesity_Type_2'
 'Obesity_Type_1' 'Insufficient_Weight' 'Obesity_Type_1' 'Obesity_Type_3'
 'Obesity_Type_1' 'Normal_Weight' 'Obesity_Type_2' 'Insufficient_Weight'
 'Obesity_Type_3' 'OverWeight_Level_2' 'Normal_Weight' 'Obesity_Type_2'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Insufficient_Weight'
 'Obesity_Type_1' 'OverWeight_Level_1' 'Insufficient_Weight'
 'Insufficient_Weight' 'Obesity_Type_1' 'Insufficient_Weight'
 'Obesity_Type_1' 'OverWeight_Level_2' 'Normal_Weight'
 'Insufficient_Weight' 'Insufficient_Weight' 'Obesity_Type_1'
 'Insufficient_Weight' 'OverWeight_Level_1' 'Normal_Weight'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Obesity_Type_2'
 'Normal_Weight' 'Insufficient_Weight' 'Obesity_Type_1' 'Obesity_Type_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Obesity_Type_2'
 'Obesity_Type_2' 'Obesity_Type_2' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'Insufficient_Weight' 'Obesity_Type_3'
 'Obesity_Type_2' 'Obesity_Type_1' 'OverWeight_Level_2' 'Obesity_Type_1'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'OverWeight_Level_1'
 'Obesity_Type_1' 'Obesity_Type_1' 'OverWeight_Level_2'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Obesity_Type_1'
 'OverWeight_Level_1' 'Obesity_Type_3' 'Insufficient_Weight'
 'Insufficient_Weight' 'Obesity_Type_3' 'Obesity_Type_1' 'Normal_Weight'
 'OverWeight_Level_1' 'Obesity_Type_3' 'Obesity_Type_1'
 'Insufficient_Weight' 'Obesity_Type_2' 'Insufficient_Weight'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Obesity_Type_1' 'Obesity_Type_3'
 'Insufficient_Weight' 'OverWeight_Level_1' 'Obesity_Type_3'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Normal_Weight'
 'Insufficient_Weight' 'Obesity_Type_3' 'Obesity_Type_1' 'Obesity_Type_1'
 'Obesity_Type_2' 'OverWeight_Level_1' 'Obesity_Type_3'
 'OverWeight_Level_1' 'OverWeight_Level_2' 'OverWeight_Level_1'
 'Obesity_Type_3' 'OverWeight_Level_1' 'Normal_Weight'
 'OverWeight_Level_1' 'OverWeight_Level_2' 'Obesity_Type_3'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Obesity_Type_3'
 'Insufficient_Weight' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'Normal_Weight']
                     precision    recall  f1-score   support
```

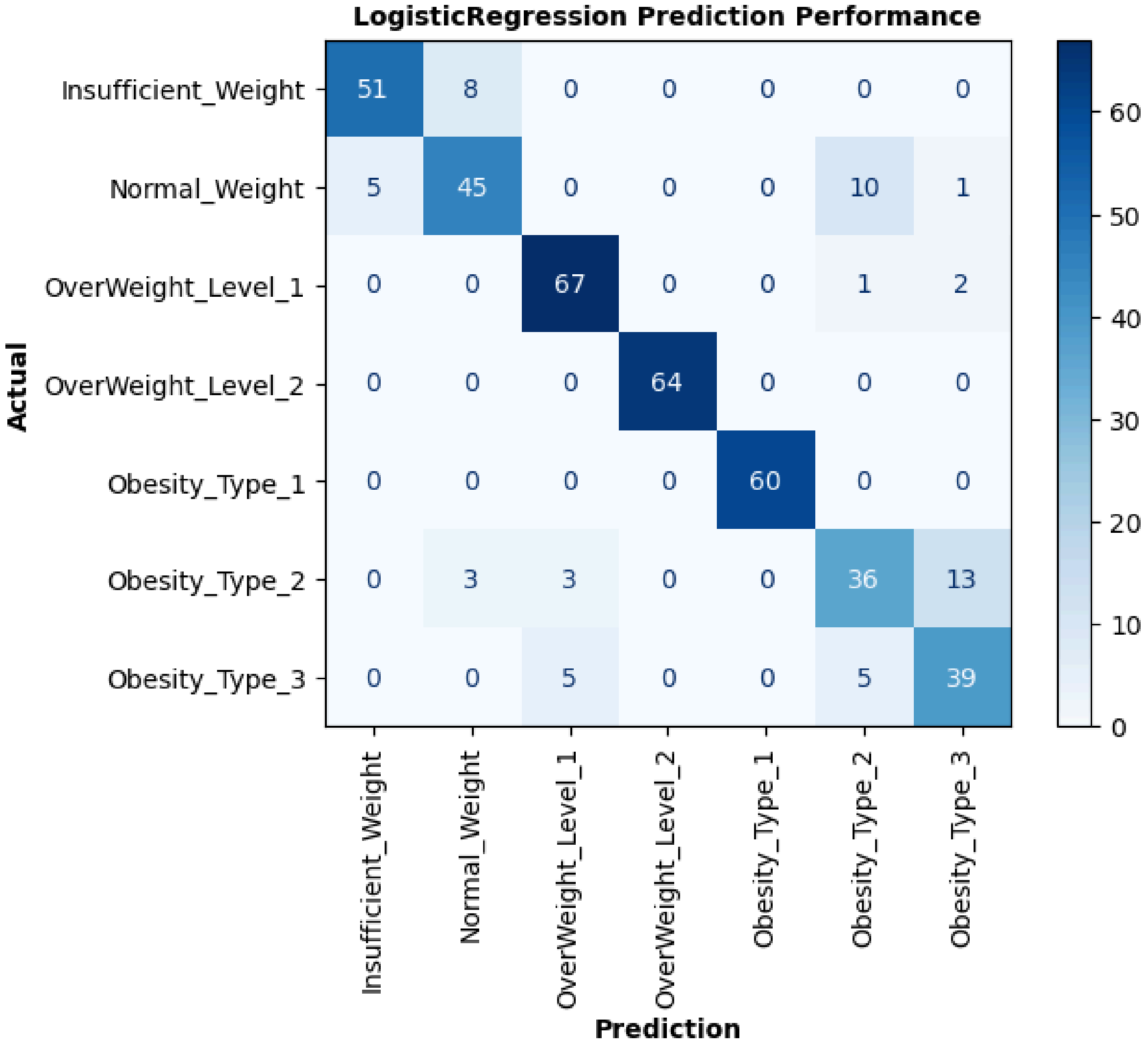
```
Insufficient_Weight       0.76      0.85      0.80        59
      Normal_Weight       0.65      0.43      0.51        61
     Obesity_Type_1       0.98      1.00      0.99        60
     Obesity_Type_2       0.58      0.58      0.58        55
     Obesity_Type_3       0.55      0.59      0.57        49
 OverWeight_Level_1       0.73      0.76      0.74        70
 OverWeight_Level_2       0.91      1.00      0.96        64

           accuracy                           0.75       418
          macro avg       0.74      0.74      0.74       418
       weighted avg       0.75      0.75      0.74       418
```

Tunning The Logistic Regression Model For PredictionTo be More Accurate

In [112…

```python
from sklearn.preprocessing import  PolynomialFeatures
from sklearn.feature_selection import SelectKBest, chi2




#Feature Interaction (Polynomial Features)
poly = PolynomialFeatures(degree=2, interaction_only=True, include_bias=False)
X_train_poly = poly.fit_transform(X_train)
X_test_poly = poly.transform(X_test)

# Feature Selection (SelectKBest)
selector = SelectKBest(score_func=chi2, k='all')  # Choose 'all' or a specific numb
X_train_selected = selector.fit_transform(np.abs(X_train_poly), y_train)  # Use abs
X_test_selected = selector.transform(np.abs(X_test_poly))

# Logistic Regression
lr = LogisticRegression(random_state=42, max_iter=500)
lr.fit(X_train_selected, y_train)
y_pred = lr.predict(X_test_selected)

# Model Performance Metrics
print(f"Accuracy: {lr.score(X_test_selected, y_test)}")

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred, labels = ['Insufficient_Weight','Normal_Weigh
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=['Insufficient_We
disp.plot(cmap='Blues', colorbar=True)
plt.title('LogisticRegression Prediction Performance', weight = 'bold', fontsize =
plt.xlabel('Prediction', weight = 'bold')
plt.ylabel('Actual', weight = 'bold')
plt.xticks(rotation=90)
plt.show()

# Classification Report
print("Predictions:", y_pred)
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.8660287081339713
```


LogisticRegression Prediction Performance
Actual
Prediction
Insufficient_Weight
Normal_Weight
OverWeight_Level_1
OverWeight_Level_2
Obesity_Type_1
Obesity_Type_2
Obesity_Type_3
51 8 0 0 0 0 0
5 45 0 0 0 10 1
0 0 67 0 0 1 2
0 0 0 64 0 0 0
0 0 0 0 60 0 0
0 3 3 0 0 36 13
0 0 5 0 0 5 39
0
10
20
30
40
50
60

```
Predictions: ['OverWeight_Level_1' 'OverWeight_Level_2' 'OverWeight_Level_1'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Obesity_Type_2'
 'Obesity_Type_3' 'Obesity_Type_1' 'Normal_Weight' 'Normal_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_1' 'Obesity_Type_1'
 'Normal_Weight' 'Obesity_Type_3' 'Obesity_Type_2' 'Obesity_Type_2'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Obesity_Type_3' 'Obesity_Type_1'
 'Obesity_Type_3' 'Normal_Weight' 'Insufficient_Weight'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Insufficient_Weight'
 'Insufficient_Weight' 'Insufficient_Weight' 'OverWeight_Level_1'
 'Normal_Weight' 'Obesity_Type_2' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Obesity_Type_1' 'Obesity_Type_2'
 'Normal_Weight' 'Insufficient_Weight' 'Normal_Weight' 'Normal_Weight'
 'OverWeight_Level_1' 'Obesity_Type_2' 'Obesity_Type_1' 'Normal_Weight'
 'Obesity_Type_2' 'OverWeight_Level_2' 'Obesity_Type_2' 'Normal_Weight'
 'Obesity_Type_1' 'Obesity_Type_3' 'Insufficient_Weight'
 'OverWeight_Level_1' 'Obesity_Type_1' 'Obesity_Type_1' 'Obesity_Type_1'
 'Obesity_Type_2' 'OverWeight_Level_1' 'Obesity_Type_3' 'Obesity_Type_3'
 'Obesity_Type_3' 'OverWeight_Level_1' 'Normal_Weight'
 'OverWeight_Level_2' 'Normal_Weight' 'OverWeight_Level_2'
 'OverWeight_Level_1' 'Obesity_Type_2' 'OverWeight_Level_1'
 'OverWeight_Level_1' 'Obesity_Type_3' 'Obesity_Type_2'
 'OverWeight_Level_1' 'Obesity_Type_2' 'OverWeight_Level_1'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Normal_Weight'
 'Insufficient_Weight' 'Insufficient_Weight' 'Obesity_Type_2'
 'Obesity_Type_2' 'Obesity_Type_1' 'OverWeight_Level_1' 'Normal_Weight'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Obesity_Type_3'
 'Normal_Weight' 'OverWeight_Level_1' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'OverWeight_Level_1' 'OverWeight_Level_2' 'OverWeight_Level_1'
 'Obesity_Type_1' 'Obesity_Type_3' 'Obesity_Type_1' 'Normal_Weight'
 'Obesity_Type_3' 'OverWeight_Level_1' 'Obesity_Type_3'
 'Insufficient_Weight' 'Normal_Weight' 'Normal_Weight' 'Normal_Weight'
 'Obesity_Type_1' 'OverWeight_Level_1' 'Normal_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_1' 'Insufficient_Weight'
 'Obesity_Type_1' 'OverWeight_Level_2' 'Obesity_Type_2'
 'OverWeight_Level_2' 'Normal_Weight' 'OverWeight_Level_2'
 'Insufficient_Weight' 'Obesity_Type_2' 'OverWeight_Level_1'
 'Insufficient_Weight' 'Obesity_Type_2' 'Obesity_Type_1' 'Obesity_Type_2'
 'Normal_Weight' 'Obesity_Type_2' 'Normal_Weight' 'OverWeight_Level_2'
 'Normal_Weight' 'Obesity_Type_3' 'OverWeight_Level_1' 'Obesity_Type_2'
 'OverWeight_Level_1' 'Obesity_Type_3' 'Normal_Weight' 'Obesity_Type_2'
 'Obesity_Type_2' 'Obesity_Type_1' 'Obesity_Type_3' 'OverWeight_Level_1'
 'Normal_Weight' 'Obesity_Type_1' 'Normal_Weight' 'Obesity_Type_3'
 'Obesity_Type_1' 'Obesity_Type_1' 'Obesity_Type_2' 'Insufficient_Weight'
 'Obesity_Type_1' 'Obesity_Type_1' 'OverWeight_Level_1' 'Normal_Weight'
 'OverWeight_Level_2' 'Insufficient_Weight' 'Obesity_Type_1'
 'Obesity_Type_1' 'OverWeight_Level_1' 'Normal_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Insufficient_Weight'
 'OverWeight_Level_2' 'Normal_Weight' 'Normal_Weight' 'Obesity_Type_1'
 'Obesity_Type_3' 'Normal_Weight' 'OverWeight_Level_1'
 'Insufficient_Weight' 'OverWeight_Level_2' 'Obesity_Type_2'
 'Insufficient_Weight' 'Obesity_Type_2' 'Obesity_Type_3' 'Obesity_Type_3'
 'Obesity_Type_3' 'Normal_Weight' 'Insufficient_Weight' 'Obesity_Type_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Insufficient_Weight'
 'Obesity_Type_3' 'Normal_Weight' 'OverWeight_Level_2' 'Obesity_Type_3'
 'OverWeight_Level_1' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'OverWeight_Level_1' 'Obesity_Type_2' 'Obesity_Type_1' 'Normal_Weight'
 'OverWeight_Level_1' 'Obesity_Type_2' 'OverWeight_Level_1'
 'Normal_Weight' 'OverWeight_Level_1' 'Obesity_Type_1'
 'OverWeight_Level_1' 'Obesity_Type_2' 'Obesity_Type_3'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Obesity_Type_3'
 'Obesity_Type_2' 'OverWeight_Level_1' 'Normal_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Obesity_Type_3'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Obesity_Type_1'
 'Obesity_Type_1' 'Obesity_Type_2' 'Obesity_Type_1' 'OverWeight_Level_1'
 'Insufficient_Weight' 'Obesity_Type_3' 'Obesity_Type_1'
 'OverWeight_Level_1' 'OverWeight_Level_2' 'Obesity_Type_3'
 'Obesity_Type_1' 'Obesity_Type_3' 'Obesity_Type_2' 'OverWeight_Level_1'
 'Obesity_Type_3' 'Obesity_Type_3' 'Obesity_Type_2' 'Insufficient_Weight'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Obesity_Type_1'
 'OverWeight_Level_1' 'OverWeight_Level_2' 'Insufficient_Weight'
 'Obesity_Type_3' 'Obesity_Type_1' 'Obesity_Type_2' 'OverWeight_Level_2'
 'Insufficient_Weight' 'Normal_Weight' 'Obesity_Type_3'
 'OverWeight_Level_2' 'Obesity_Type_1' 'Obesity_Type_2' 'Obesity_Type_1'
 'Normal_Weight' 'Normal_Weight' 'OverWeight_Level_1'
 'Insufficient_Weight' 'Obesity_Type_1' 'Normal_Weight' 'Obesity_Type_1'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Obesity_Type_2'
 'OverWeight_Level_2' 'Obesity_Type_1' 'OverWeight_Level_1'
 'Obesity_Type_2' 'Normal_Weight' 'Insufficient_Weight'
 'OverWeight_Level_1' 'Obesity_Type_1' 'Insufficient_Weight'
 'Obesity_Type_1' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'Normal_Weight' 'OverWeight_Level_1' 'Obesity_Type_2'
 'OverWeight_Level_2' 'Insufficient_Weight' 'Normal_Weight'
 'Obesity_Type_3' 'Obesity_Type_3' 'OverWeight_Level_1' 'Obesity_Type_2'
 'OverWeight_Level_1' 'Obesity_Type_2' 'OverWeight_Level_2'
 'Obesity_Type_1' 'OverWeight_Level_2' 'Insufficient_Weight'
 'Obesity_Type_2' 'OverWeight_Level_1' 'OverWeight_Level_1'
 'Obesity_Type_3' 'Obesity_Type_3' 'Normal_Weight' 'Insufficient_Weight'
 'Obesity_Type_1' 'Insufficient_Weight' 'Obesity_Type_2'
 'OverWeight_Level_1' 'Insufficient_Weight' 'Insufficient_Weight'
 'Insufficient_Weight' 'Insufficient_Weight' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Insufficient_Weight'
 'Obesity_Type_2' 'Obesity_Type_1' 'Insufficient_Weight' 'Obesity_Type_1'
 'Obesity_Type_3' 'Obesity_Type_1' 'Normal_Weight' 'Obesity_Type_2'
 'Insufficient_Weight' 'Obesity_Type_3' 'OverWeight_Level_2'
 'Obesity_Type_3' 'Normal_Weight' 'OverWeight_Level_1'
 'OverWeight_Level_1' 'Insufficient_Weight' 'Obesity_Type_1'
 'OverWeight_Level_1' 'Insufficient_Weight' 'Insufficient_Weight'
 'Obesity_Type_1' 'Insufficient_Weight' 'Obesity_Type_1'
 'OverWeight_Level_2' 'Normal_Weight' 'Insufficient_Weight'
 'Insufficient_Weight' 'Obesity_Type_1' 'Normal_Weight'
 'OverWeight_Level_1' 'Normal_Weight' 'OverWeight_Level_1'
 'OverWeight_Level_1' 'Obesity_Type_2' 'Obesity_Type_2'
 'Insufficient_Weight' 'Obesity_Type_1' 'Obesity_Type_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Obesity_Type_2'
 'Obesity_Type_2' 'Obesity_Type_2' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'Normal_Weight' 'Obesity_Type_3' 'Obesity_Type_2'
 'Obesity_Type_1' 'OverWeight_Level_2' 'Obesity_Type_1'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'OverWeight_Level_1'
 'Obesity_Type_1' 'Obesity_Type_1' 'OverWeight_Level_2'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Obesity_Type_1'
 'OverWeight_Level_1' 'Obesity_Type_3' 'Insufficient_Weight'
 'Insufficient_Weight' 'Obesity_Type_3' 'Obesity_Type_1' 'Normal_Weight'
 'Obesity_Type_3' 'OverWeight_Level_1' 'Obesity_Type_1'
 'Insufficient_Weight' 'Obesity_Type_2' 'Insufficient_Weight'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Obesity_Type_1' 'Obesity_Type_3'
 'Insufficient_Weight' 'OverWeight_Level_1' 'Obesity_Type_3'
 'OverWeight_Level_1' 'OverWeight_Level_1' 'Normal_Weight'
 'Insufficient_Weight' 'Normal_Weight' 'Obesity_Type_1' 'Obesity_Type_1'
 'Insufficient_Weight' 'OverWeight_Level_1' 'Obesity_Type_3'
 'OverWeight_Level_1' 'OverWeight_Level_2' 'OverWeight_Level_1'
 'Obesity_Type_3' 'Obesity_Type_3' 'Normal_Weight' 'OverWeight_Level_1'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Insufficient_Weight'
 'OverWeight_Level_2' 'Obesity_Type_3' 'Insufficient_Weight'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'OverWeight_Level_2'
 'OverWeight_Level_2' 'OverWeight_Level_2' 'Normal_Weight']
                     precision    recall  f1-score   support
```

```
Insufficient_Weight       0.91      0.86      0.89        59
      Normal_Weight       0.80      0.74      0.77        61
     Obesity_Type_1       1.00      1.00      1.00        60
     Obesity_Type_2       0.69      0.65      0.67        55
     Obesity_Type_3       0.71      0.80      0.75        49
 OverWeight_Level_1       0.89      0.96      0.92        70
 OverWeight_Level_2       1.00      1.00      1.00        64

           accuracy                           0.87       418
          macro avg       0.86      0.86      0.86       418
       weighted avg       0.87      0.87      0.87       418
```

In [ ]:

In [ ]: