sity-levels-using-machine-learning

February 14, 2025

0.0.1 Possible Data Preprocessing Steps

1. **Handling Missing Values** – Check if any attributes contain missing values and handle them accordingly.
2. **Encoding Categorical Features** – Convert categorical variables (`Gender`, `family_history`, `FAVC`, `CAEC`, `SMOKE`, `SCC`, `CALC`, `MTRANS`, `Obesity_level`) into numerical values.
3. **Feature Scaling** – Standardize `Age`, `Height`, `Weight`, `CH2O`, `FAF`, and `TUE` since they have different scales.
4. **Balancing Classes** – Check for class imbalance in the `Obesity_level` column and apply techniques like SMOTE if needed.

0.0.2 Exploratory Data Analysis (EDA)

- **Distribution of Obesity Levels** – Visualize the target variable's distribution using bar plots.
- **Correlation Analysis** – Use a heatmap to examine relationships between features.
- **Impact of Eating Habits & Lifestyle** – Analyze the effect of `FAVC`, `FCVC`, `CH2O`, `CAEC`, `FAF`, and `TUE` on obesity levels.
- **Comparison by Gender & Age** – Investigate how obesity levels vary by gender and age groups.

0.0.3 Modeling Approach

- **Classification Models** – Try Logistic Regression, Decision Trees, Random Forest, XGBoost, or Neural Networks.
- **Feature Importance** – Identify the most influential factors for obesity prediction.
- **Hyperparameter Tuning** – Use GridSearchCV or RandomizedSearchCV to optimize model performance.

[29]:
```
import numpy as np
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

```python
from sklearn.metrics import accuracy_score, classification_report,␣
↪confusion_matrix, roc_auc_score, f1_score
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from xgboost import XGBClassifier
from lightgbm import LGBMClassifier
from catboost import CatBoostClassifier
```

[12]:
```python
df = pd.read_csv("/kaggle/input/obesity-prediction/Obesity prediction.csv")
```

[13]:
```python
df.head()
```

[13]:
```
   Gender   Age  Height  Weight family_history 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

  SMOKE  CH2O  SCC  FAF  TUE        CALC                 MTRANS  \
0    no   2.0   no  0.0  1.0          no  Public_Transportation
1   yes   3.0  yes  3.0  0.0   Sometimes  Public_Transportation
2    no   2.0   no  2.0  1.0  Frequently  Public_Transportation
3    no   2.0   no  2.0  0.0  Frequently                Walking
4    no   2.0   no  0.0  0.0   Sometimes  Public_Transportation

               Obesity
0        Normal_Weight
1        Normal_Weight
2        Normal_Weight
3   Overweight_Level_I
4  Overweight_Level_II
```

[14]:
```python
df.isnull().sum()
```

[14]:
```
Gender            0
Age               0
Height            0
Weight            0
family_history    0
FAVC              0
FCVC              0
NCP               0
```

```
CAEC              0
SMOKE             0
CH2O              0
SCC               0
FAF               0
TUE               0
CALC              0
MTRANS            0
Obesity           0
dtype: int64
```

[15]:
```
df.shape
```

[15]:
```
(2111, 17)
```

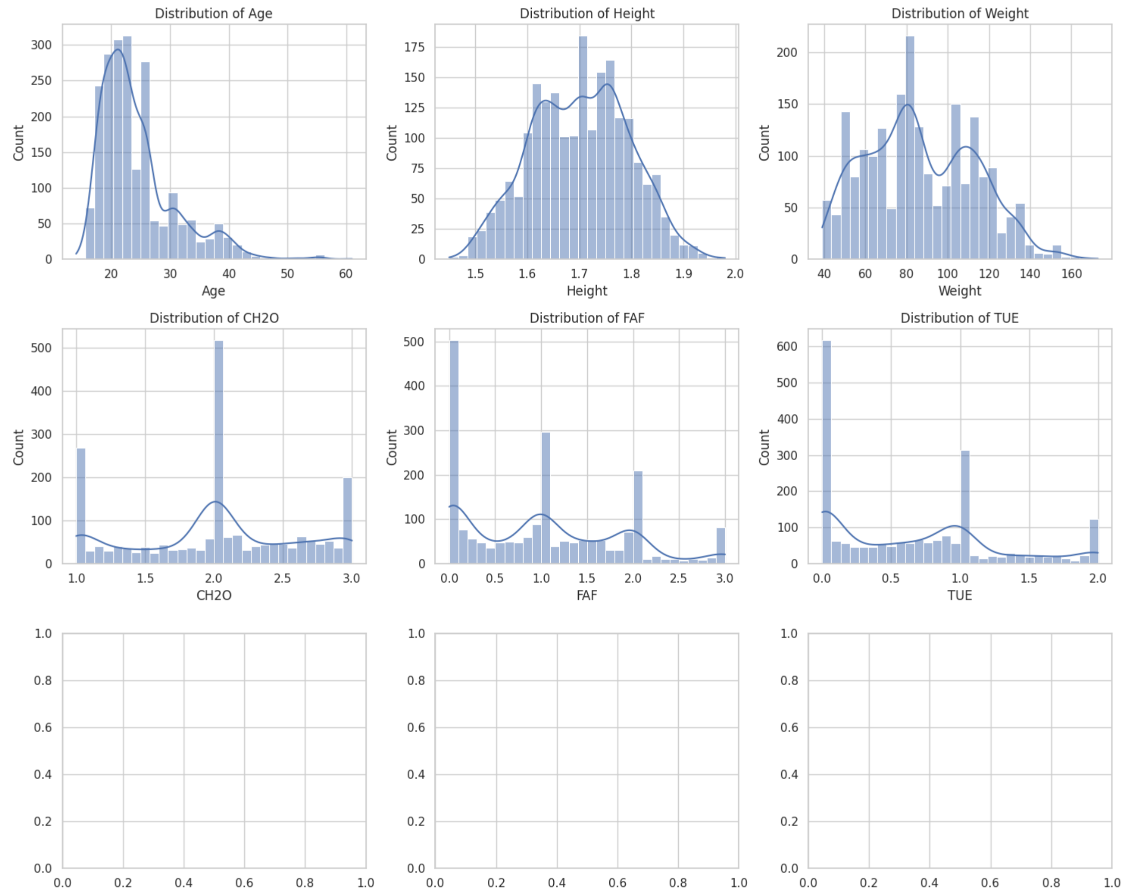
[16]:
```
# Set style
sns.set(style="whitegrid")

# 1. Checking dataset info and summary
print(df.info())
print(df.describe())
```

```
<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  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  Obesity         2111 non-null   object
dtypes: float64(8), object(9)
memory usage: 280.5+ KB
None
```

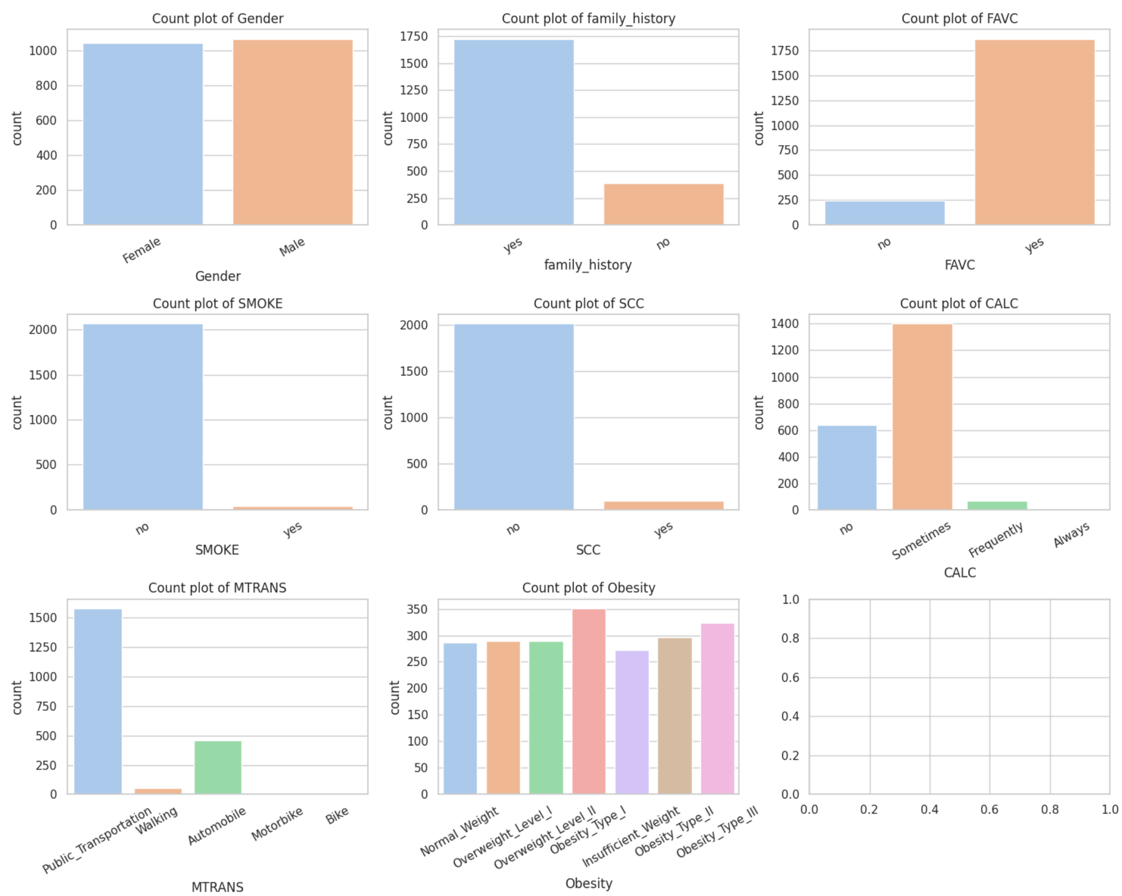
```
               Age       Height       Weight         FCVC          NCP  \
count  2111.000000  2111.000000  2111.000000  2111.000000  2111.000000
mean     24.312600     1.701677    86.586058     2.419043     2.685628
std       6.345968     0.093305    26.191172     0.533927     0.778039
min      14.000000     1.450000    39.000000     1.000000     1.000000
25%      19.947192     1.630000    65.473343     2.000000     2.658738
50%      22.777890     1.700499    83.000000     2.385502     3.000000
75%      26.000000     1.768464   107.430682     3.000000     3.000000
max      61.000000     1.980000   173.000000     3.000000     4.000000

              CH2O          FAF          TUE
count  2111.000000  2111.000000  2111.000000
mean      2.008011     1.010298     0.657866
std       0.612953     0.850592     0.608927
min       1.000000     0.000000     0.000000
25%       1.584812     0.124505     0.000000
50%       2.000000     1.000000     0.625350
75%       2.477420     1.666678     1.000000
max       3.000000     3.000000     2.000000
```

[17]:

```python
# 2. Univariate Analysis (Histograms, KDE, Count plots)
fig, axes = plt.subplots(3, 3, figsize=(15, 12))
columns = ['Age', 'Height', 'Weight', 'CH2O', 'FAF', 'TUE']

for i, col in enumerate(columns):
    sns.histplot(df[col], kde=True, bins=30, ax=axes[i//3, i%3])
    axes[i//3, i%3].set_title(f'Distribution of {col}')

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

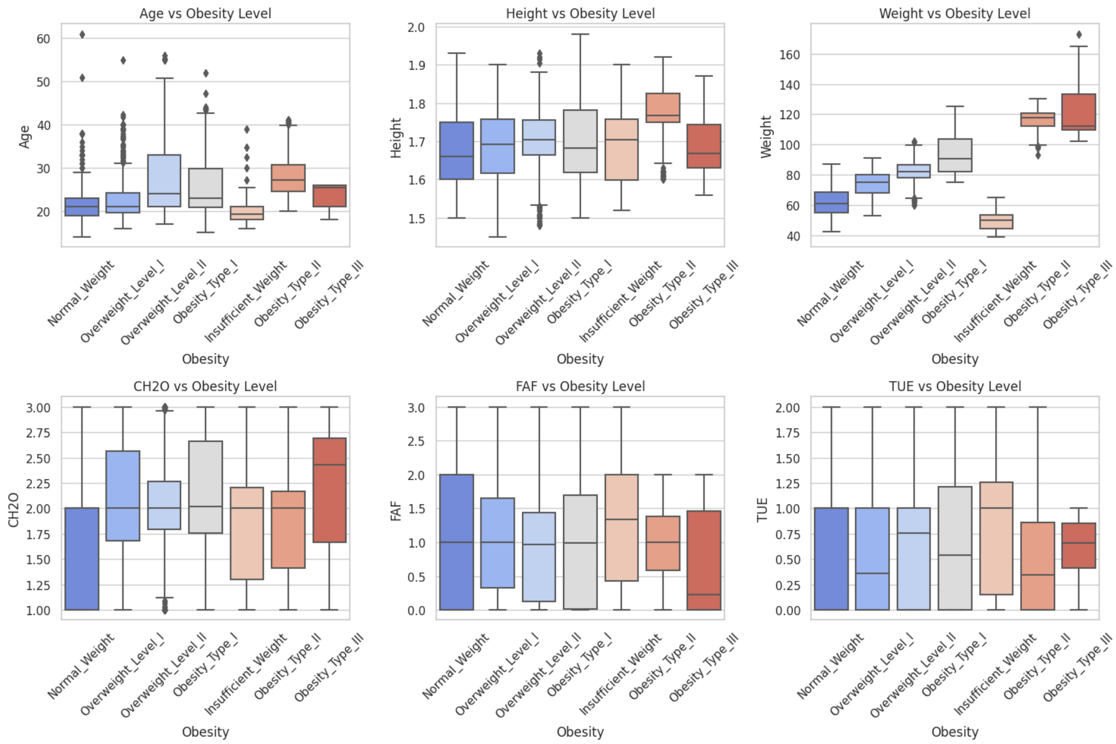


[18]:
```
# 3. Count plots for categorical variables
categorical_features = ['Gender', 'family_history', 'FAVC', 'SMOKE', 'SCC',␣
    ↪'CALC', 'MTRANS', 'Obesity']
fig, axes = plt.subplots(3, 3, figsize=(15, 12))

for i, col in enumerate(categorical_features):
    sns.countplot(data=df, x=col, ax=axes[i//3, i%3], palette='pastel')
    axes[i//3, i%3].set_xticklabels(axes[i//3, i%3].get_xticklabels(),␣
    ↪rotation=30)
    axes[i//3, i%3].set_title(f'Count plot of {col}')

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

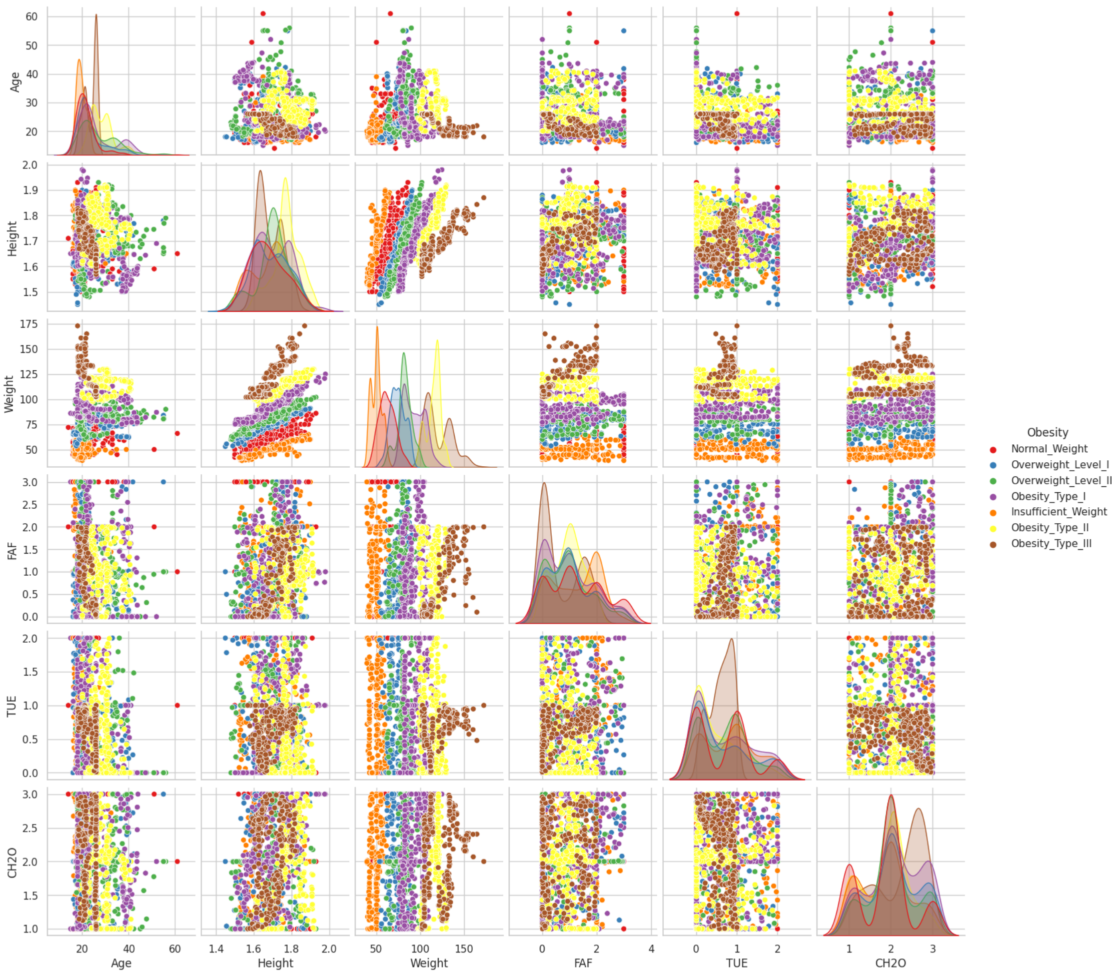


[20]:

```python
# 4. Bivariate Analysis (Box Plots, Scatter Plots)
fig, axes = plt.subplots(2, 3, figsize=(15, 10))
num_vars = ['Age', 'Height', 'Weight', 'CH2O', 'FAF', 'TUE']

for i, col in enumerate(num_vars):
    sns.boxplot(x='Obesity', y=col, data=df, ax=axes[i//3, i%3], palette='coolwarm')
    axes[i//3, i%3].set_xticklabels(axes[i//3, i%3].get_xticklabels(), rotation=45)
    axes[i//3, i%3].set_title(f'{col} vs Obesity Level')

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



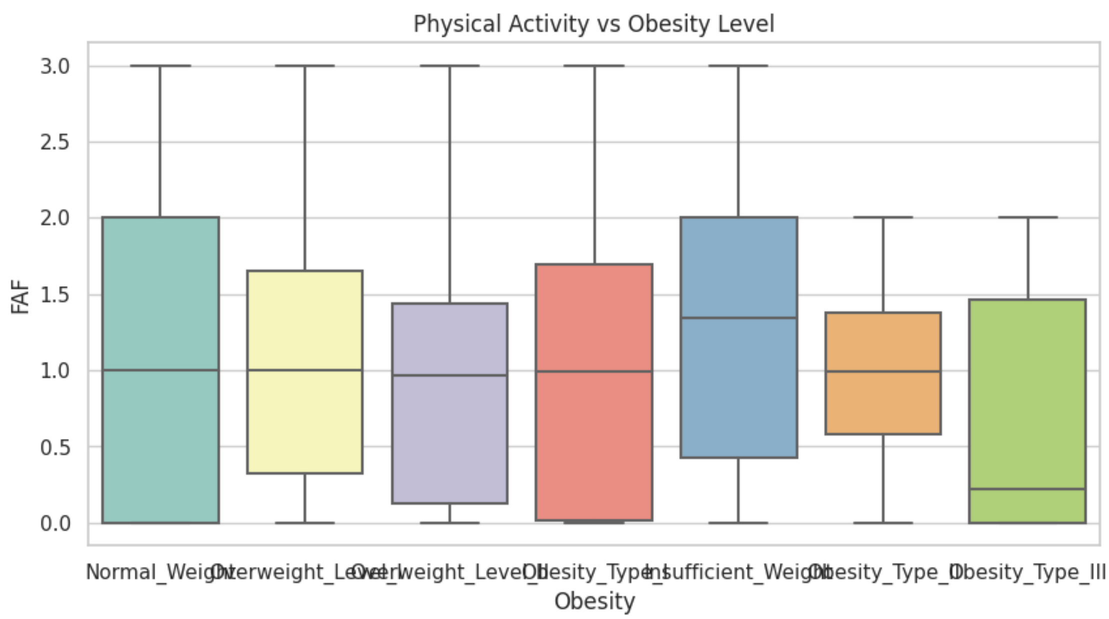
[22]:
```
# 5. Pairplot for numerical variables
sns.pairplot(df[['Age', 'Height', 'Weight', 'FAF', 'TUE', 'CH2O', 'Obesity']],␣
↪hue='Obesity', palette='Set1')
plt.show()
```



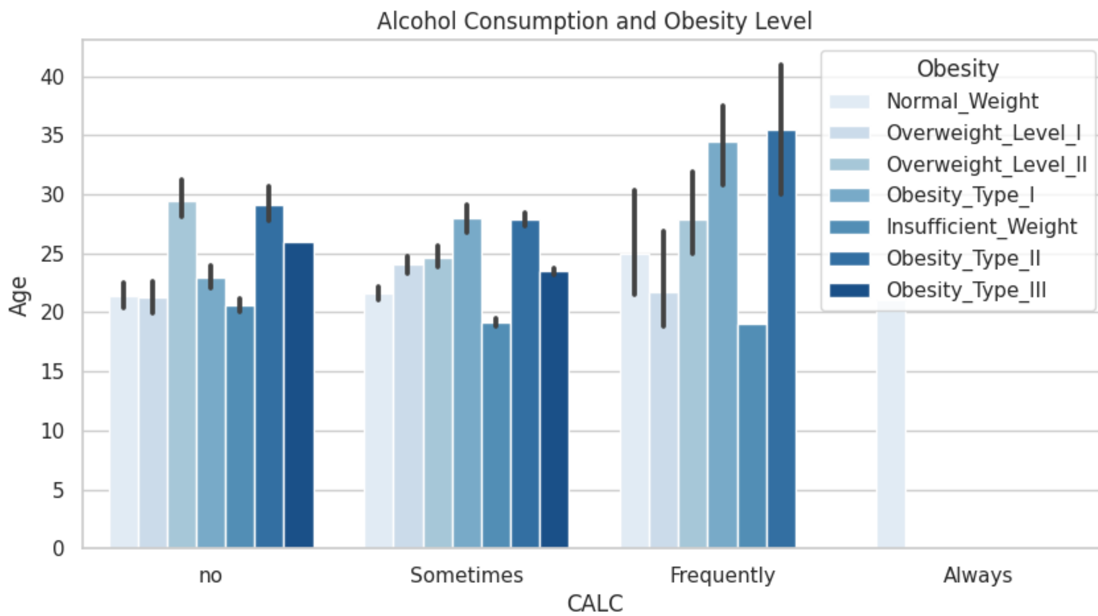
[24]:
```
# 7. Violin Plots
plt.figure(figsize=(12, 6))
sns.violinplot(x='Obesity', y='Weight', data=df, palette='muted')
plt.title('Weight Distribution by Obesity Level')
plt.show()
```

[25]:

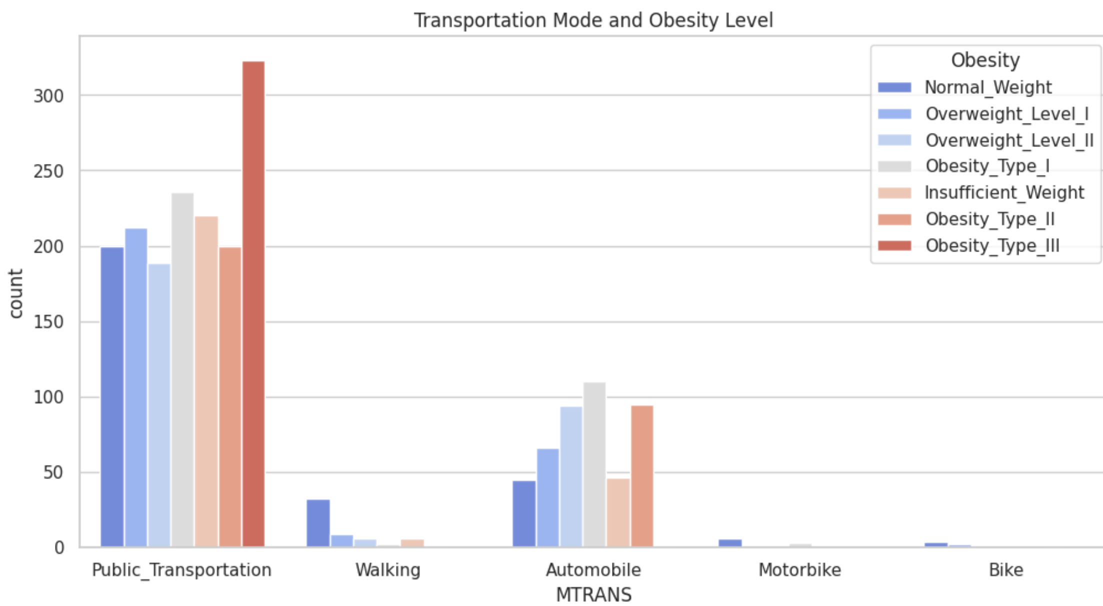
```
# 8. Box plot of Physical Activity Frequency vs Obesity Level
plt.figure(figsize=(10, 5))
sns.boxplot(x='Obesity', y='FAF', data=df, palette='Set3')
plt.title('Physical Activity vs Obesity Level')
plt.show()
```



[27]:
```python
# 9. Bar plot of Alcohol Consumption vs Obesity Level
plt.figure(figsize=(10, 5))
sns.barplot(x='CALC', y='Age', hue='Obesity', data=df, palette='Blues')
plt.title('Alcohol Consumption and Obesity Level')
plt.show()
```



[28]:
```python
# 10. Count plot of Transportation vs Obesity Level
plt.figure(figsize=(12, 6))
sns.countplot(x='MTRANS', hue='Obesity', data=df, palette='coolwarm')
plt.title('Transportation Mode and Obesity Level')
plt.show()
```



[38]:
```
# Encoding categorical variables
label_encoders = {}
categorical_features = ['Gender','CAEC', 'family_history', 'FAVC', 'SMOKE',␣
↪'SCC', 'CALC', 'MTRANS', 'Obesity']
for col in categorical_features:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le
```

[39]:
```
# Feature Engineering: Creating BMI column
df['BMI'] = df['Weight'] / (df['Height'] ** 2)
X = df.drop(columns=['Obesity'])
y = df['Obesity']
```

[40]:
```
# Splitting dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)

# Standardizing numerical features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

[41]:
```
X_train
```

[41]:
```
array([[-1.01311923, -0.53264595, -0.76802941, …, -0.52565676,
         0.50676114, -0.89297113],
       [-1.01311923, -0.54423543,  0.54607823, …, -0.52565676,
         0.50676114,  1.72897222],
       [-1.01311923, -0.23925802, -0.4278957 , …, -0.52565676,
         0.50676114,  0.02553165],
       …,
       [-1.01311923, -0.22534243, -0.55353551, …,  1.41169828,
         0.50676114, -0.04053735],
       [-1.01311923, -0.22377429, -0.78767706, …,  1.41169828,
         0.50676114,  0.27151926],
       [ 0.98705066, -0.68708178,  1.24263647, …, -0.52565676,
         0.50676114, -0.49825615]])
```

[42]:
```python
# Initialize models
models = {
    "Logistic Regression": LogisticRegression(),
    "Random Forest": RandomForestClassifier(),
    "Gradient Boosting": GradientBoostingClassifier(),
    "SVM": SVC(probability=True),
    "KNN": KNeighborsClassifier(),
    "Decision Tree": DecisionTreeClassifier(),
    "Naive Bayes": GaussianNB(),
    "XGBoost": XGBClassifier(),
    "LightGBM": LGBMClassifier(),
    "CatBoost": CatBoostClassifier(verbose=0)
}
```

[43]:
```python
# Train and evaluate models
performance = []
for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    y_pred_prob = model.predict_proba(X_test) if hasattr(model,␣
 ↪"predict_proba") else None

    accuracy = accuracy_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred, average='weighted')
    roc_auc = roc_auc_score(y_test, y_pred_prob, multi_class='ovr') if␣
 ↪y_pred_prob is not None else None
    cm = confusion_matrix(y_test, y_pred)

    print(f"{name} Performance:")
    print(f"Accuracy: {accuracy:.4f}")
    print(f"F1 Score: {f1:.4f}")
    if roc_auc is not None:
        print(f"ROC AUC Score: {roc_auc:.4f}")
```

```python
    print("Classification Report:")
    print(classification_report(y_test, y_pred))
    print("Confusion Matrix:")
    print(cm)
    print("-" * 50)

    performance.append([name, accuracy, f1, roc_auc])
```

```
Logistic Regression Performance:
Accuracy: 0.9031
F1 Score: 0.9020
ROC AUC Score: 0.9928
Classification Report:
              precision    recall  f1-score   support

           0       0.89      1.00      0.94        56
           1       0.94      0.73      0.82        62
           2       0.97      0.92      0.95        78
           3       0.92      0.98      0.95        58
           4       1.00      1.00      1.00        63
           5       0.79      0.80      0.80        56
           6       0.79      0.88      0.83        50

    accuracy                           0.90       423
   macro avg       0.90      0.90      0.90       423
weighted avg       0.91      0.90      0.90       423

Confusion Matrix:
[[56  0  0  0  0  0  0]
 [ 7 45  0  0  0  7  3]
 [ 0  0 72  5  0  0  1]
 [ 0  0  1 57  0  0  0]
 [ 0  0  0  0 63  0  0]
 [ 0  3  0  0  0 45  8]
 [ 0  0  1  0  0  5 44]]
--------------------------------------------------
Random Forest Performance:
Accuracy: 0.9953
F1 Score: 0.9953
ROC AUC Score: 0.9999
Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.98      0.99        56
           1       0.97      1.00      0.98        62
           2       1.00      1.00      1.00        78
           3       1.00      1.00      1.00        58
```

```
           4       1.00      1.00      1.00        63
           5       1.00      0.98      0.99        56
           6       1.00      1.00      1.00        50

    accuracy                           1.00       423
   macro avg       1.00      0.99      1.00       423
weighted avg       1.00      1.00      1.00       423

Confusion Matrix:
[[55  1  0  0  0  0  0]
 [ 0 62  0  0  0  0  0]
 [ 0  0 78  0  0  0  0]
 [ 0  0  0 58  0  0  0]
 [ 0  0  0  0 63  0  0]
 [ 0  1  0  0  0 55  0]
 [ 0  0  0  0  0  0 50]]
--------------------------------------------------
Gradient Boosting Performance:
Accuracy: 0.9716
F1 Score: 0.9717
ROC AUC Score: 0.9993
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.95      0.96        56
           1       0.91      0.98      0.95        62
           2       1.00      0.95      0.97        78
           3       0.94      1.00      0.97        58
           4       1.00      1.00      1.00        63
           5       1.00      0.93      0.96        56
           6       0.98      1.00      0.99        50

    accuracy                           0.97       423
   macro avg       0.97      0.97      0.97       423
weighted avg       0.97      0.97      0.97       423

Confusion Matrix:
[[53  3  0  0  0  0  0]
 [ 1 61  0  0  0  0  0]
 [ 0  0 74  4  0  0  0]
 [ 0  0  0 58  0  0  0]
 [ 0  0  0  0 63  0  0]
 [ 0  3  0  0  0 52  1]
 [ 0  0  0  0  0  0 50]]
--------------------------------------------------
SVM Performance:
Accuracy: 0.9220
F1 Score: 0.9225
```

```
ROC AUC Score: 0.9934
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.95      0.95        56
           1       0.76      0.89      0.82        62
           2       0.96      0.96      0.96        78
           3       0.97      0.98      0.97        58
           4       1.00      1.00      1.00        63
           5       0.84      0.77      0.80        56
           6       0.98      0.88      0.93        50

    accuracy                           0.92       423
   macro avg       0.93      0.92      0.92       423
weighted avg       0.93      0.92      0.92       423

Confusion Matrix:
[[53  3  0  0  0  0  0]
 [ 2 55  1  0  0  3  1]
 [ 0  0 75  2  0  1  0]
 [ 0  0  1 57  0  0  0]
 [ 0  0  0  0 63  0  0]
 [ 0 13  0  0  0 43  0]
 [ 0  1  1  0  0  4 44]]
--------------------------------------------------
KNN Performance:
Accuracy: 0.8322
F1 Score: 0.8251
ROC AUC Score: 0.9545
Classification Report:
              precision    recall  f1-score   support

           0       0.72      0.93      0.81        56
           1       0.66      0.47      0.55        62
           2       0.87      0.92      0.89        78
           3       0.95      0.98      0.97        58
           4       0.98      1.00      0.99        63
           5       0.85      0.70      0.76        56
           6       0.74      0.80      0.77        50

    accuracy                           0.83       423
   macro avg       0.82      0.83      0.82       423
weighted avg       0.83      0.83      0.83       423

Confusion Matrix:
[[52  3  0  0  0  1  0]
 [17 29  4  0  0  3  9]
 [ 0  0 72  2  0  1  3]
```

```
 [ 0  0  1 57  0  0  0]
 [ 0  0  0  0 63  0  0]
 [ 3  9  3  0  0 39  2]
 [ 0  3  3  1  1  2 40]]
--------------------------------------------------
Decision Tree Performance:
Accuracy: 0.9645
F1 Score: 0.9646
ROC AUC Score: 0.9800
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.95      0.95        56
           1       0.91      0.97      0.94        62
           2       1.00      0.94      0.97        78
           3       0.92      1.00      0.96        58
           4       1.00      1.00      1.00        63
           5       1.00      0.91      0.95        56
           6       0.96      1.00      0.98        50

    accuracy                           0.96       423
   macro avg       0.96      0.97      0.96       423
weighted avg       0.97      0.96      0.96       423

Confusion Matrix:
[[53  3  0  0  0  0  0]
 [ 2 60  0  0  0  0  0]
 [ 0  0 73  5  0  0  0]
 [ 0  0  0 58  0  0  0]
 [ 0  0  0  0 63  0  0]
 [ 0  3  0  0  0 51  2]
 [ 0  0  0  0  0  0 50]]
--------------------------------------------------
Naive Bayes Performance:
Accuracy: 0.8936
F1 Score: 0.8917
ROC AUC Score: 0.9919
Classification Report:
              precision    recall  f1-score   support

           0       0.84      1.00      0.91        56
           1       0.88      0.73      0.80        62
           2       0.95      0.81      0.88        78
           3       0.84      0.98      0.90        58
           4       1.00      1.00      1.00        63
           5       0.87      0.86      0.86        56
           6       0.87      0.92      0.89        50
```

```
    accuracy                           0.89       423
   macro avg       0.89      0.90      0.89       423
weighted avg       0.90      0.89      0.89       423

Confusion Matrix:
[[56  0  0  0  0  0  0]
 [11 45  0  0  0  6  0]
 [ 0  1 63 11  0  0  3]
 [ 0  0  1 57  0  0  0]
 [ 0  0  0  0 63  0  0]
 [ 0  4  0  0  0 48  4]
 [ 0  1  2  0  0  1 46]]
--------------------------------------------------
XGBoost Performance:
Accuracy: 0.9882
F1 Score: 0.9882
ROC AUC Score: 0.9999
Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.98      0.99        56
           1       0.95      1.00      0.98        62
           2       1.00      0.99      0.99        78
           3       0.98      1.00      0.99        58
           4       1.00      1.00      1.00        63
           5       1.00      0.95      0.97        56
           6       0.98      1.00      0.99        50

    accuracy                           0.99       423
   macro avg       0.99      0.99      0.99       423
weighted avg       0.99      0.99      0.99       423

Confusion Matrix:
[[55  1  0  0  0  0  0]
 [ 0 62  0  0  0  0  0]
 [ 0  0 77  1  0  0  0]
 [ 0  0  0 58  0  0  0]
 [ 0  0  0  0 63  0  0]
 [ 0  2  0  0  0 53  1]
 [ 0  0  0  0  0  0 50]]
--------------------------------------------------
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of
testing was 0.002000 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 2323
[LightGBM] [Info] Number of data points in the train set: 1688, number of used
features: 17
```

```
[LightGBM] [Info] Start training from score -2.056021
[LightGBM] [Info] Start training from score -2.015199
[LightGBM] [Info] Start training from score -1.821828
[LightGBM] [Info] Start training from score -1.954836
[LightGBM] [Info] Start training from score -1.866779
[LightGBM] [Info] Start training from score -1.975979
[LightGBM] [Info] Start training from score -1.950661
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
LightGBM Performance:
Accuracy: 0.9882
F1 Score: 0.9882
ROC AUC Score: 0.9999
Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.96      0.98        56
           1       0.94      1.00      0.97        62
           2       1.00      1.00      1.00        78
           3       1.00      1.00      1.00        58
           4       1.00      1.00      1.00        63
           5       1.00      0.95      0.97        56
           6       0.98      1.00      0.99        50

    accuracy                           0.99       423
   macro avg       0.99      0.99      0.99       423
weighted avg       0.99      0.99      0.99       423

Confusion Matrix:
[[54  2  0  0  0  0  0]
 [ 0 62  0  0  0  0  0]
 [ 0  0 78  0  0  0  0]
 [ 0  0  0 58  0  0  0]
 [ 0  0  0  0 63  0  0]
 [ 0  2  0  0  0 53  1]
 [ 0  0  0  0  0  0 50]]
--------------------------------------------------
```

```
CatBoost Performance:
Accuracy: 0.9882
F1 Score: 0.9883
ROC AUC Score: 0.9999
Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.95      0.97        56
           1       0.93      1.00      0.96        62
           2       1.00      1.00      1.00        78
           3       1.00      1.00      1.00        58
           4       1.00      1.00      1.00        63
           5       1.00      0.96      0.98        56
           6       1.00      1.00      1.00        50

    accuracy                           0.99       423
   macro avg       0.99      0.99      0.99       423
weighted avg       0.99      0.99      0.99       423

Confusion Matrix:
[[53  3  0  0  0  0  0]
 [ 0 62  0  0  0  0  0]
 [ 0  0 78  0  0  0  0]
 [ 0  0  0 58  0  0  0]
 [ 0  0  0  0 63  0  0]
 [ 0  2  0  0  0 54  0]
 [ 0  0  0  0  0  0 50]]
--------------------------------------------------
```

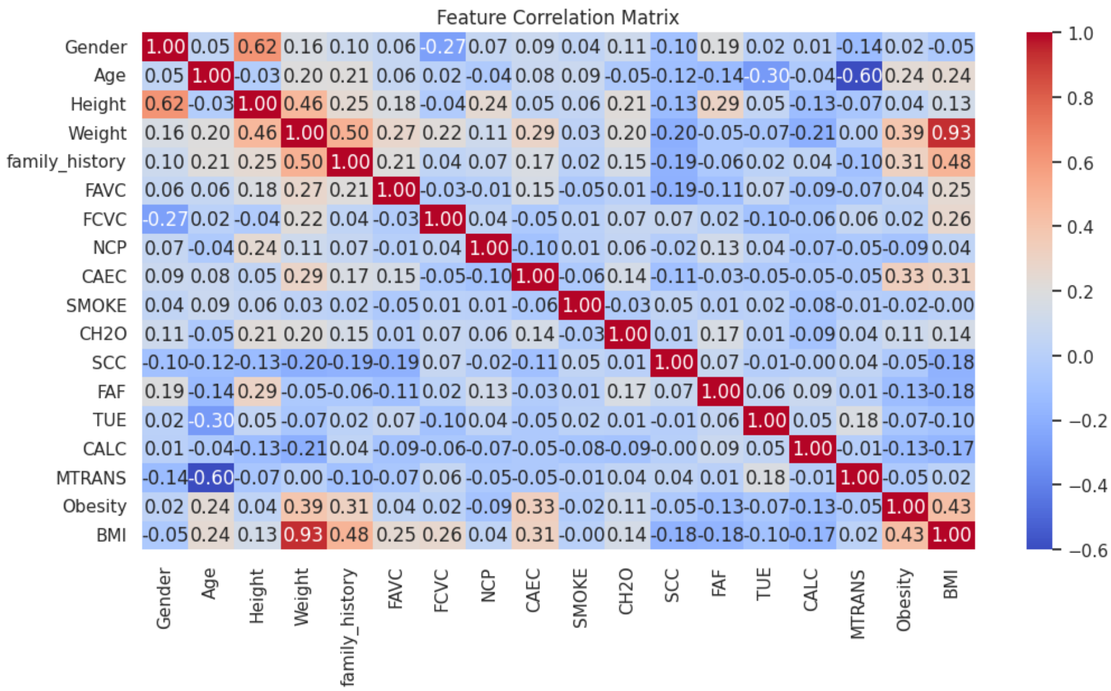
[44]:

```python
# Convert performance metrics to DataFrame
performance_df = pd.DataFrame(performance, columns=['Model', 'Accuracy', 'F1␣
↪Score', 'ROC AUC'])
print(performance_df)
```

```
                 Model  Accuracy  F1 Score   ROC AUC
0  Logistic Regression  0.903073  0.901984  0.992756
1        Random Forest  0.995272  0.995288  0.999890
2    Gradient Boosting  0.971631  0.971736  0.999329
3                  SVM  0.921986  0.922485  0.993406
4                  KNN  0.832151  0.825086  0.954503
5        Decision Tree  0.964539  0.964600  0.979975
6          Naive Bayes  0.893617  0.891705  0.991903
7              XGBoost  0.988180  0.988169  0.999897
8             LightGBM  0.988180  0.988199  0.999886
9             CatBoost  0.988180  0.988268  0.999947
```

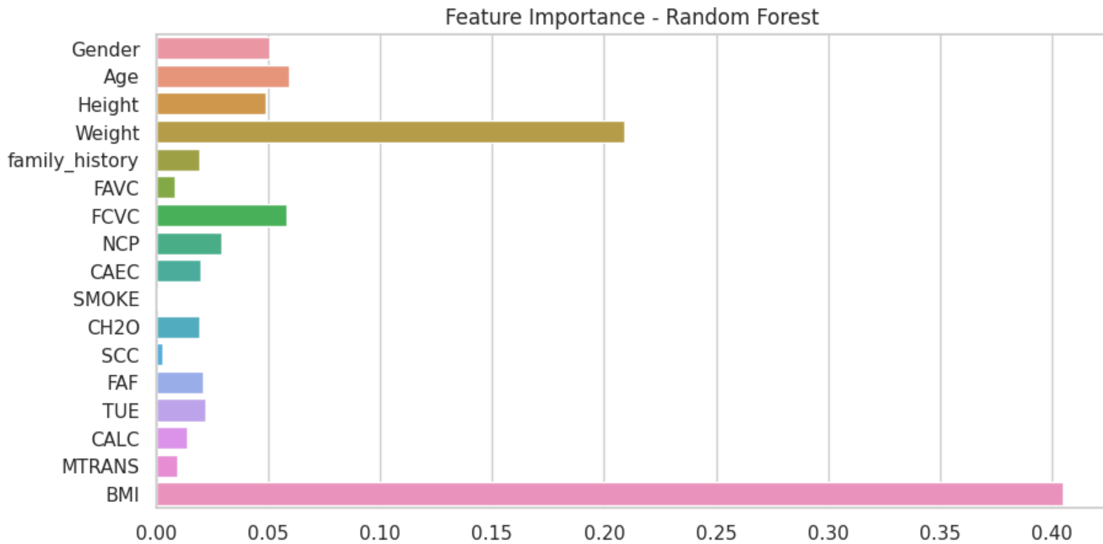
[45]:

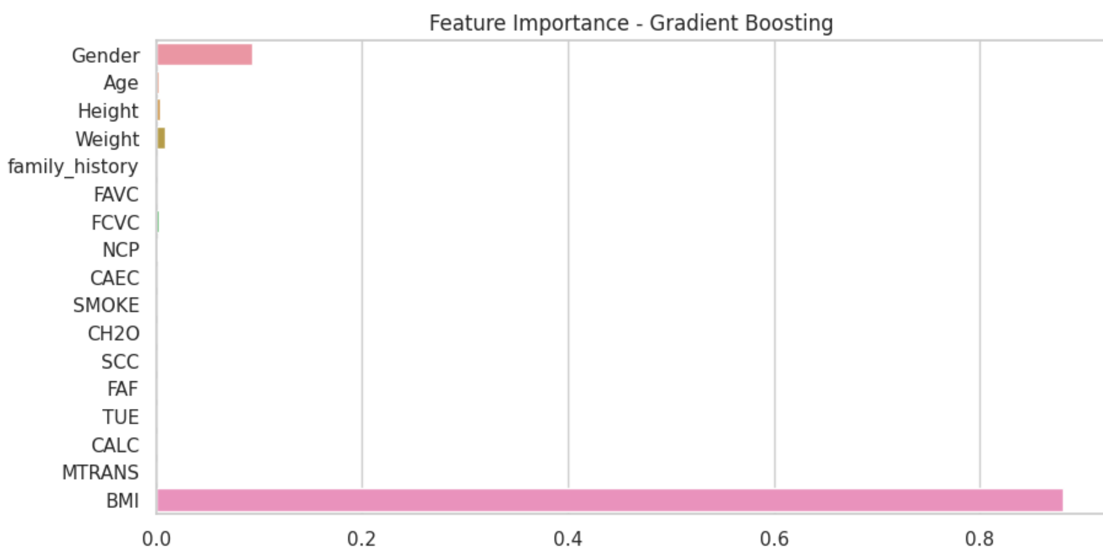
```
# Correlation Matrix
plt.figure(figsize=(12, 6))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt='.2f')
plt.title("Feature Correlation Matrix")
plt.show()
```

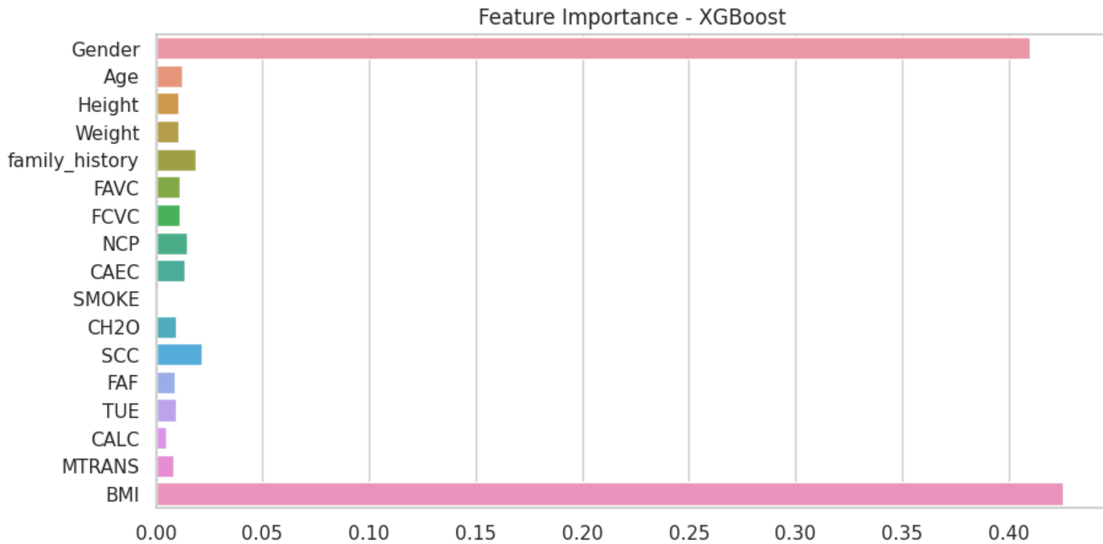


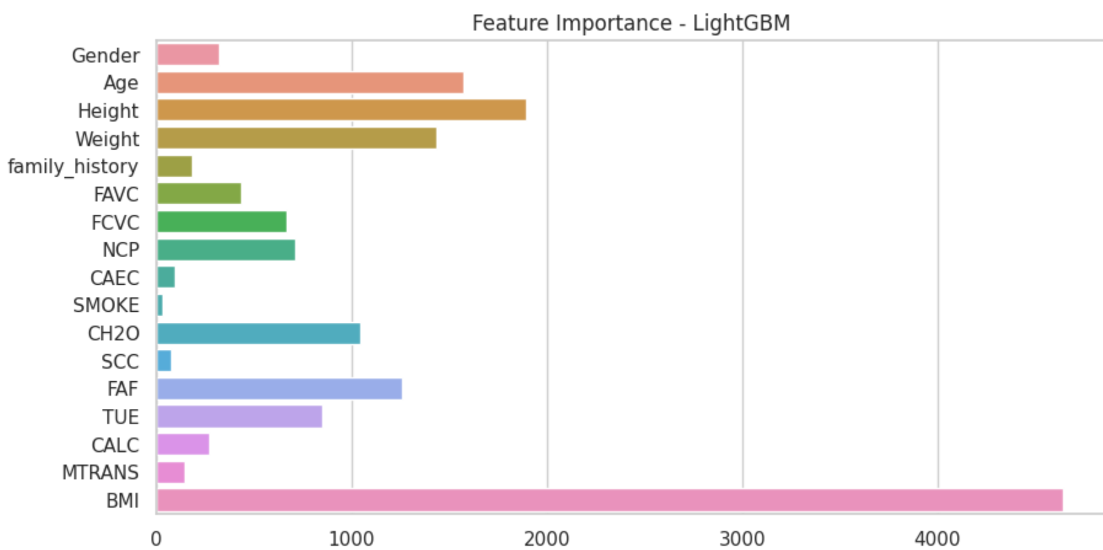
[46]:

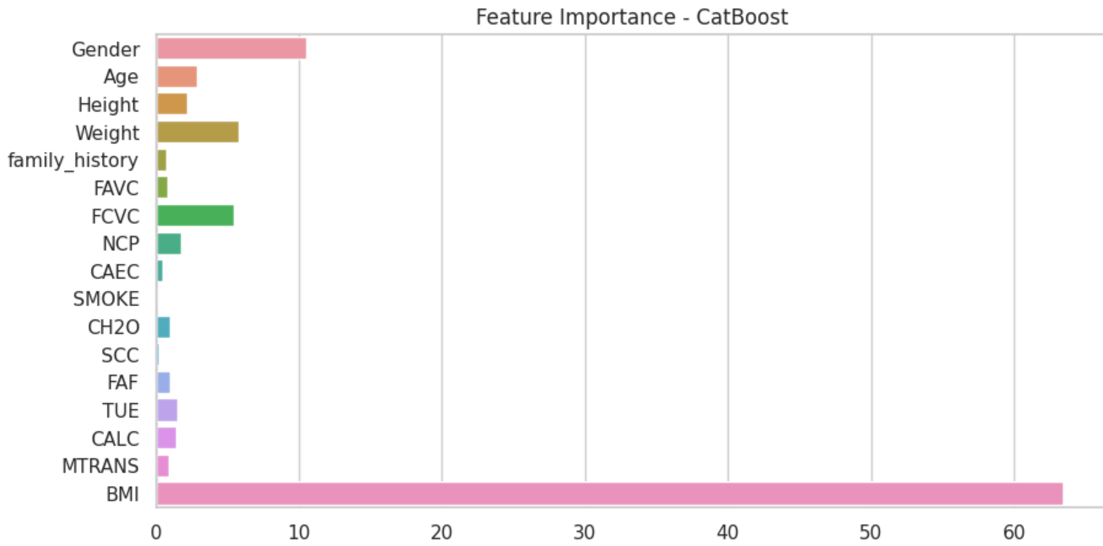
```
# Feature Importance for Tree-based models
tree_models = ["Random Forest", "Gradient Boosting", "XGBoost", "LightGBM",␣
↪"CatBoost"]
for name in tree_models:
    if name in models:
        model = models[name]
        feature_importance = model.feature_importances_
        feature_names = X.columns

        plt.figure(figsize=(10, 5))
        sns.barplot(x=feature_importance, y=feature_names)
        plt.title(f"Feature Importance - {name}")
        plt.show()
```


Feature Importance - Random Forest
Gender
Age
Height
Weight
family_history
FAVC
FCVC
NCP
CAEC
SMOKE
CH2O
SCC
FAF
TUE
CALC
MTRANS
BMI
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40


Feature Importance - Gradient Boosting
Gender
Age
Height
Weight
family_history
FAVC
FCVC
NCP
CAEC
SMOKE
CH2O
SCC
FAF
TUE
CALC
MTRANS
BMI
0.0
0.2
0.4
0.6
0.8


Feature Importance - XGBoost
Gender
Age
Height
Weight
family_history
FAVC
FCVC
NCP
CAEC
SMOKE
CH2O
SCC
FAF
TUE
CALC
MTRANS
BMI
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40


Feature Importance - LightGBM
Gender
Age
Height
Weight
family_history
FAVC
FCVC
NCP
CAEC
SMOKE
CH2O
SCC
FAF
TUE
CALC
MTRANS
BMI
0
1000
2000
3000
4000



0.1 About the Author

Name: Arif Mia

Profession: Machine Learning Engineer & Data Scientist

0.1.1 Career Objective

My goal is to contribute to groundbreaking advancements in artificial intelligence and data science, empowering companies and individuals with data-driven solutions. I strive to simplify complex challenges, craft innovative projects, and pave the way for a smarter and more connected future.

As a **Machine Learning Engineer** and **Data Scientist**, I am passionate about using machine learning, deep learning, computer vision, and advanced analytics to solve real-world problems. My expertise lies in delivering impactful solutions by leveraging cutting-edge technologies.

0.1.2 Skills

- **Artificial Intelligence & Machine Learning**
- **Computer Vision & Predictive Analytics**
- **Deep Learning & Natural Language Processing (NLP)**
- **Python Programming & Automation**

- **Data Visualization & Analysis**

- **End-to-End Model Development & Deployment**

0.1.3 Featured Projects

Lung Cancer Prediction with Deep Learning
Achieved 99% accuracy in a computer vision project using 12,000 medical images across three classes. This project involved data preprocessing, visualization, and model training to detect cancer effectively.

Ghana Crop Disease Detection Challenge
Developed a model using annotated images to identify crop diseases with bounding boxes, addressing real-world agricultural challenges and disease mitigation.

Global Plastic Waste Analysis
Utilized GeoPandas, Matplotlib, and machine learning models like RandomForestClassifier and CatBoostClassifier to analyze trends in plastic waste management.

Twitter Emotion Classification
Performed exploratory data analysis and built a hybrid machine learning model to classify Twitter sentiments, leveraging text data preprocessing and visualization techniques.

0.1.4 Technical Skills

- **Programming Languages:** Python , SQL , R

- **Data Visualization Tools:** Matplotlib , Seaborn , Tableau , Power BI

- **Machine Learning & Deep Learning:** Scikit-learn , TensorFlow , PyTorch

- **Big Data Technologies:** Hadoop , Spark

- **Model Deployment:** Flask , FastAPI , Docker

0.1.5 Connect with Me

Email: arifmiahcse@gmail.com

LinkedIn: www.linkedin.com/in/arif-miah-8751bb217

GitHub: https://github.com/Arif-miad

Kaggle: https://www.kaggle.com/arifmia

Let's turn ideas into reality! If you're looking for innovative solutions or need collaboration on exciting projects, feel free to reach out.

How does this look? Feel free to suggest changes or updates!