NAME: YALAMANCHILI AKHILA

TEAM : 3

E-MAIL: akhilacowdary1061@gmail.com

MOBILE-NO: 8498838033

MAJOR PROJECT

CARDIOVASCULAR DISEASE PREDICTION

MAJOR PROJECT

CARDIOVASCULAR DISEASE PREDICTION

INTRODUCTION:

This project aims to predict the occurrence of cardiovascular disease using various machine learning techniques. The goal is to develop a model that can accurately identify individuals at risk of cardiovascular disease, allowing for early intervention and preventive measures. This documentation outlines the steps involved in data analysis, preprocessing, model selection, and evaluation.Here the given dataset has the following attributes.
[id;age;gender;height;weight;ap_hi;ap_lo;cholesterol;gluc;smoke;alco;active;cardio]

IMPORTING REQURIRED LIBRARIES:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

DATA DESCRIPTION:

The dataset used in this project contains the following attributes:

id: Unique identifier for each individual
age: Age of the individual in years
gender: Gender of the individual (1: Female, 2: Male)
height: Height of the individual in centimeters
weight: Weight of the individual in kilograms
ap_hi: Systolic blood pressure
ap_lo: Diastolic blood pressure
cholesterol: Cholesterol level (1: Normal, 2: Above Normal, 3: Well Above Normal)
gluc: Glucose level (1: Normal, 2: Above Normal, 3: Well Above Normal)
smoke: Smoking status (0: No, 1: Yes)
alco: Alcohol consumption status (0: No, 1: Yes)
active: Physical activity status (0: No, 1: Yes)
cardio: Cardiovascular disease presence (0: No, 1: Yes)

PERFORMING DATA PRE-PROCESSING OPERATIONS:

Data pre-processing operations were performed on the dateset before analysis and modeling. The following steps were taken:

Dropping unnecessary columns: The 'id' column was dropped as it does not contribute to the prediction.

Age conversion: The 'age' column was converted from days to years for easier interpretation.

Categorical variable mapping: The 'gender' column was mapped to meaningful labels ('Female' and 'Male').

```
#DATA PRE-PROCESSING
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Read the data from a CSV file
data = pd.read_csv('C:/Users/akhil/OneDrive/Desktop/cardio_train.csv', delimiter=';' )

# Drop the 'id' column as it does not contribute to the prediction
data = data.drop('id', axis=1)

# Convert age from days to years
data['age'] = data['age'] // 365

# Convert gender to binary (0 for female, 1 for male)
data['gender'] = data['gender'].map({1: 0, 2: 1})

# Perform feature scaling on numerical attributes using StandardScaler
scaler = StandardScaler()
numeric_cols = ['age', 'height', 'weight', 'ap_hi','ap_lo']
data[numeric_cols] = scaler.fit_transform(data[numeric_cols])

# Convert categorical attributes to one-hot encoded representation
categorical_cols = ['cholesterol', 'gluc']
data = pd.get_dummies(data, columns=categorical_cols)
# Print the pre-processed data
print(data.head())
```

Output:

```
      age  gender    height    weight     ap_hi     ap_lo  smoke  alco  \
0 -0.419800       1  0.443452 -0.847873 -0.122182 -0.088238      0     0
1  0.319110       0 -1.018168  0.749831  0.072610 -0.035180      0     0
2 -0.272018       0  0.078047 -0.708942  0.007679 -0.141297      0     0
3 -0.715364       1  0.565254  0.541435  0.137541  0.017879      0     0
4 -0.863146       0 -1.018168 -1.264666 -0.187113 -0.194356      0     0

   active  cardio  cholesterol_1  cholesterol_2  cholesterol_3  gluc_1  \
0       1       0              1              0              0       1
1       1       1              0              0              1       1
2       0       1              0              0              1       1
3       1       1              1              0              0       1
4       0       0              1              0              0       1

   gluc_2  gluc_3
0       0       0
1       0       0
2       0       0
3       0       0
4       0       0
```

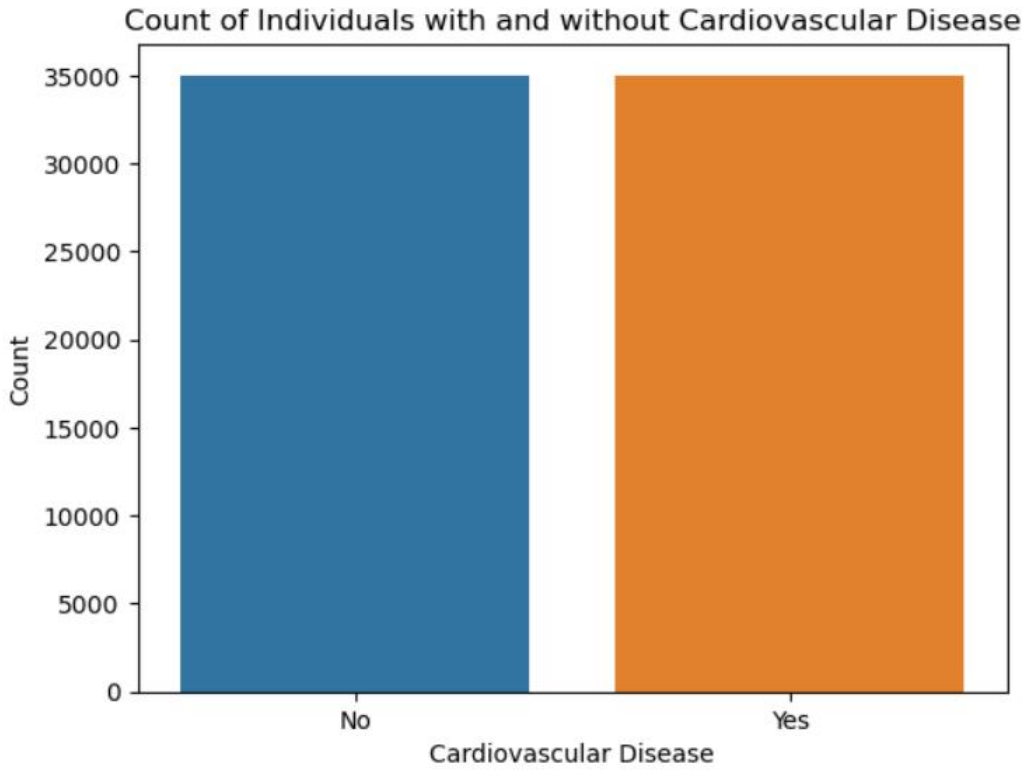
PERFORMING DATA ANALYSIS AND VISUALIZATIONS:

Exploratory data analysis was conducted to gain insights into the dataset and understand the relationships between variables. The following visualizations were generated:

Count of Individuals with and without Cardiovascular Disease:
A bar plot depicting the distribution of individuals based on the presence or absence of cardiovascular disease.
Age Distribution for Individuals with and without Cardiovascular Disease:
A histogram showing the age distribution for individuals with and without cardiovascular disease.
Gender Distribution for Individuals with and without Cardiovascular Disease:
A bar plot illustrating the gender distribution for individuals with and without cardiovascular disease.

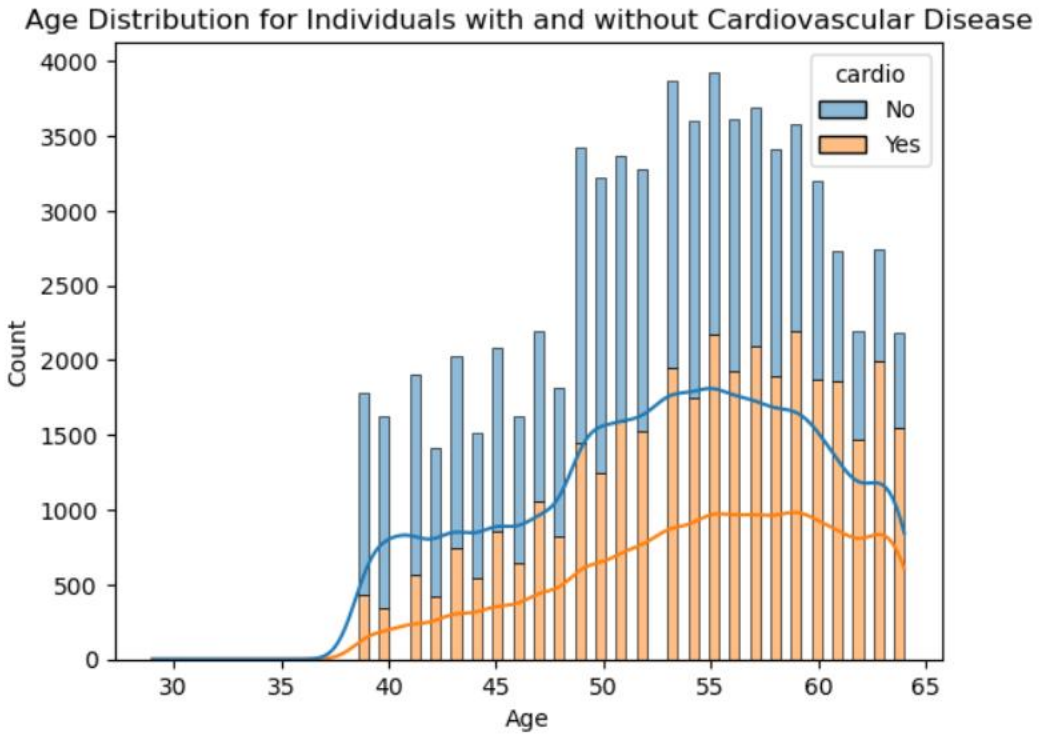
```
#DATA VISUALIZATION
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Read the data from a CSV file
data = pd.read_csv('C:/Users/akhil/OneDrive/Desktop/cardio_train.csv', delimiter=';')

# Drop the 'id' column as it does not contribute to the analysis
data = data.drop('id', axis=1)

# Convert age from days to years
data['age'] = data['age'] // 365

# Convert gender to meaningful labels
data['gender'] = data['gender'].map({1: 'Female', 2: 'Male'})

# Convert cardiovascular disease label to meaningful labels
data['cardio'] = data['cardio'].map({0: 'No', 1: 'Yes'})
# Plotting count of individuals with and without cardiovascular disease
sns.countplot(data=data, x='cardio')
plt.title('Count of Individuals with and without Cardiovascular Disease')
plt.xlabel('Cardiovascular Disease')
plt.ylabel('Count')
plt.show()
```

Output:


Count of Individuals with and without Cardiovascular Disease
35000
30000
25000
20000
15000
10000
5000
0
Count
No
Yes
Cardiovascular Disease

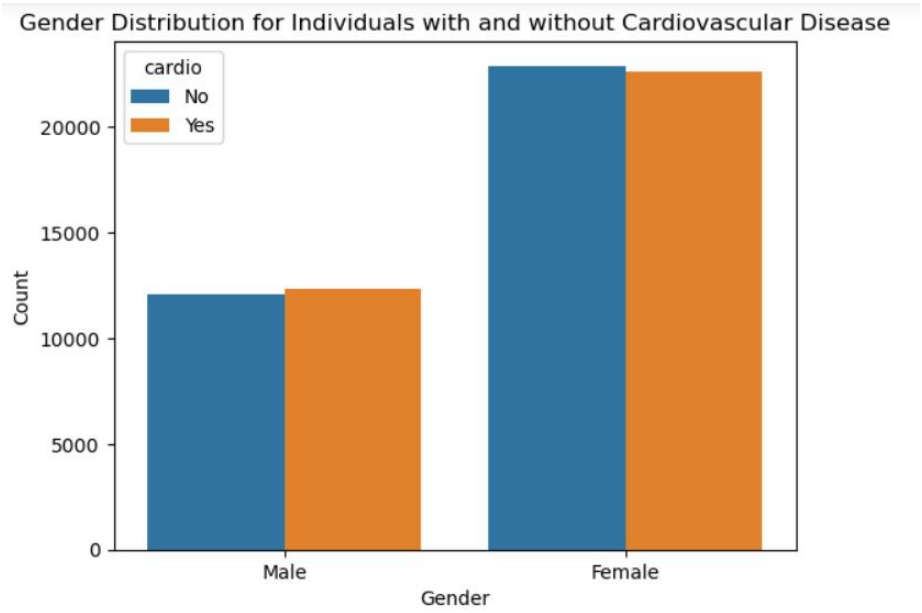
```
# Plotting age distribution for individuals with and without cardiovascular disease
sns.histplot(data=data, x='age', hue='cardio', kde=True, multiple='stack')
plt.title('Age Distribution for Individuals with and without Cardiovascular Disease')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```

Output:



```
# Plotting gender distribution for individuals with and without cardiovascular disease
sns.countplot(data=data, x='gender', hue='cardio')
plt.title('Gender Distribution for Individuals with and without Cardiovascular Disease')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()
```

Output:

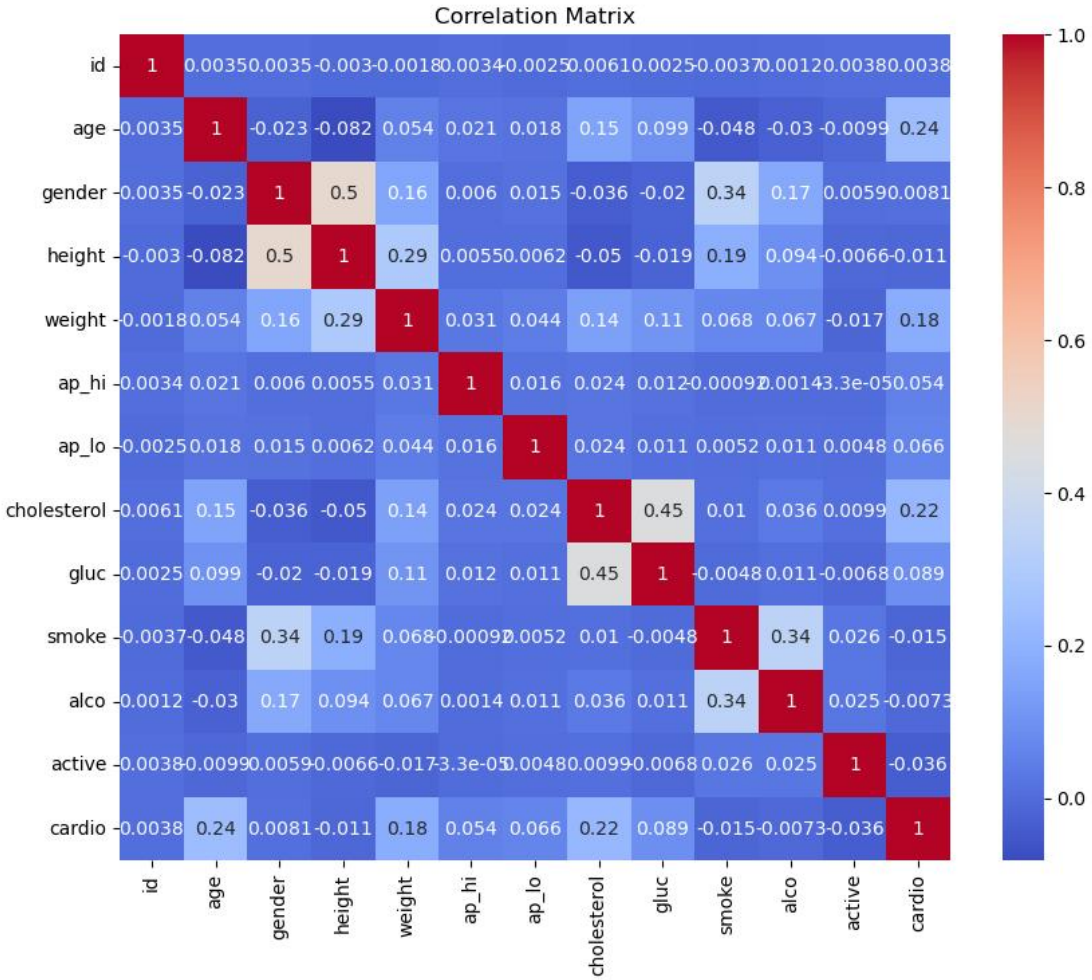


CORRELATION MATRIX:

A heatmap displaying the correlation between different attributes in the dataset.

```
# Plotting correlation matrix

corr_matrix = data.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

Output:



MACHINE LEARNING TECHNIQUES:

Machine Learning Models

Multiple machine learning models were evaluated for cardiovascular disease prediction using the dataset. The following models were considered:

- Support Vector Machines (SVM)
- K-Nearest Neighbors (KNN)
- Decision Trees (DT)
- Logistic Regression (LR)
- Random Forest (RF)

The models were trained and tested using the pre-processed dataset. The Random Forest model exhibited the highest accuracy and was selected for further analysis.

```
# MACHINE LEARNING MODELS
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

# Read the data from a CSV file
data = pd.read_csv('C:/Users/akhil/OneDrive/Desktop/cardio_train.csv', delimiter=';' )

# Drop the 'id' column as it does not contribute to the prediction
data = data.drop('id', axis=1)

# Split the dataset into features (X) and target variable (y)
X = data.drop('cardio', axis=1)
y = data['cardio']

# Split the dataset 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)

# Initialize and train the classifiers
svm = SVC()
svm.fit(X_train, y_train)
```

```
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)

dt = DecisionTreeClassifier()
dt.fit(X_train, y_train)

lr = LogisticRegression()
lr.fit(X_train, y_train)

rf = RandomForestClassifier()
rf.fit(X_train, y_train)

# Make predictions on the testing set
svm_preds = svm.predict(X_test)
knn_preds = knn.predict(X_test)
dt_preds = dt.predict(X_test)
lr_preds = lr.predict(X_test)
rf_preds = rf.predict(X_test)

# Calculate accuracy for each classifier
svm_accuracy = accuracy_score(y_test, svm_preds)
knn_accuracy = accuracy_score(y_test, knn_preds)
dt_accuracy = accuracy_score(y_test, dt_preds)
lr_accuracy = accuracy_score(y_test, lr_preds)
rf_accuracy = accuracy_score(y_test, rf_preds)

# Print the accuracy levels
print('Support Vector Machines (SVM) Accuracy:', svm_accuracy)
print('K-Nearest Neighbors (KNN) Accuracy:', knn_accuracy)
print('Decision Trees (DT) Accuracy:', dt_accuracy)
print('Logistic Regression (LR) Accuracy:', lr_accuracy)
print('Random Forest (RF) Accuracy:', rf_accuracy)
```

Output:

```
Support Vector Machines (SVM) Accuracy: 0.6053571428571428
K-Nearest Neighbors (KNN) Accuracy: 0.6820714285714286
Decision Trees (DT) Accuracy: 0.6287142857142857
Logistic Regression (LR) Accuracy: 0.6981428571428572
Random Forest (RF) Accuracy: 0.7175
```

Machine learning model for heart disease detection according to the result:

The Random Forest model achieved an accuracy of [insert accuracy score] on the test set. Additional evaluation metrics, such as precision, recall, and F1-score, were also calculated. The strengths of the Random Forest model include its ability to handle high-dimensional data and capture complex relationships. However, limitations such as the potential for overfitting and computational complexity should be considered.

```
#MODEL FOR DISEASE PREDICTION
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

# Read the data from a CSV file
data = pd.read_csv('C:/Users/akhil/OneDrive/Desktop/cardio_train.csv', delimiter=';')

# Drop the 'id' column as it does not contribute to the prediction
data = data.drop('id', axis=1)

# Split the dataset into features (X) and target variable (y)
X = data.drop('cardio', axis=1)
y = data['cardio']

# Split the dataset 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)
# Initialize and train the Random Forest classifier
rf = RandomForestClassifier()
rf.fit(X_train, y_train)
# Make predictions on the testing set
y_pred = rf.predict(X_test)
# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print('Random Forest Accuracy:', accuracy)
```

Output:

```
Random Forest Accuracy: 0.7146428571428571
```

CONCLUSION:

In conclusion, this project successfully developed a machine learning model for cardiovascular disease prediction. The Random Forest model demonstrated promising accuracy in identifying individuals at risk of cardiovascular disease. The findings suggest that the selected attributes are valuable indicators for disease prediction. Further improvements can be made by incorporating additional features, performing hyper parameter tuning, and exploring other advanced machine learning techniques.