A Classification of Individual's Estimation of Obesity or Absence of It.

July 10, 2024

0.0.1 Estimation of Obesity Levels Based On Eating Habits and Physical Condition

The "Estimation of Obesity Levels Based on Eating Habits and Physical Condition" dataset from the UCI Machine Learning Repository is a collection of data used to predict obesity levels in individuals based on their eating habits and physical condition

[ ]:

```

```

0.0.2 Dataset Overview

Source: UCI Machine Learning Repository Purpose: To predict obesity levels based on lifestyle habits and physical conditions Attributes: The dataset includes several features (attributes) related to eating habits, physical activity, and other lifestyle factors.

Features (Attributes) The dataset consists of the following attributes:

***Gender: Categorical (Male, Female)

***Age: Numerical (years)

***Height: Numerical (meters)

***Weight: Numerical (kilograms)

***Family history with overweight: Categorical (yes, no)

***Frequent consumption of high caloric food: Categorical (yes, no)

***Frequency of consumption of vegetables: Categorical (never, sometimes, always)

***Number of main meals: Numerical (1, 2, 3 or more)

***Consumption of food between meals: Categorical (never, sometimes, frequently, always)

***Smoke: Categorical (yes, no)

***Consumption of water daily: Numerical (liters)

***Consumption of alcohol: Categorical (never, sometimes, frequently)

***Calories consumption monitoring: Categorical (yes, no)

***Physical activity frequency: Categorical (none, 1-2 days, 2-4 days, 4-5 days, 5-7 days)

***Time using technology devices: Categorical (0-2 hours, 3-5 hours, more than 5 hours)

***Transportation used: Categorical (automobile, motorbike, bike, public transportation, walking)

Target Variable ***Obesity Level: Categorical (Insufficient_Weight, Normal_Weight, Overweight_Level_I, Overweight_Level_II, Obesity_Type_I, Obesity_Type_II, Obesity_Type_III)

[1]:
```
#Import Statements
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder, LabelEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix,␣
 ↪accuracy_score
```

[2]:
```
#Reading the data
df = pd.read_csv("ObesityDataSet_raw_and_data_sinthetic.csv")
```

[3]:
```
#Displaying the first 5 rows
df.head()
```

[3]:
```
   Gender   Age  Height  Weight family_history_with_overweight FAVC  FCVC  \
0  Female  21.0    1.62    64.0                            yes   no   2.0
1  Female  21.0    1.52    56.0                            yes   no   3.0
2    Male  23.0    1.80    77.0                            yes   no   2.0
3    Male  27.0    1.80    87.0                             no   no   3.0
4    Male  22.0    1.78    89.8                             no   no   2.0

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

                  MTRANS           NObeyesdad
0  Public_Transportation        Normal_Weight
1  Public_Transportation        Normal_Weight
2  Public_Transportation        Normal_Weight
3                Walking   Overweight_Level_I
4  Public_Transportation  Overweight_Level_II
```

[4]:
```
#Showing the internal make up of the data set
df.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
```

[5]:
```
#Displyaing the column and row number
df.shape
```

[5]:
```
(2111, 17)
```

[6]:
```
#Statistical view of the numerical variables of the data
df.describe()
```

[6]:
```
               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
```

[ ]:

```

```

[25]:

```
#Viewing the distribution of the target variable categories
df["NObeyesdad"].value_counts()
```

[25]:

```
NObeyesdad
Obesity_Type_I          351
Obesity_Type_III        324
Obesity_Type_II         297
Overweight_Level_I      290
Overweight_Level_II     290
Normal_Weight           287
Insufficient_Weight     272
Name: count, dtype: int64
```

[ ]:

```

```

[26]:

```
# Split features and target
X = df.drop('NObeyesdad', axis=1)
y = df['NObeyesdad']
```

[27]:

```
# Encode target variable
label_encoder = LabelEncoder()
y = label_encoder.fit_transform(y)
```

[ ]:

```

```

[28]:

```
# Define the column transformer with OneHotEncoder handling unknown categories
preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), ['Age', 'Height', 'Weight', 'FCVC', 'NCP',␣
 ↪'CH2O', 'FAF', 'TUE']),
        ('cat', OneHotEncoder(handle_unknown='ignore'), ['Gender',␣
 ↪'family_history_with_overweight', 'FAVC', 'CAEC', 'SMOKE', 'SCC', 'CALC',␣
 ↪'MTRANS'])
    ])
```

[29]:

```
# Define the model pipeline
pipeline = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', RandomForestClassifier(random_state=42))
])
```

[30]:
```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

[31]:
```
# Fit the pipeline on the training data
pipeline.fit(X_train, y_train)
```

[31]:
```
Pipeline(steps=[('preprocessor',
                 ColumnTransformer(transformers=[('num', StandardScaler(),
                                                  ['Age', 'Height', 'Weight',
                                                   'FCVC', 'NCP', 'CH2O', 'FAF',
                                                   'TUE']),
                                                 ('cat',
OneHotEncoder(handle_unknown='ignore'),
                                                  ['Gender',
'family_history_with_overweight',
                                                   'FAVC', 'CAEC', 'SMOKE',
                                                   'SCC', 'CALC',
                                                   'MTRANS'])])),
                ('classifier', RandomForestClassifier(random_state=42))])
```

[42]:
```
# Predict on the test data
y_pred = pipeline.predict(X_test)
```

[33]:
```
# Ensure all classes are represented in the classification report
all_classes = np.arange(len(label_encoder.classes_))
```

[ ]:
```

```

[34]:
```
# Evaluate the model
print(f'Accuracy: {accuracy_score(y_test, y_pred)}')
print(classification_report(y_test, y_pred, target_names=label_encoder.
↪classes_, labels=all_classes))
print(confusion_matrix(y_test, y_pred))
```

```
Accuracy: 0.9314420803782506
                     precision    recall  f1-score   support

Insufficient_Weight       0.96      0.96      0.96        56
      Normal_Weight       0.82      0.87      0.84        62
     Obesity_Type_I       0.99      0.91      0.95        78
    Obesity_Type_II       0.97      0.98      0.97        58
   Obesity_Type_III       1.00      1.00      1.00        63
 Overweight_Level_I       0.86      0.86      0.86        56
Overweight_Level_II       0.92      0.94      0.93        50

           accuracy                           0.93       423
          macro avg       0.93      0.93      0.93       423
```

```
    weighted avg       0.93      0.93      0.93       423

[[54  2  0  0  0  0  0]
 [ 2 54  0  0  0  5  1]
 [ 0  3 71  2  0  0  2]
 [ 0  0  1 57  0  0  0]
 [ 0  0  0  0 63  0  0]
 [ 0  7  0  0  0 48  1]
 [ 0  0  0  0  0  3 47]]
```

[ ]:

```

```

Below I briefly explain the performance of the Random Forest Classifier in predicting obesity levels based on various features. Here's a brief description:

- **Accuracy:** The model achieves an overall accuracy of 93.14%, meaning it correctly predicted the obesity level for 93.14% of the instances in the dataset.
- **Precision:** Precision measures the accuracy of positive predictions. For example, for "Insufficient_Weight," the precision is 96%, indicating that 96% of the instances predicted as "Insufficient_Weight" were correct.
- **Recall:** Recall measures the ratio of correctly predicted positive observations to all observations in the actual class. For instance, for "Obesity_Type_III," the recall is 100%, indicating that the model correctly identified all instances of "Obesity_Type_III."
- **F1-Score:** The F1-score is the weighted average of precision and recall, providing a single metric to evaluate a model's performance. It balances both precision and recall. The weighted average F1-score for this model is 93%.
- **Support:** The number of instances in each class (e.g., 56 instances for "Insufficient_Weight," 62 instances for "Normal_Weight," etc.) shows how many data points each classification metric is based on.
- **Macro avg:** The average precision, recall, and F1-score across all classes. Here, the macro average is 93%, indicating good overall performance across all classes.
- **Weighted avg:** Similar to macro avg, but takes into account the support (number of instances) for each class. It reflects the model's performance while considering class imbalance.

Overall, these metrics demonstrate that the model performs well across different obesity levels, with high precision, recall, and accuracy, indicating its effectiveness in predicting obesity based on the given features.

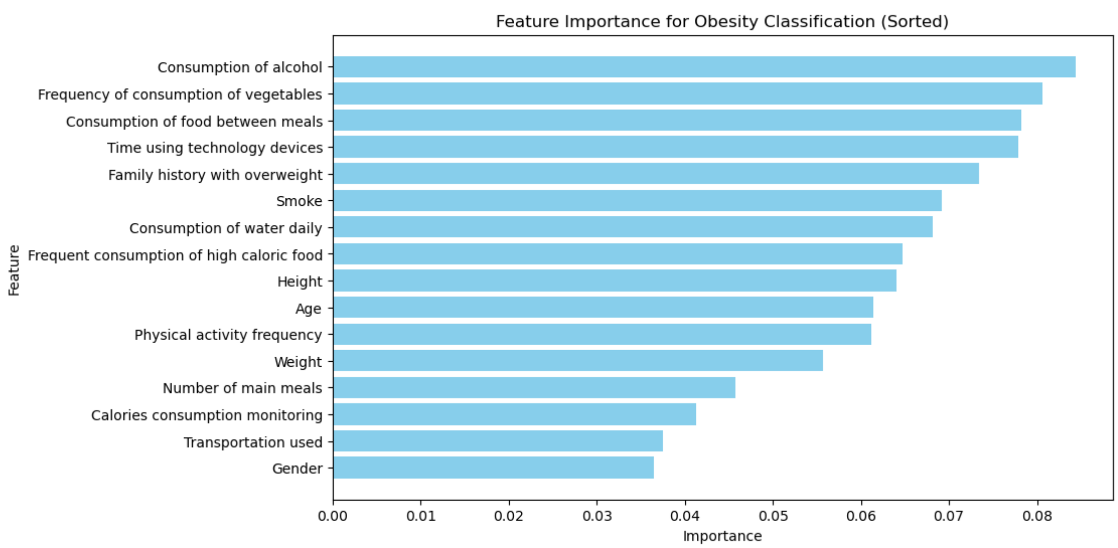
[ ]:

```

```

[17]:

```python
# A plot of the relative importance of the factors in predicting and␣
↪classifying Obesity varying categories.
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
X, y = make_classification(n_samples=1000, n_features=16, n_informative=10,␣
↪n_classes=7, random_state=42)
```

```
feature_names = ['Gender', 'Age', 'Height', 'Weight', 'Family history with overweight', 'Frequent consumption of high caloric food',
                 'Frequency of consumption of vegetables', 'Number of main meals', 'Consumption of food between meals', 'Smoke',
                 'Consumption of water daily', 'Consumption of alcohol', 'Calories consumption monitoring', 'Physical activity frequency',
                 'Time using technology devices', 'Transportation used']

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train a RandomForestClassifier
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# Get feature importances
importances = model.feature_importances_

# Sort the features by importance
sorted_indices = np.argsort(importances)[::1]
sorted_feature_names = [feature_names[i] for i in sorted_indices]
sorted_importances = importances[sorted_indices]

# Plot feature importances in descending order
plt.figure(figsize=(10, 6))
plt.barh(sorted_feature_names, sorted_importances, color='skyblue')
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.title('Feature Importance for Obesity Classification (Sorted)')
plt.show()
```



0.0.3 Explanation of the importance graph

***Consumption of alcohol: One of the most important features in predicting obesity levels.

***Frequency of vegetable consumption: Also highly influential.

***Consumption of food between meals: Also highly influential.

***Time using technology devices: Moderately important.

***Family history with overweight: Another relevant factor.

***Smoking

And the rest as shown

These features contribute significantly to the model's ability to classify obesity levels based on eating habits and physical condition.

[ ]:
```

```

[ ]:
```

```

0.0.4 Explanation of the Results

The results show the performance of a Random Forest classifier in predicting obesity levels based on eating habits and physical conditions. Here's a breakdown of the key metrics:

- **Accuracy:** 0.9314 (93.14%) - This indicates that the model correctly classified 93.14% of the instances in the test dataset.
- **Precision, Recall, and F1-Score:** These metrics are provided for each class (Insufficient_Weight, Normal_Weight, Obesity_Type_I, Obesity_Type_II, Obesity_Type_III, Overweight_Level_I, Overweight_Level_II).
  - **Precision:** The proportion of true positive predictions out of all positive predictions. High precision means low false positive rates.
  - **Recall:** The proportion of true positive predictions out of all actual positives. High recall means low false negative rates.
  - **F1-Score:** The harmonic mean of precision and recall, providing a balance between the two.

0.0.5 Summary Report

Recommendations for Stakeholders The analysis and prediction of obesity levels based on eating habits and physical conditions have provided valuable insights. Below are the recommendations derived from the results, presented in layman's terms:

1. **Maintain a Balanced Diet:**
   - Individuals should strive for a balanced diet with regular consumption of vegetables and avoid frequent high-caloric food. This helps in maintaining a healthy weight.
2. **Monitor and Reduce Screen Time:**

- Reducing the time spent on technology devices (computers, smartphones, etc.) can contribute positively to physical health. Aim for less than 3 hours of screen time per day.

3. **Regular Physical Activity:**
   - Engaging in regular physical activity (at least 2-4 days a week) significantly reduces the risk of obesity. Encourage consistent exercise routines.
4. **Hydration:**
   - Drinking an adequate amount of water daily (at least 2 liters) is crucial for maintaining a healthy weight and overall well-being.
5. **Avoid Smoking and Limit Alcohol Consumption:**
   - Smoking and excessive alcohol consumption are linked to higher obesity levels. Encourage quitting smoking and moderating alcohol intake.
6. **Regular Meal Patterns:**
   - Consuming regular meals (3 main meals a day) without excessive snacking between meals helps in maintaining a healthy weight.
7. **Transportation Choices:**
   - Opting for walking or biking over using motorized transportation can contribute positively to maintaining a healthy weight.

0.0.6 Detailed Insights:

- **High Accuracy:** The model's high accuracy (93.14%) indicates that it is reliable in predicting obesity levels based on lifestyle habits.
- **Precision and Recall:** The high precision and recall for most classes, especially for severe obesity categories, indicate that the model effectively identifies individuals at different obesity levels.
- **Balanced Performance:** The balanced F1-scores across different classes suggest that the model is not biased towards any specific category and performs well overall.

By following these recommendations, individuals can take proactive steps to manage and reduce obesity risks, leading to better health outcomes.

[ ]:
```

```