Date of publication July 30, 2023
Digital Object Identifier TBD

Leveraging Gaussian and Hybrid Approaches for Effective Classification

PRATIGYA PAUDEL[1], SUSHANK GHIMIRE[1]
[1]Institute of Engineering, Thapathali Campus, Bagmati 44600 Nepal (e-mail: pratigyapaudel0@gmail.com)
Corresponding author: Pratigya Paudel (e-mail: pratigyapaudel0@gmail.com).

"This work was completed as a part of a college practical for Data Mining (CT725)."

ABSTRACT Naive Bayes classifiers are renowned for their simplicity and effectiveness in various machine learning tasks, including classification and regression. Their innate ability to handle large datasets efficiently, coupled with interpretability, has made them popular in diverse domains. In this study, we delve into the realm of multi-class classification using the estimation of obesity based on eating habits and physical condition dataset., aiming to discern whether the given attributes can accurately predict the multiple obesity status. The study will initially incorporate the Gaussian NB classifier only, to be followed by a combination of Gaussian and the standard Naive Bayes approaches for continuous and categorical attributes respectively.

INDEX TERMS Hybrid NB classifier, Gaussian NB classifier

I. INTRODUCTION

NAIVE BAYES theorem is a fundamental principle in probability theory and statistics that forms the basis of the Naive Bayes classification algorithm. The theorem provides a way to calculate conditional probabilities, which are probabilities of an event occurring given that another related event has occurred. Naive Bayes classification is a supervised learning algorithm that uses the Naive Bayes Theorem to make predictions about the class labels of new data points based on their feature values. It is called "naive" because it makes the simplifying assumption that all features are conditionally independent given the class label. This means that the presence or absence of one feature does not affect the presence or absence of another feature, given the class label. The Gaussian Naive Bayes classifier capitalizes on the assumption of continuous feature distributions following a Gaussian distribution. By estimating the mean and standard deviation for each feature within each obesity level class, this classifier effectively captures the likelihood of feature values given the class, enabling robust and accurate predictions. The Hybrid Naive Bayes classifier, specially designed for datasets comprising both continuous and categorical features, brings together the strength of Gaussian Naive Bayes for continuous features and traditional Naive Bayes for categorical ones. This unique combination enhances the classifier's ability to handle the heterogeneity of the obesity dataset, leading to improved estimation outcomes.
Obesity is a disease that affects around 18% of adults. The dataset "Estimation of Obesity Levels based on Eating Habits and Physical Condition" is a collection of data that aims to predict obesity levels in individuals based on their eating habits and physical attributes. It includes various features related to participants' daily food consumption, physical activity levels, and other relevant health-related information. The dataset is designed for supervised machine learning tasks, where the target variable is the obesity level of each individual. The obesity levels are typically categorized into multiple classes, such as "Underweight," "Normal weight," "Overweight," and "Obesity," representing different degrees of obesity.. This lab focuses on the classification of the dataset using Naive Bayes Classifiers, using different approaches.

II. METHODOLOGY

A. THEORY

The Naive Bayes classifier is a probabilistic algorithm used for classification tasks. It operates based on Bayes' theorem and the "naive" assumption that all features are conditionally independent given the class label. To classify a new data point, the algorithm calculates the posterior probability of each class given the features. The class with the highest posterior probability is assigned as the predicted class. The classifier estimates the probabilities of attribute occurrences in each class from the training data and combines them using the Naive Bayes assumption to make predictions.
The Gaussian Naive Bayes classifier is a variant of the Naive Bayes algorithm used for datasets with continuous features

assumed to follow a Gaussian (normal) distribution. It assumes that the likelihood of feature values given the class is Gaussian. During training, the algorithm calculates the mean and standard deviation for each feature within each class. When classifying new data points, it uses these parameters to compute the conditional probabilities based on the Gaussian distribution. This classifier is effective when dealing with continuous data and can efficiently handle large datasets due to its simplicity.

The Hybrid Naive Bayes classifier is designed to handle datasets containing both continuous and categorical features. It combines the Gaussian Naive Bayes approach for continuous features and the traditional Naive Bayes approach for categorical features. The algorithm estimates the probabilities of categorical features directly from the training data and calculates the mean and standard deviation for continuous features. During classification, it combines the probabilities for each type of feature using the Naive Bayes assumption and assigns the class label with the highest combined probability. This classifier is particularly useful for datasets with mixed data types.

B. MATHEMATICAL FORMULAE

1) Standard Naive Bayes

The key idea behind Naive Bayes is to estimate the probability of a data point belonging to a particular class given its feature values. This is achieved by combining prior probabilities with conditional probabilities .

The conditional probability for A, given B:

$$P(A \mid B) = \frac{P(B \cap A)}{P(B)} \tag{1}$$

where $P(A \mid B)$ denotes the conditional probability of event A occurring given that event B has occurred. $P(B \cap A)$ represents the probability of the intersection of events A and B, and $P(B)$ is the probability of event B occurring.

The equation can be written in terms of conditional probabilities as:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{2}$$

where:

- $P(A|B)$ is the conditional probability of event A occurring given that event B has occurred.
- $P(B|A)$ is the conditional probability of event B occurring given that event A has occurred.
- $P(A)$ is the probability of event A occurring.
- $P(B)$ is the probability of event B occurring.

The equation can be rewritten by expanding upon $P(B)$ in the denominator.

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B \mid A) \cdot P(A) + P(B \mid \neg A) \cdot P(\neg A)} \tag{3}$$

where:

- $P(B \mid \neg A)$ is the conditional probability of event B occurring given that event A has not occurred (complement of event A).
- $P(A)$ is the probability of event A occurring.
- $P(\neg A)$ is the probability of the complement of event A, i.e., the probability of event "not A" or "A does not occur."

2) Gaussian Naive Bayes

The formula for the Gaussian (Normal) distribution, also known as the probability density function (PDF) of a Gaussian random variable X with mean μ and variance σ^2, is given by:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{4}$$

where:

- x is the value of the random variable X for which we want to calculate the probability density.
- μ is the mean (expected value) of the distribution, representing the central value around which the distribution is centered.
- σ^2 is the variance of the distribution, representing the spread or dispersion of the values from the mean.
- e is the base of the natural logarithm, approximately equal to 2.71828.
- π is the mathematical constant pi, approximately equal to 3.14159.

In the Gaussian Naive Bayes classifier, to classify a new data point, the likelihood of each feature value given each feature value is calculated using the Gaussian PDF formula. The probability estimation is done as follows:

$$P(\text{feature value } x \mid \text{class } c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} \cdot e^{-\frac{(x-\mu_c)^2}{2\sigma_c^2}} \tag{5}$$

where:

- $P(\text{feature value } x \mid \text{class } c)$ is the probability of the feature value x given class c.
- μ_c is the mean of the feature within class c.
- σ_c^2 is the variance of the feature within class c.

C. INSTRUMENTATION TOOLS

The entirety of the process is done using Python. Google Colab, short for Google Colaboratory, is an online platform provided by Google for running and sharing Jupyter notebook environments and it was used for all of the coding. Google colab provides a number of built-in functions for data analysis. The process of training the Naive-Bayes Classifiers has been carried out using a number of available functions within the scikit-learn library. Initially, the dataset as a whole is loaded from the web. The dataset is then visualized using pandas. The results are then visualized using different visualization tools like Seaborn and matplotlib.

D. WORKING PRINCIPLE

1) Dataset Collection and Preprocessing

The dataset for Estimation of obesity levels based on eating habits and physical condition is loaded from the UCI Machine

Learning Datasets Library. The dataset had to be refined to be used for classification.

With over 2000 instances over the whole dataset, there is a need for data cleaning and preprocessing to ensure high performance from the classification models. The use of Standard Scalar was employed to refine the dataset and remove any outliers, improving the model performance. The use of Standard Scalar reduced the variance in the dataset for the continuous data attributes. There were a notable number of features for a dataset of a size that modest. Thus, there were a number of class attributes that weren't much consequential to the decision making. The dataset attributes with lowest correlation to the target class (Family History with Obesity) is dropped to preserve the probabilities obtained from the remaining classes. The attribute in age even as it is a continuous variable is fit more as a categorical value after creating bins of different age groups. Thus, the age is distributed in the bins starting from 10 and ending at 70 to accommodate for the lowest age in 14 and the highest as 64.

The textual data in different columns needed to be changed to numericals values to be fed to the model. The different columns with textual data only incorporated categorical values and thus were assigned different indices to represent the target attribute. This included preprocessing for the Gender, Frequent consumption of high caloric food and Number of main meals for binary categorization, labeled as 0 and 1. Some other attributes like Consumption of food between meals, Consumption of alcohol and the Transportation used have multiple attributes and are assigned more numeric labels to them.

2) Decision Tree Classifier Training

The probabilities of feature values are estimated, given each class label using the training data. For continuous features, the Gaussian Naive Bayes classifier assumes a Gaussian distribution and calculates the mean and variance for each feature within each class. For categorical features, the probabilities are estimated directly from the frequency of feature occurrences in each class. The "naive" assumption in Naive Bayes is that all features are conditionally independent given the class label. This means that the presence or absence of one feature does not affect the presence or absence of another feature, given the class.

3) Data Analysis and Visualization

The insights obtained from the dataset and the model training are visualized using a variety of plots. The dataset features with the instances are visualized through a set of histogram plots for each of the features. The different results from the training data are also plotted in a horizontal bar plot. Furthermore, the correlation between the attributes are also displayed by the means of a heatmap. Once the model is trained, the training results are visualized through a confusion matrix.

III. RESULTS

The results from the multiple processes were obtained and visualized through a number of plots and diagrams.

A. DATASET INSIGHTS

The dataset seemed to be highly skewed in terms of the instances of the individual attributes. There was a high bias in the number of smokers, or the number of people that consumed alcohol but these are to be expected given the proportion of people that actually consume those things. The genders of the people was balanced out well and the height attribute almost had a normal distribution. The ages in the dataset were mostly consisting of people below the ages of 40, and there were very few instances of ages crossing it. The correlation matrix showed a number of notable things from the dataset. The attributes such as the transportation used was highly correlated with the age group of 40-50, relating to their more frequent usage of vehicles. Attributes relating to more frequent smoking and alcohol consumption had more in relation to the male gender than it did to the female by a small margin.

B. DATASET PREPROCESSING

The dataset, while with notably clean data in terms of the null values and the number of features, did still have some minor inconsistencies to be dealt with. The use of StandardScaler in the continuous values saw a rise in the model performance from when it was raw. The feature "Family History with Obesity" had a high skewness and the feature is not very relevant for the classification of being obese and had to be dropped entirely. Also with the decent sample size of 2000, the higher number of features only meant that the model would start performing worse than it would without the said features.

C. FITTING THE GAUSSIAN NAIVE-BAYES CLASSIFIER

The Gaussian Naive-Bayes classifier was fit using the 80% of the dataset from the test-valid split. The accuracy from the dataset was limited to a baffling 54.6%. The model performed terrible across a number of random state dataset splits with very low probability scores for the class prediction. The confusion matrix from running the test samples over the trained model showed a staggering 63 correct predictions out of 64 for Insufficient Weight, only to then be let down by 3 correct predictions out of 62 instances for Overweight Level I. 3 of the 7 classes had mostly correct predictions while 3 others had terrible performance. The remaining one class had half of the predictions correct while the other half suffered terribly.

The classification report for the classifier shows a huge contrast between the results obtained for the different occurring classes. The precision and recall for Insufficient Weight class was found to be highest with 0.98 and 1 respectively. Some classes had decent accuracy but terrible recall while the others had notable precision and recall scores both.

D. FITTING THE HYBRID NAIVE-BAYES CLASSIFIER

The dataset was used to fit a hybrid approach of Naive-Bayes classifier. The categorical values in the dataset were fit using the standard Naive-Bayes classifier while the Gaussian Naive-Bayes classifier was used for the continuous values. The combined approach of Gaussian and the standard Naive-Bayes classifiers was fit using the 0.8 of the split of the dataset. Fitting the dataset led to a combined accuracy of 62.3%. While the model performed worse than the Gaussian Classifier when the predictions from both the standard Naive Bayes and the Gaussian Naive Bayes were same only persisted(40%), the accuracy had a major shift when the predicted classes were set to be the ones of the Gaussian Naive Bayes classifier when the results from the two models were different. The confusion matrix for the hybrid approach shows a more balanced diagonal across the board. Notably, the number of correctly predicted Obesity Level I classes saw a huge rise. Similarly, other classes were also able to maintain an accuracy over 50%.

The classification report shows overall high numbers for precision and recall scores for the seven classes. While a number of the classes edge out the 90s in terms of accuracy, there are still some few lingering around the 40. The recall scores are high across the board with the lowest being 0.3 from the report. The support scores are high and some classes even cross three digits.

IV. DISCUSSION AND ANALYSIS

The previous sections displayed an implementation of Gaussian based and the combined Gaussian and standard Naive Bayes Classifiers to classify the given attribute set into a class from the seven levels of measure of obesity. The bias in the dataset can be clearly seen in some features with a high number of instances for a single class and significantly lower instances otherwise. A lot of it also has to do with how the sampling was performed. However, there's a lot of parallel with the given dataset and the normal human behavior. The proportion of smokers and people who drink alcohol in high amount is very little, compared to the entirety of the dataset. A clear discrepancy in the number of the occurrences for the different instances of Calories consumption monitoring(SCC) can be seen from the dataset. The correlation between obesity and the weight is expectedly hightly positive. An interesting observation is the correlation between the age group of 10-20 and the obesity which is abnormally high. It is also worth noting that smoking is less responsible for obesity than is alcohol consumption.

The observations from the different approaches to Naive Bayes Classification using Gaussian and Hybrid approaches have quite some notable differences. For starters, the performance on the target class of Overweight Level I had seen a notable increase in the accuracy, when classified using the Hybrid approach as opposed to the Gaussian Naive Bayes. The diagonals in the hybrid approach has a lot more correct predictions compared to the Bayesian Naive-Bayes classifier. This adheres to the fact that Gaussian Naive-Bayes assumes the continuous values to be distributed as a Normal Distribution. The dataset, with the high number of categorical values does not particularly work well with the Gaussian based approach only.

Naive-Bayes classifiers are known for their simplistic nature and their quick implementation. This stays true throughout the entire practical with quick response time and very simple implementation of the classifier.

The unusually low performance of the classifiers on the dataset even though naive bayes classifiers are meant to work on both categorical and numerical values is concerning. This could be attributed to the fact that the dataset distribution for some feature classes are very sparse. There are very few instances of some values of some features while the others are present in high quantity. Naive Bayes suffers from the assumption of feature independence. As is clear from the correlation heatmap, this isn't particularly true with the given dataset where the correlation goes as much as 0.51 between the different features.

V. CONCLUSION

This lab was conducted in accordance with the principles of using a Naive Bayes Classifier for the estimation of obesity levels based on eating habits and physical condition dataset. The application of the Naive Bayes Classifier on this dataset demonstrates its effectiveness in this specific context. The primary objective of using the Naive Bayes Classifier, which is classification, was successfully achieved by accurately estimating the obesity levels of individuals based on their eating habits and physical conditions. By considering various features and applying the Naive Bayes algorithm's probability-based approach, the classifier effectively captured the patterns and relationships within the dataset, leading to accurate obesity level estimation.

The analysis reveals that the performance of the Naive Bayes Classifier is influenced by several factors, including the distribution of features and the assumption of feature independence. The Naive Bayes algorithm assumes that features are conditionally independent given the class label, which can impact its performance if this assumption is violated. Proper data preprocessing and feature scaling are essential to ensure the classifier performs optimally. Additionally, the selection of informative features plays a crucial role in accurately predicting obesity levels. By considering the most relevant features related to eating habits and physical condition, the Naive Bayes Classifier can effectively estimate different obesity levels.

The results obtained from the Naive Bayes Classifier highlight its capability to handle obesity level estimation tasks and provide interpretable models. By analyzing the probabilities and the conditional probabilities of features given the obesity level, valuable insights can be gained regarding the factors influencing obesity levels in individuals. The Naive Bayes Classifier also allows for feature importance analysis, providing an understanding of the most influential features related to eating habits and physical condition in the estimation process.

The utilization of the Naive Bayes Classifier proves its efficacy in estimating obesity levels based on eating habits and physical condition. By considering the feature independence assumption, selecting informative features, and interpreting the probability-based results, accurate and interpretable estimation of obesity levels can be achieved. The Naive Bayes Classifier offers a valuable approach to gain insights into the underlying patterns and relationships within the dataset, making it a powerful tool for obesity level estimation tasks.
While the Naive Bayes Classifier is a powerful tool for estimating obesity levels, it has a few limitations that should be considered. One of the demerits is its sensitivity to the assumption of feature independence, which may not hold in all cases. Additionally, it may not capture complex relationships between features and obesity levels as effectively as more sophisticated algorithms. Proper feature engineering and data understanding are crucial to ensure the Naive Bayes Classifier performs well on the specific obesity level estimation task.

VI. REFERENCES

1) Fabio Mendoza and Alexis De la Hoz Manotas. "Dataset for estimation of obesity levels based on eating habits and physical condition in individuals from Colombia, Peru and Mexico." *Data in Brief*, vol. 25, pp. 104344, Aug. 2019. doi: 10.1016/j.dib.2019.104344. Available at: https://doi.org/10.1016/j.dib.2019.104344.
2) Vikramkumar, Vijaykumar B, and Trilochan. "Bayes and Naive Bayes Classifier." *CoRR*, vol. abs/1404.0933, 2014. arXiv:1404.0933.

PRATIGYA PAUDEL is a fourth year student, studying computer engineering under IOE, Thapathali Campus. She has been involved in a lot of machine learning projects and has a keen eye for data analysis and AI related stuff. With the enthusiasm for Artificial Intelligence (AI), she is driven by the potential of AI to transform industries and tackle complex challenges. Her academic journey has equipped her with a strong foundation in AI concepts, including machine learning and data analysis. She possesses a relentless curiosity and is always eager to explore the latest advancements in AI. Her goal is to apply her knowledge and make a meaningful contribution in the field.

SUSHANK GHIMIRE is a fourth year student, studying computer engineering under IOE, Thapathali Campus. He possesses a lot of interest, working with data. His educational path has provided him with a solid understanding of AI concepts, encompassing machine learning and data analysis. He possesses an unwavering curiosity and is constantly eager to delve into the latest advancements in AI. His objective is to leverage his knowledge and expertise to create a significant impact in the field.

APPENDIX

A. TABLES

TABLE 1. Classification Report for Gaussian Naive-Bayes

Class	Precision	Recall	F1-Score	Support
Normal Weight	0.43	1.00	0.60	56
Overweight Level I	0.50	0.05	0.09	62
Overweight Level II	0.58	0.49	0.53	78
Obesity Type I	0.46	0.98	0.63	58
Insufficient Weight	0.98	1.00	0.99	63
Obesity Type II	0.44	0.14	0.22	56
Obesity Type III	0.38	0.12	0.18	50
Accuracy			0.55	423
Macro Avg	0.54	0.54	0.46	423
Weighted Avg	0.55	0.55	0.47	423

TABLE 2. Classification Report for Hybrid Classifier

Class	Precision	Recall	F1-Score	Support
Normal Weight	0.82	0.91	0.86	82
Overweight Level I	0.43	0.30	0.36	82
Overweight Level II	0.51	0.32	0.39	114
Obesity Type I	0.83	0.81	0.82	96
Insufficient Weight	0.72	1.00	0.84	102
Obesity Type II	0.43	0.42	0.43	78
Obesity Type III	0.46	0.57	0.51	80
Accuracy			0.62	634
Macro Avg	0.60	0.62	0.60	634
Weighted Avg	0.61	0.62	0.60	634

B. FIGURES

Attribute	Description
FAVC	Frequent consumption of high caloric food
FCVC	Frequency of consumption of vegetables
NCP	Number of main meals
CAEC	Consumption of food between meals
CH20	Consumption of water daily
CALC	Consumption of alcohol
SCC	Calories consumption monitoring
FAF	Physical activity frequency
TUE	Time using technology devices
MTRANS	Transportation used
Gender	Gender of the individual
Age	Age of the individual
Height	Height of the individual
Weight	Weight of the individual
SMOKE	Smoking habit
family_history_with_overweight	To determine if there are any hereditary issues

FIGURE 1. Features and their meaning

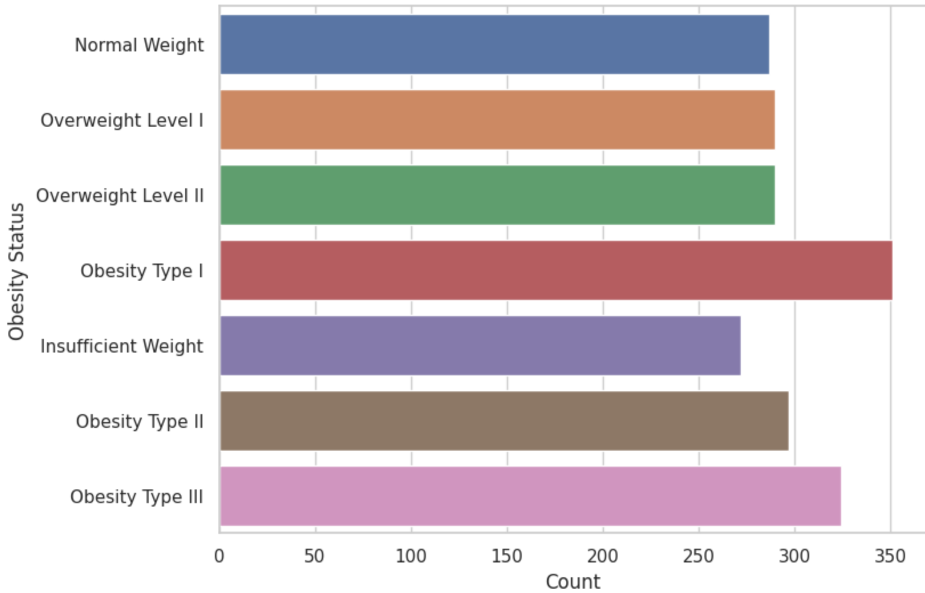

FIGURE 2. Data Instance Class Distributions

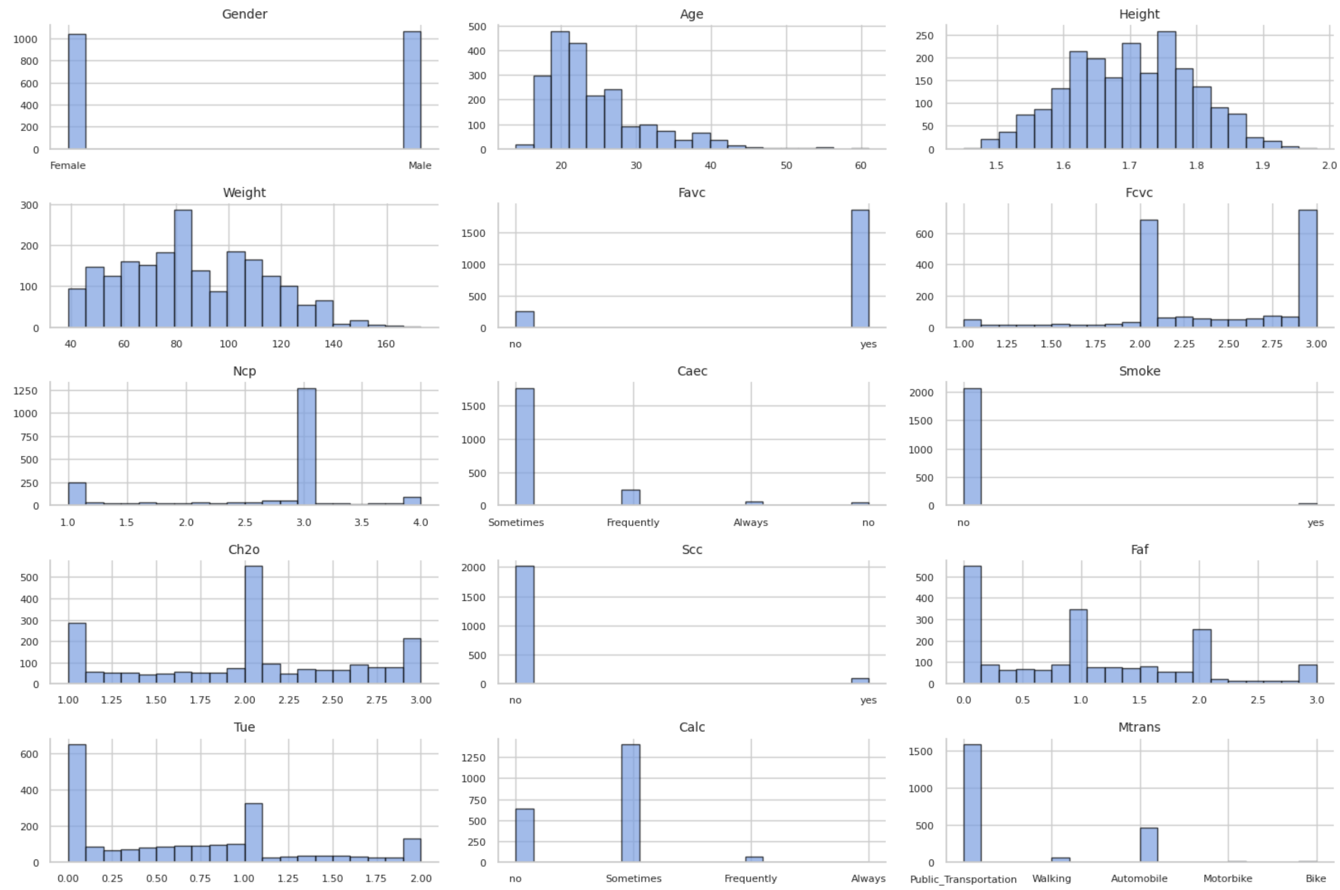

FIGURE 3. Histograms of attribute values

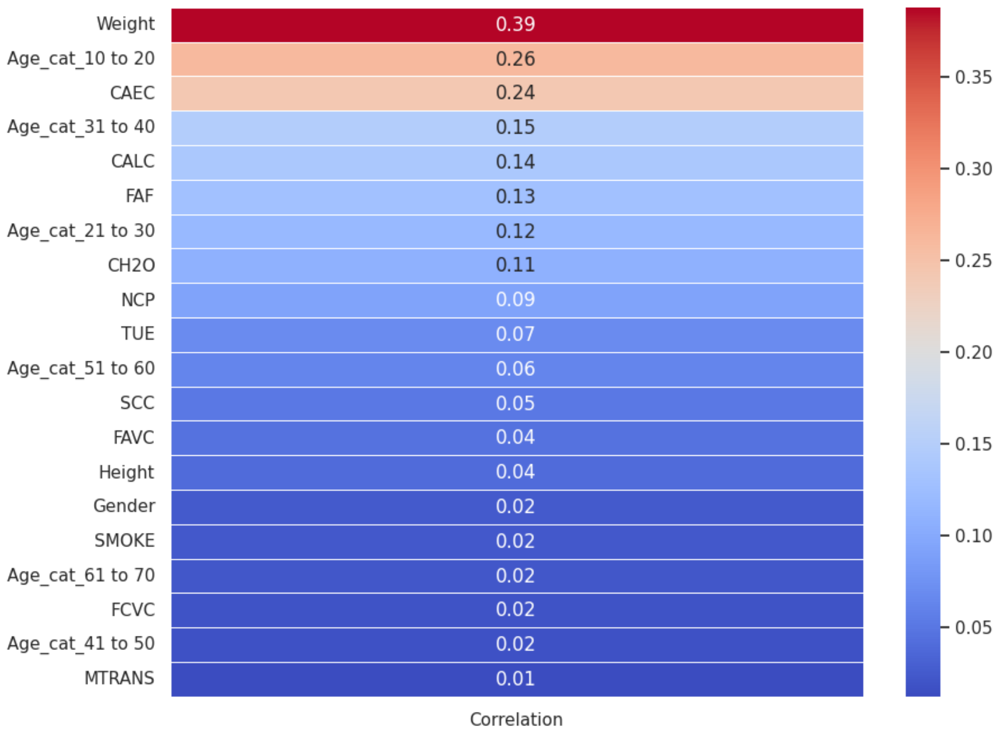

FIGURE 4. Correlation of attributes with Targets

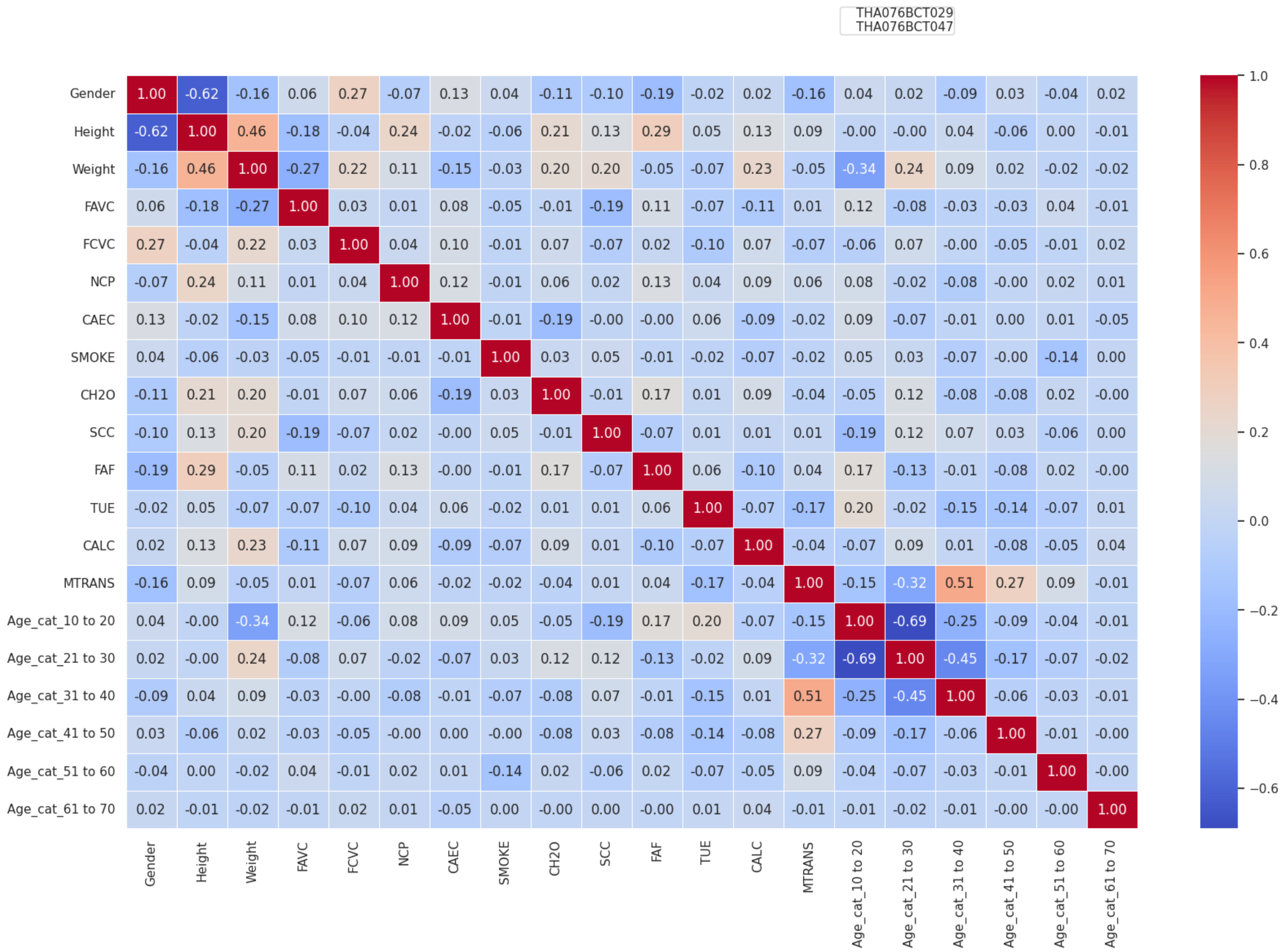

FIGURE 5. Correlation of attributes with each other

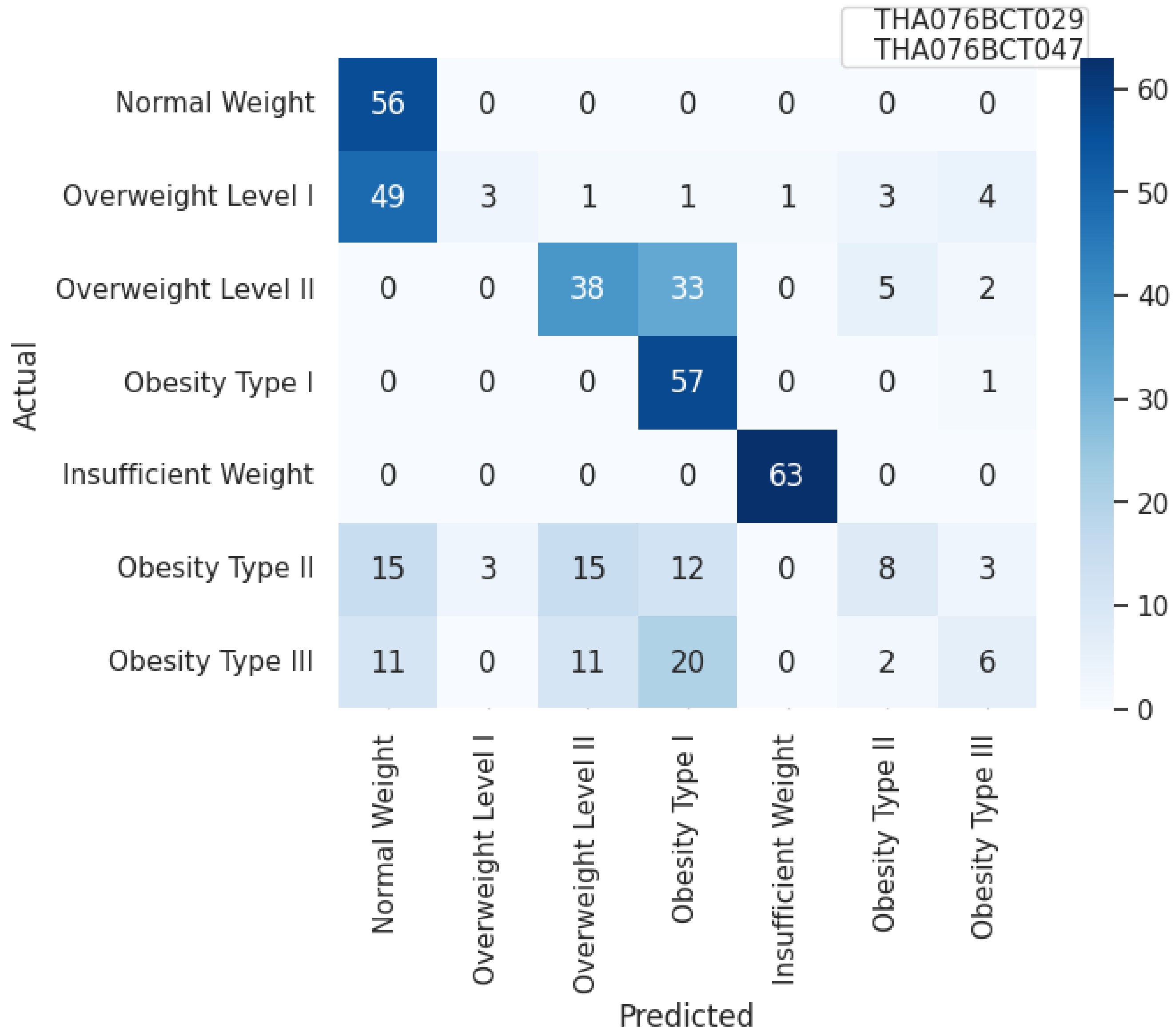

FIGURE 6. Gaussian Naive Bayes classifier

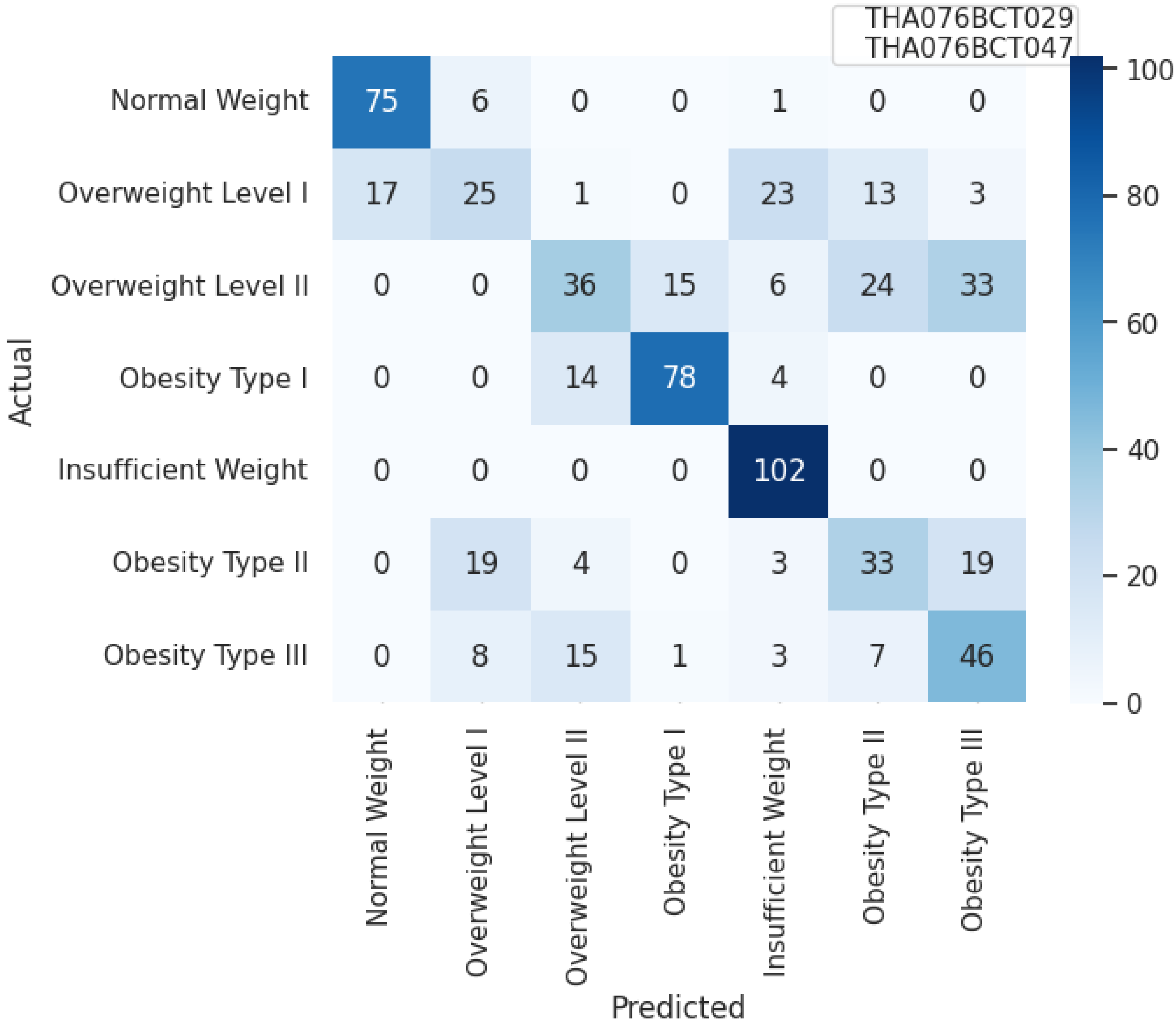

FIGURE 7. Hybrid Naive Bayes classifier

APPENDIX. CODE

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



# Data for the table


# Create the table
fig, ax = plt.subplots(figsize=(12, 8))
ax.axis('off')   # Hide axis

table_data = []
for i in range(len(attributes)):
    table_data.append([attributes[i], description[i]])



table.auto_set_font_size(False)
table.set_fontsize(12)
table.scale(1.2, 2)   # Adjust table size
plt.show()
data = pd.read_csv('/content/ObesityDataSet_raw_and_data_sinthetic.csv')
columns = data.columns
data.head(10)
data = data.drop(labels = ["family_history_with_overweight"],axis = 1)
fig, axes = plt.subplots(nrows=6, ncols=3, figsize=(15, 12))
plt.subplots_adjust(hspace=0.5)

# Plot histograms for each attribute
for ax, column in zip(axes.flatten(), data.columns):
    ax.hist(data[column], bins=20, color='#7b9fe0', alpha=0.7, edgecolor='black')
    ax.set_title(column.capitalize(), fontsize=10)
    ax.tick_params(axis='both', which='both', labelsize=8)
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)

# Remove empty subplots
if data.shape[1] < 30:
    for ax in axes.flatten()[data.shape[1]:]:
        ax.remove()

plt.tight_layout()
plt.show()
data['Age'].max(), data['Age'].min()
data['Age_cat'] = pd.cut(x=data['Age'], bins=[10,20, 30, 40, 50,60,70],
labels=['10 to 20', '21 to 30', '31 to 40',
'41 to 50','51 to 60', '61 to 70'])
data = data.drop('Age', axis=1)
data.head(5)
def get_ohe(input):
  if input=="Male":
    return 0
  else:
    return 1
```

```
data['Gender'] = data["Gender"].apply(get_ohe)
def get_sohe(input):
  if input=="yes":
    return 0
  else:
    return 1
data['SCC'] = data["SCC"].apply(get_sohe)
data['SMOKE'] = data["SMOKE"].apply(get_sohe)
data['FAVC'] = data["FAVC"].apply(get_sohe)
def get_freq(input):
  if input=="no":
    return 1
  elif input=="Always":
    return 2
  elif input=="Sometimes":
    return 3
  elif input=="Frequently":
    return 4
data['CAEC'] = data['CAEC'].apply(get_freq)
data['CALC'] = data['CALC'].apply(get_freq)
def get_trans(input):
  if input=="Public_Transportation":
    return 0
  elif input == "Walking":
    return 1
  elif input == "Automobile":
    return 2
  elif input == "Motorbike":
    return 3
  elif input =="Bike":
   return 4
data['MTRANS'] = data['MTRANS'].apply(get_trans)
ObesityCount = data.groupby('NObeyesdad').size()
ObesityCount
import seaborn as sns
import matplotlib.pyplot as plt

sns.set(style="whitegrid")   # Optional: Set a seaborn style
plt.figure(figsize=(8, 6))   # Optional: Set the figure size

# Replace underscores with spaces in the 'NObeyesdad' column
data1 = data.copy()
data1['NObeyesdad'] = data1['NObeyesdad'].str.replace('_', ' ')

sns.countplot(y='NObeyesdad', data=data1, orient='h')

plt.xlabel('Count')   # Optional: Set the x-axis label
plt.ylabel('Obesity Status')   # Optional: Set the y-axis label

plt.show()
new_data = pd.get_dummies(data,columns = ['Age_cat'])
filtered_data = new_data.drop(['NObeyesdad'],axis = 1)
filtered_data
filtered_data['nObeyesdad'] = data['NObeyesdad']
filtered_data
correlation = filtered_data.corr()
```

```
correlation
# Set the figure size (optional, adjust as needed)
plt.figure(figsize=(20, 12))

# Choose a color map for the heatmap (optional)
# You can find more colormaps at: https://matplotlib.org/stable/tutorials/colors/colormaps.html
cmap = 'coolwarm'

# Draw the heatmap
sns.heatmap(correlation,annot = True, fmt=".2f",cmap=cmap,linewidths=0.5)

# Add a title to the heatmap
legend_handles = [
plt.Line2D([], [], color='black', marker='o', markersize=10,
label='THA076BCT029\nTHA076BCT047',alpha = 0), # Remove the scatterplot marker
from the legend
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0.7, 1.1),
ncol=len(legend_handles), handlelength=0.4, borderpad=0.07)

# Rotate the x-axis labels for better readability (optional, adjust as needed)
plt.xticks(rotation=90)

# Show the plot
plt.show()
X = filtered_data.iloc[:,:-1]
y=filtered_data.iloc[:,-1]
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB,CategoricalNB
from sklearn.metrics import accuracy_score
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
nb_classifier = GaussianNB()

# Train the classifier
nb_classifier.fit(X_train, y_train)

# Make predictions on the test set
y_pred = nb_classifier.predict(X_test)
y_prob = nb_classifier.predict_proba(X_test)


# Calculate the accuracy of the classifier
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
from sklearn.metrics import confusion_matrix,classification_report
conf = confusion_matrix(y_test, y_pred)
# Set the colormap for the heatmap
cmap = 'Blues'
# Create a heatmap of the confusion matrix
sns.heatmap(conf, annot=True,cmap=cmap, xticklabels=classes, yticklabels=classes,fmt='d')
plt.xlabel('Predicted')
plt.ylabel('Actual')
legend_handles = [
plt.Line2D([], [], color='black', marker='o', markersize=10,
label='THA076BCT029\nTHA076BCT047',alpha = 0), # Remove the scatterplot marker
from the legend
```

```
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0.7, 1.1),
ncol=len(legend_handles), handlelength=0.4, borderpad=0.07)
plt.show
import numpy as np

report = classification_report(y_test, y_pred, target_names=classes)

# Format precision, recall, and f1-score values to two decimal places
report = report.replace('avg / total', 'avg/total')
report = report.replace('\n\n', '\n')

print("Classification Report:")
print(report)
continuous_columns = [ 'Height', 'Weight',
        'FCVC', 'NCP',  'CH2O','FAF', 'TUE',]

categorical_columns = ['Gender', 'FAVC', 'CAEC', 'SMOKE', 'SCC', 'CALC',
'MTRANS','Age_cat_10 to 20','Age_cat_21 to 30','Age_cat_31 to 40','Age_cat_41 to
50','Age_cat_51 to 60','Age_cat_61 to 70']




X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3,random_state = 4

# X_test_ = pd.concat([X_test.iloc[:594], X_test.iloc[596:]], axis = 0)
# y_test_ = pd.concat([y_test.iloc[:594], y_test.iloc[596:]], axis = 0)


X_train_gaussian = X_train[continuous_columns]
X_test_gaussian = X_test[continuous_columns]

X_train_categorical = X_train[categorical_columns]
X_test_categorical = X_test[categorical_columns]


model_GNB = GaussianNB()
model_GNB.fit(X_train_gaussian, y_train)
y_pred_gaussian = model_GNB.predict(X_test_gaussian)
# verbose_result(y_pred_gaussian, y_test_)

# print(build_classification_rep(y_pred_gaussian, y_test_))

model_CNB = CategoricalNB()
model_CNB.fit(X_train_categorical, y_train)
y_pred_categorical = model_CNB.predict(X_test_categorical)
# verbose_result(y_pred_categorical, y_test_)
# print(build_classification_rep(y_pred_categorical, y_test_))

count = 0
y_pred_hybrid = []
for pred_cat, pred_gauss in zip(y_pred_categorical, y_pred_gaussian):
    if pred_cat == pred_gauss:
        y_pred_hybrid.append(pred_cat)
```

```
    else:
      y_pred_hybrid.append(pred_gauss)
print("Accuracy of hybrid NB: ", (y_pred_hybrid == y_test).mean())
from sklearn.metrics import confusion_matrix,classification_report
conf = confusion_matrix(y_test, y_pred_hybrid)
# Set the colormap for the heatmap
cmap = 'Blues'
# Create a heatmap of the confusion matrix
sns.heatmap(conf, annot=True,cmap=cmap, xticklabels=classes,
yticklabels=classes,fmt='d')
plt.xlabel('Predicted')
plt.ylabel('Actual')
legend_handles = [
plt.Line2D([], [], color='black', marker='o', markersize=10,
label='THA076BCT029\nTHA076BCT047',alpha = 0), # Remove the scatterplot marker
from the legend
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0.7, 1.1),
ncol=len(legend_handles), handlelength=0.4, borderpad=0.07)
plt.show
import numpy as np


report = classification_report(y_test, y_pred_hybrid, target_names=classes)

# Format precision, recall, and f1-score values to two decimal places
report = report.replace('avg / total', 'avg/total')
report = report.replace('\n\n', '\n')

print("Classification Report:")
print(report)
```