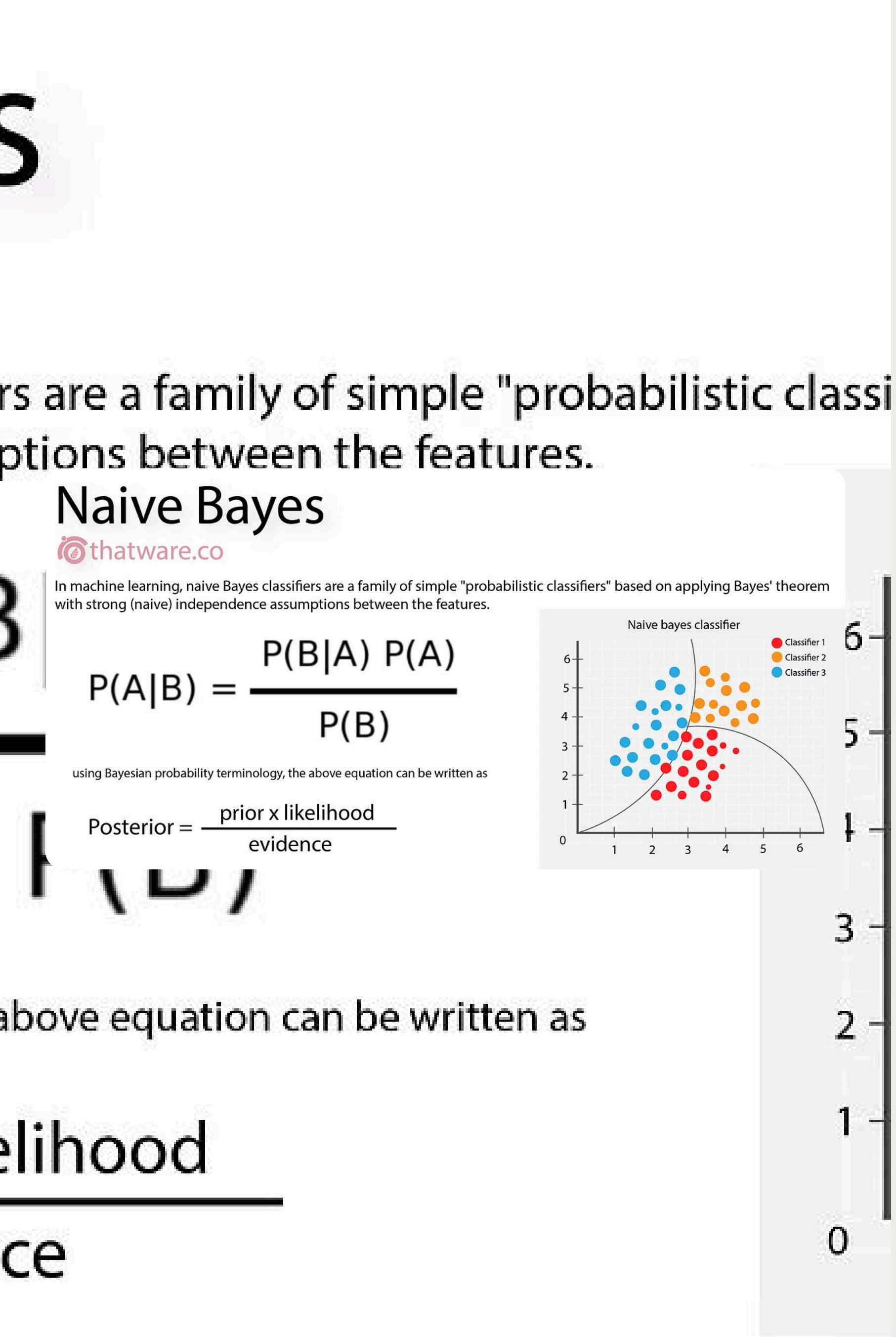


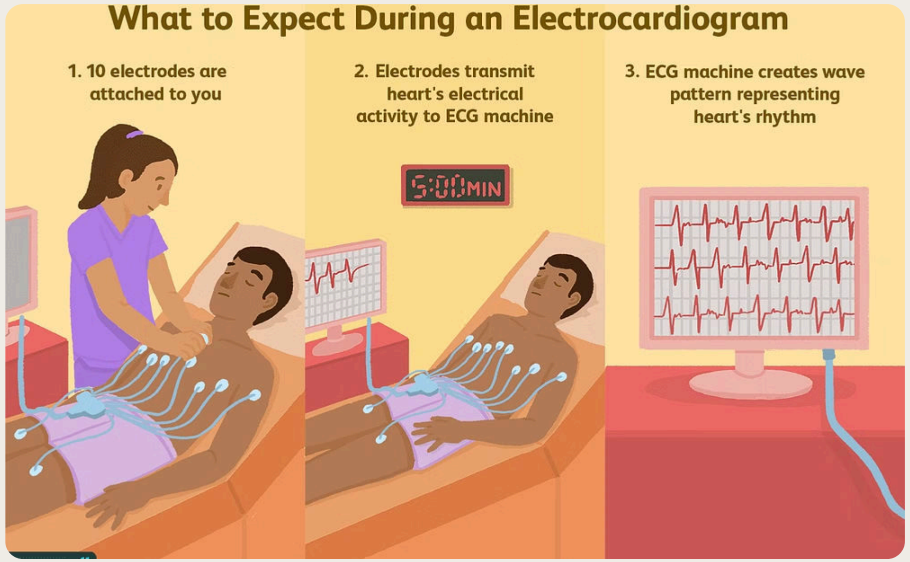
Introduction to Naive Bayes algorithm

Naive Bayes is a simple but powerful algorithm used for classification tasks in machine learning. It is based on the Bayes theorem with the "naive" assumption of independence between features. This algorithm is widely used in various applications, including text classification, spam filtering, and medical diagnosis.

by Kshitij Rastogi

Overview of Heart Attack Dataset

The Heart Attack Dataset contains various attributes such as age, gender, cholesterol levels, and other health metrics that can be used to predict the likelihood of a heart attack occurrence. Understanding the dataset's structure and the significance of each attribute is crucial for effective analysis and prediction using the Naive Bayes algorithm.



Heart Health

An illustration depicting the concept of heart health and well-being.

Preprocessing the dataset for Naive Bayes

Before applying the Naive Bayes algorithm, it's essential to preprocess the Heart Attack Dataset. This involves tasks such as handling missing data, normalizing features, and encoding categorical variables. The quality of preprocessing significantly impacts the accuracy and performance of the algorithm in the subsequent stages.

Data Cleaning

Removing or imputing missing values, handling outliers, and ensuring data integrity.

Feature Scaling

Normalizing features to ensure a uniform scale across the dataset.

Categorical Encoding

Converting categorical variables into numerical representations for algorithm processing.

Implementing Naive Bayes algorithm for analysis and prediction

Implementing the Naive Bayes algorithm involves training the model using the preprocessed Heart Attack Dataset. The algorithm's probabilistic nature makes it well-suited for analyzing the dataset's attributes and predicting the likelihood of heart attack occurrences based on the provided data.

Data Analysis

Utilizing the algorithm to gain insights into the dataset's patterns and relationships.

Prediction

Forecasting the likelihood of heart attack occurrences based on the trained model.

Evaluating the performance of Naive Bayes on Heart Attack Dataset

Assessing the performance of the Naive Bayes algorithm involves evaluating metrics such as accuracy, precision, recall, and F1 score. These metrics provide insights into how well the algorithm can classify instances of heart attack occurrences and non-occurrences based on the dataset's attributes.

Accuracy

Accuracy

The proportion of correct predictions out of the total predictions made.

Confusion Matrix

Confusion Matrix

A confusion matrix is **a table that is used to define the performance of a classification algorithm.**

Writing the machine learning code for Naive Bayes

Developing the machine learning code for Naive Bayes involves implementing the algorithm using a programming language such as Python or R. This code encompasses tasks such as dataset loading, preprocessing, model training, and performance evaluation using relevant metrics and visualizations.

1

Data Loading

Loading the Heart Attack Dataset and extracting necessary information for further processing.

2

Preprocessing

Performing data preprocessing tasks such as cleaning, scaling, and encoding.

3

Model Training

Training the Naive Bayes model using the preprocessed dataset.

4

Evaluation

Evaluating the model's performance using relevant metrics and visualizations.

Code :

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

df = pd.read_csv(r'C:\Users\Kshitij\Downloads\Heart Attack Analysis & Prediction\heart.csv')

df.head

df.isnull().sum()

df.info()

df[df.duplicated()]

df.drop_duplicates(keep = 'first', inplace = True)

x = (df.sex.value_counts())
print(f'Number of people having sex as 1 are {x[1]} and Number of people having sex as 0 are {x[0]}')
p = sns.countplot(data = df, x = 'sex')
plt.show()

x = (df.cp.value_counts())
print(x)
p = sns.countplot(data = df, x = 'cp')
plt.show()

plt.figure(figsize = (10,10))
sns.displot(df.age, color = 'red', label = "age", kde = True)
plt.legend()
```

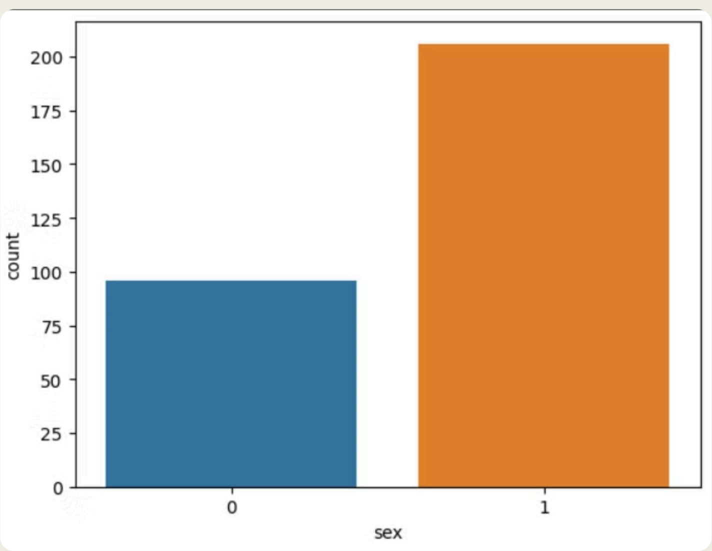
```
plt.figure(figsize = (10,10))
sns.displot(df.trtbps, color = 'green', label = "Resting Blood Pressure", kde = True)
plt.legend()

plt.figure(figsize = (10,10))
sns.displot(df[df['output']==0]['age'], color = 'green', kde = True)
sns.displot(df[df['output']==1]['age'], color = 'red', kde = True)
plt.title("Attack Vs Age")
plt.show()

plt.figure(figsize = (10,10))
sns.displot(df[df['output']==0]['chol'], color = 'green', kde = True)
sns.displot(df[df['output']==1]['chol'], color = 'red', kde = True)
plt.title("Attack Vs Age")
plt.show()

y = df.iloc[:, -1].values
X = df.iloc[:, 1:-1].values

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 40, test_size = 0.19)

from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(X_train, y_train)

y_pred = gnb.predict(X_test)
```

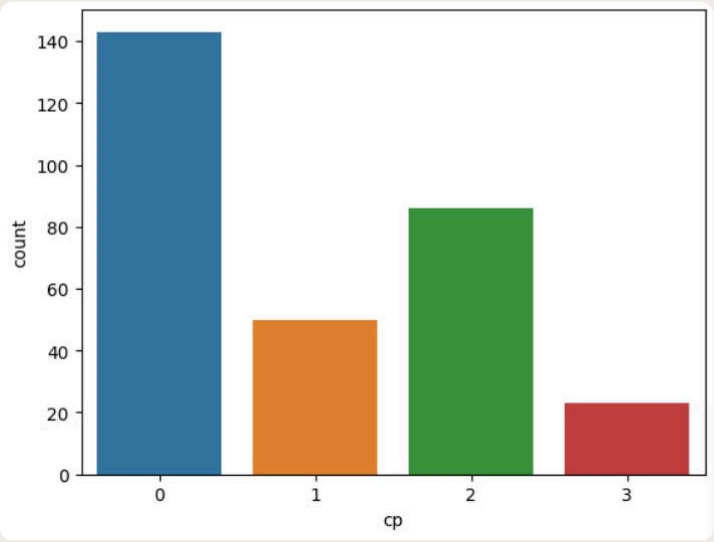
```
from sklearn.metrics import confusion_matrix
print("Confusion Matrix :\n",confusion_matrix(y_pred, y_test))

from sklearn.metrics import accuracy_score
print("Accuracy Score :\n",round(accuracy_score(y_pred, y_test)*100,2), "%")
```

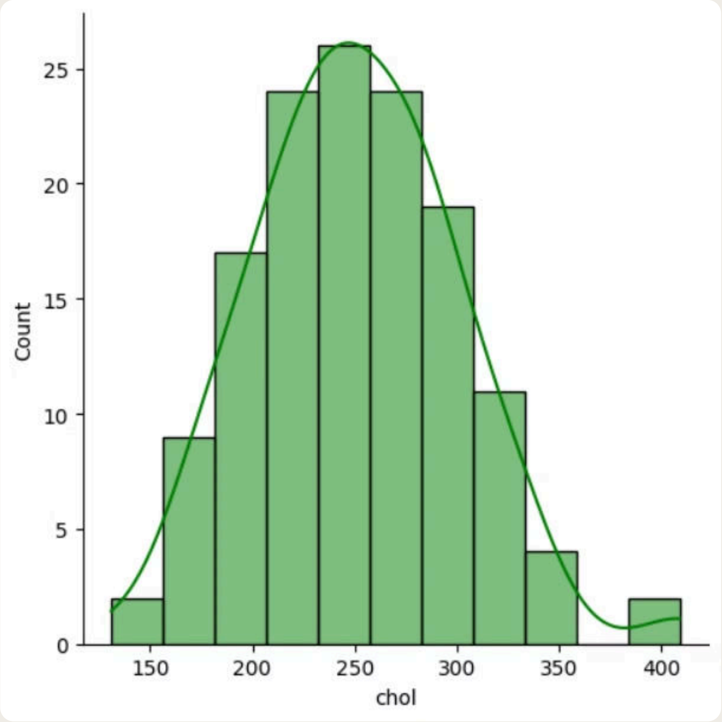
✓ 10.6s

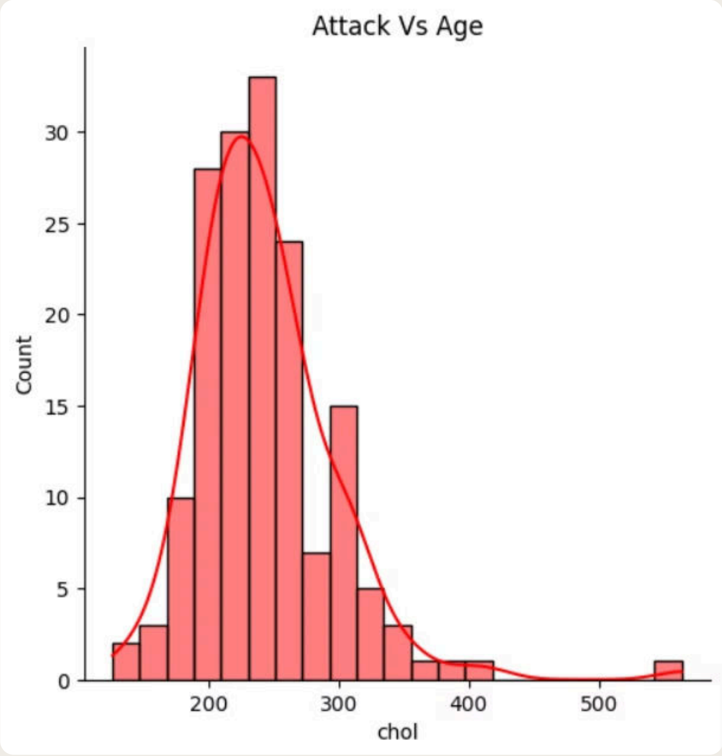
Output :

```
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trtbps    303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalachh  303 non-null    int64
 8   exng      303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slp       303 non-null    int64
 11  caa       303 non-null    int64
 12  thall     303 non-null    int64
 13  output    303 non-null    int64
dtypes: float64(1), int64(13)
```










```
Confusion Matrix :
 [[22  3]
 [ 2 31]]
```

```
Accuracy Score :
 91.38 %
```

Results and insights from the analysis

The analysis using the Naive Bayes algorithm on the Heart Attack Dataset has yielded valuable insights into the factors influencing heart attack occurrences. These insights can aid medical practitioners and researchers in understanding the correlations between various health metrics and the likelihood of heart attack incidents.

1. **Age and Risk**

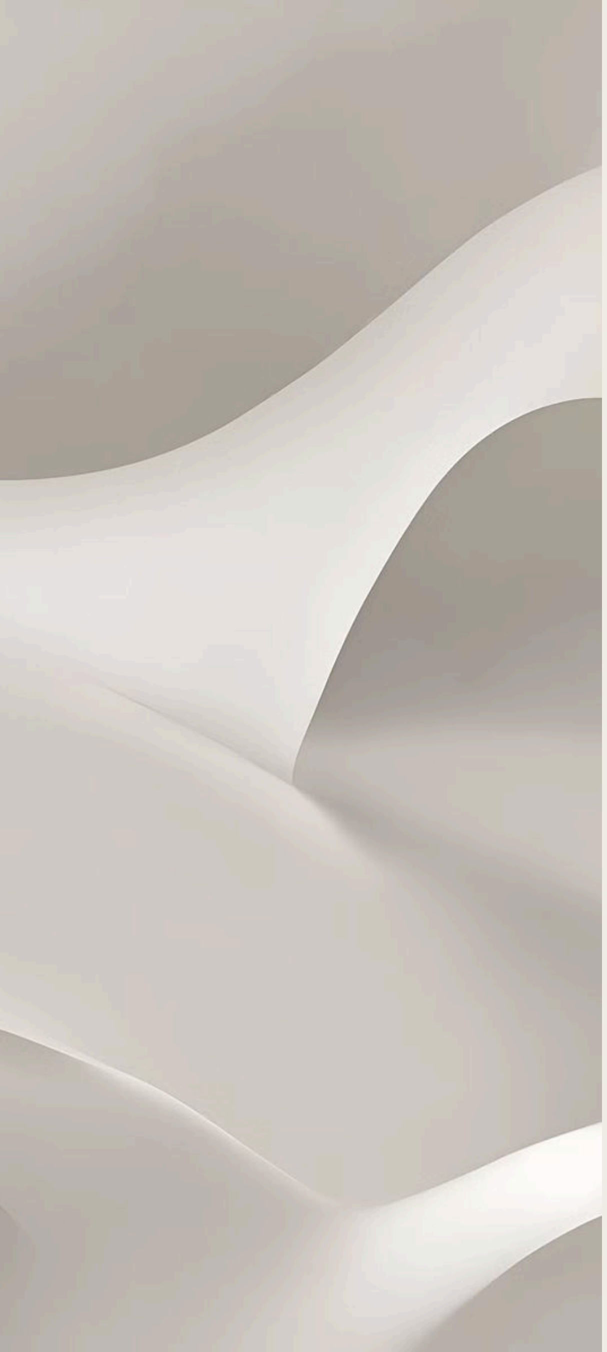
   The relationship between age and the likelihood of heart attacks.

2. **Cholesterol Levels**

   The impact of cholesterol levels on heart health and risk assessment.

3. **Gender Disparities**

   Insights into gender-specific risk factors for heart attacks.



Conclusion and future scope

The application of the Naive Bayes algorithm on the Heart Attack Dataset has provided valuable outcomes and opens avenues for future research. By gaining a deeper understanding of the factors influencing heart attack occurrences, the potential for early detection, prevention, and personalized treatment strategies becomes more promising.

Early Detection

Utilizing algorithmic predictions for proactive heart health monitoring.

Personalized Medicine

Using insights to develop targeted treatment plans for at-risk individuals.

Expanded Research

Exploring additional health metrics and their influence on heart attack occurrences.

Presented By :

Kshitij Rastogi (RA2111027010051)

Ansab Aalim (RA2111027010030)