Project Deliverable 4 (Final)
Project Title: Cardiovascular Disease Prediction

Nur Alam (ID-1189508)
Department of Computer Science
Lakehead University

Introduction

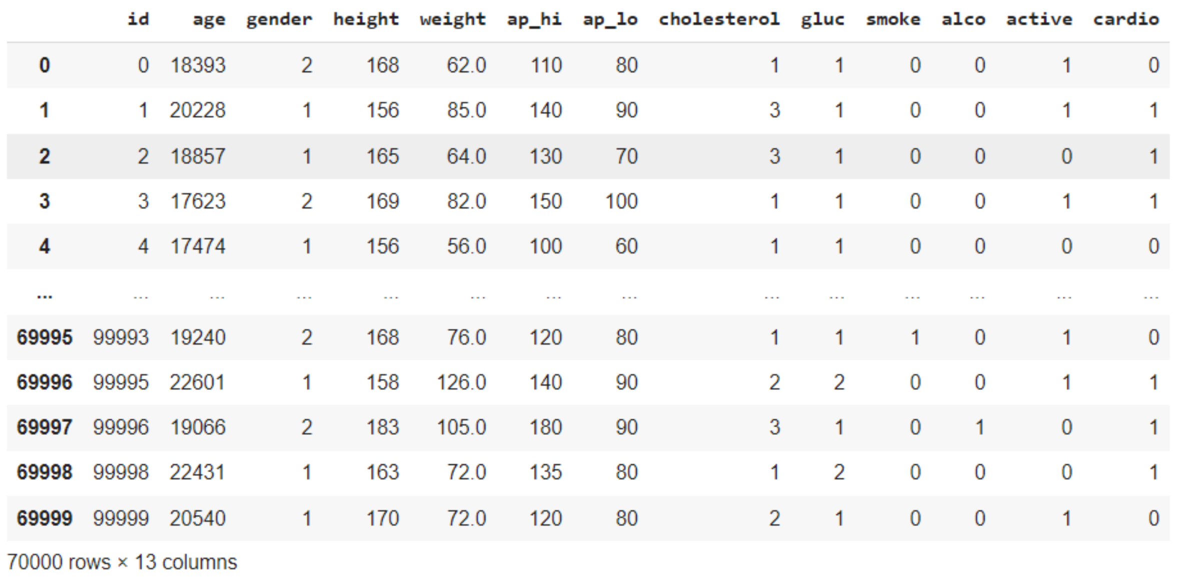
In the ever-changing world of healthcare, the combination of advanced technology and medical science is reshaping how we predict diseases, especially in cardiovascular health. Recent years have seen significant progress in using machine learning algorithms to uncover patterns in vast datasets. These algorithms, known for their ability to process complex information, play a crucial role in predicting cardiovascular diseases, a major cause of illness and death worldwide. Machine learning's role in predicting cardiovascular diseases is rooted in its ability to analyze a wide range of patient data. Beyond traditional medical records, these algorithms consider various factors like personal medical history, lifestyle choices, and physiological parameters. This comprehensive approach helps create sophisticated models that can identify subtle risk factors and predict the likelihood of cardiovascular events more accurately than traditional methods. The impact of this approach goes beyond prediction, signaling a shift in healthcare strategies. Early detection, thanks to machine learning algorithms, not only gives a crucial time advantage but also empowers healthcare professionals with personalized insights. Equipped with this information, medical practitioners can implement targeted interventions and preventive measures tailored to each patient, marking a new era of precision medicine. As the collaboration between machine learning and cardiovascular health advances, the future points to more than just accurate predictions – it promises proactive measures. This evolution has the potential to transform preventive medicine. With each improvement in machine learning algorithms, the possibility of identifying, mitigating, and preventing cardiovascular diseases at an individual level becomes more tangible. The ongoing partnership between technology and healthcare not only bodes well for individuals' well-being but also holds the potential to reshape the broader landscape of global public health.

Goal / What I'm trying to prove

The main goal of this project is to use machine learning to predict heart disease accurately. Understanding the complexities of heart health and its vulnerability to various conditions, we aim to develop effective tools for diagnosis and prediction. Aligned with global health efforts, particularly those led by the World Health Organization, we're focusing on preventing and managing heart diseases through improved risk assessment, standardized treatment, and enhanced healthcare systems. By leveraging innovative techniques like machine learning and vast patient data, we want to evaluate individual risks of heart disease. We're using various machine-learning algorithms, such as decision trees, random forests, k-nearest neighbors (KNN) and support vector machine and popular Python libraries like Pandas and Scikit-Learn. Data privacy is a top priority in handling medical data. Through data cleaning, adjustments, and exploratory analysis, our ultimate goal is to create a model that accurately predicts the likelihood of developing heart disease. We're simplifying and optimizing the process by extracting meaningful features and reducing the complexity of the data. This project aims to showcase the potential of machine learning in improving heart disease prediction and management, contributing valuable insights at the intersection of healthcare and technology.

Dataset

The dataset consists of comprehensive health information for 70,000 individuals, with each person represented by a unique identifier and 11 features, including a target variable indicating the presence or absence of cardiovascular disease (CVD). Here's an in-depth look at the dataset:



	id	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active	cardio
0	0	18393	2	168	62.0	110	80	1	1	0	0	1	0
1	1	20228	1	156	85.0	140	90	3	1	0	0	1	1
2	2	18857	1	165	64.0	130	70	3	1	0	0	0	1
3	3	17623	2	169	82.0	150	100	1	1	0	0	1	1
4	4	17474	1	156	56.0	100	60	1	1	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
69995	99993	19240	2	168	76.0	120	80	1	1	1	0	1	0
69996	99995	22601	1	158	126.0	140	90	2	2	0	0	1	1
69997	99996	19066	2	183	105.0	180	90	3	1	0	1	0	1
69998	99998	22431	1	163	72.0	135	80	1	2	0	0	0	1
69999	99999	20540	1	170	72.0	120	80	2	1	0	0	1	0

70000 rows × 13 columns

Fig 1: Cardiovascular Disease dataset

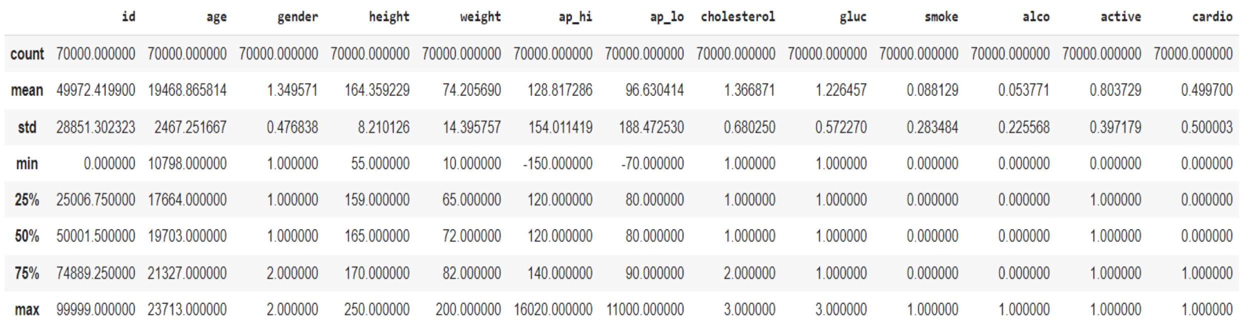
1. id (Unique Identifier): This serves as a distinctive identifier for each individual, facilitating data management and differentiation between records.
2. age (Age in Days): Representing age in days, this feature can be converted into years for practical interpretation. Age is a critical factor in predicting cardiovascular diseases, given its strong correlation with health outcomes.
3. gender: A categorical variable denoting the gender of everyone. Careful examination of encoding is necessary. Gender is often considered in health-related predictions due to its potential influence on cardiovascular health.
4. height (Height in cm): This numerical feature provides insight into an individual's physical stature, which can be a relevant factor in predicting cardiovascular health.
5. weight (Weight in kg): Another numerical feature, weight is crucial for calculating Body Mass Index (BMI), contributing to the understanding of an individual's overall health and potential cardiovascular risk.
6. ap_hi (Systolic Blood Pressure): This numerical feature indicates the systolic blood pressure, a vital metric in cardiovascular health. Elevated systolic blood pressure is a well-established risk factor for CVD.
7. ap_lo (Diastolic Blood Pressure): Representing diastolic blood pressure, this numerical feature complements systolic pressure in assessing overall blood pressure health, a key determinant of cardiovascular risk.
8. cholesterol: A categorical variable indicating blood cholesterol levels. Elevated cholesterol is a known risk factor for cardiovascular diseases and is integral to risk assessment.
9. gluc (Blood Glucose Level): This numerical feature provides information on blood glucose levels, offering insights into potential diabetes risk, a significant contributor to cardiovascular diseases.
10. smoke (Smoking Status): This binary feature denotes smoking status, a well-established risk factor for cardiovascular diseases.
11. alco (Alcohol Consumption Status): This binary feature signifies alcohol consumption status, contributing to the overall assessment of lifestyle-related cardiovascular risk factors.
12. active (Physical Activity): Indicating whether an individual engages in physical activity, this binary feature is vital, considering the protective role of exercise against cardiovascular diseases.
13. cardio (Target - Presence/Absence of Cardiovascular Disease): This target variable classifies individuals as either having (1) or not having (0) cardiovascular disease. This is the focal point for predictive modeling, aiming to leverage the other features to identify patterns and make accurate predictions regarding cardiovascular health.

In essence, this dataset encompasses a diverse set of health-related features, offering a comprehensive view of factors influencing cardiovascular health. Analyzing and modeling this dataset could uncover valuable insights, guiding efforts in cardiovascular disease prevention and management.

Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is an approach to analyzing and visualizing data sets to summarize their main characteristics, often with the help of statistical graphics and other data visualization methods. The primary goal of EDA is to gain insights into the underlying structure of the data, detect patterns, identify anomalies, and formulate hypotheses that can inform further analysis.

Initially, we can visualize the descriptive statistics of our dataset to gain a comprehensive understanding of its characteristics. This includes exploring key metrics such as the mean, standard deviation, minimum and maximum values, as well as the 25th, 50th (median), and 75th percentiles for each feature. Visualizing these statistics can provide insights into the central tendencies, spread, and distribution of the data. This exploratory visualization lays the groundwork for identifying potential outliers, assessing the range of values, and informing subsequent data analysis and preprocessing steps.

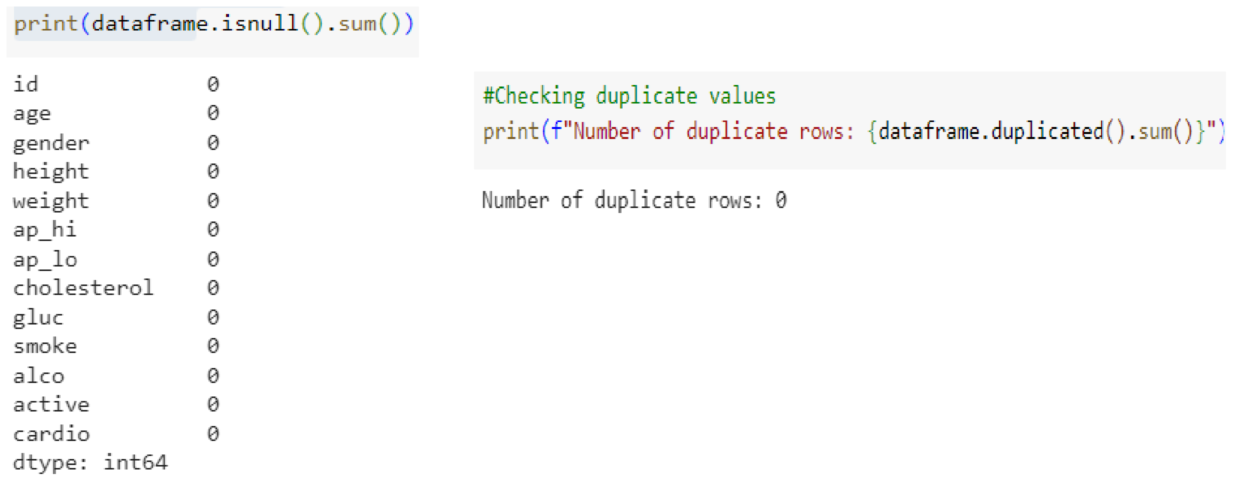


	id	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active	cardio
count	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000	70000.000000
mean	49972.419900	19468.865814	1.349571	164.359229	74.205690	128.817286	96.630414	1.366871	1.226457	0.088129	0.053771	0.803729	0.499700
std	28851.302323	2467.251667	0.476838	8.210126	14.395757	154.011419	188.472530	0.680250	0.572270	0.283484	0.225568	0.397179	0.500003
min	0.000000	10798.000000	1.000000	55.000000	10.000000	-150.000000	-70.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000
25%	25006.750000	17664.000000	1.000000	159.000000	65.000000	120.000000	80.000000	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000
50%	50001.500000	19703.000000	1.000000	165.000000	72.000000	120.000000	80.000000	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000
75%	74889.250000	21327.000000	2.000000	170.000000	82.000000	140.000000	90.000000	2.000000	1.000000	0.000000	0.000000	1.000000	1.000000
max	99999.000000	23713.000000	2.000000	250.000000	200.000000	16020.000000	11000.000000	3.000000	3.000000	1.000000	1.000000	1.000000	1.000000

Fig 2: Statistics of each feature in our Dataset

The above statistics offer a comprehensive overview of the dataset's key features. The 'id' column serves as a unique identifier for everyone, ranging from 0 to 99999. The 'age' column represents the age of individuals in days, with a mean of approximately 19468.87 days and a standard deviation of 2467.25 days. The 'gender' column indicates binary gender (1 for female, 2 for male), where the mean of 1.35 suggests a predominantly female representation. The 'height' and 'weight' columns provide information on individuals' physical characteristics, with mean values of 164.36 cm and 74.21 kg, respectively. The 'ap_hi' and 'ap_lo' columns represent systolic and diastolic blood pressure, with notable variability and potential outliers, as reflected in the large standard deviations and extreme values. The 'cholesterol' and 'gluc' columns categorize cholesterol and glucose levels, with mean values indicating generally normal levels. The 'smoke,' 'alco,' and 'active' columns represent lifestyle factors, and 'cardio' is the target variable indicating the presence (1) or absence (0) of cardiovascular disease. The statistics provide a foundation for understanding the dataset's characteristics, suggesting potential areas for data cleaning, exploration, and analysis, such as handling outliers in blood pressure measurements and further investigating the distribution of age.

#Checking 'null values' and 'duplicate values':



```
print(dataframe.isnull().sum())
```

```
id             0
age            0
gender         0
height         0
weight         0
ap_hi          0
ap_lo          0
cholesterol    0
gluc           0
smoke          0
alco           0
active         0
cardio         0
dtype: int64
```

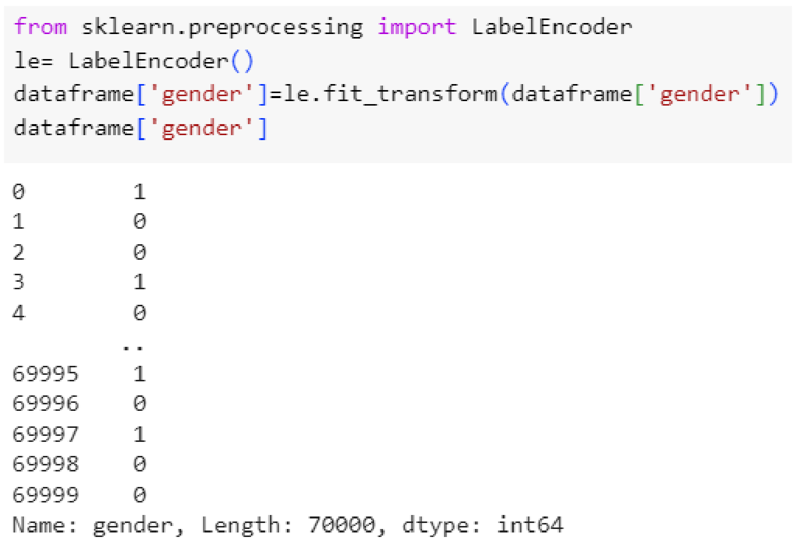
```
#Checking duplicate values
print(f"Number of duplicate rows: {dataframe.duplicated().sum()}")
```

```
Number of duplicate rows: 0
```

Fig 3: Screenshot of checking missing/null values and duplicate values.

We have conducted an initial examination of our dataset and found no null values, indicating that our data is complete. The absence of null values is crucial for the robustness of our models. In the absence of complete data, addressing missing values would be a vital step during the data preprocessing phase. Additionally, we performed checks for duplicate values and identified none, ensuring the integrity of our dataset. This clean and complete dataset provides a solid foundation for subsequent exploratory analysis and modeling endeavors.

#Label Encoding for "gender" attribute:



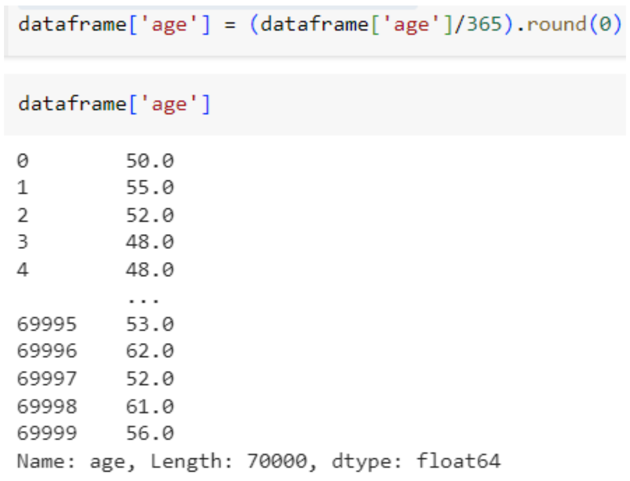
```
from sklearn.preprocessing import LabelEncoder
le= LabelEncoder()
dataframe['gender']=le.fit_transform(dataframe['gender'])
dataframe['gender']
```

```
0        1
1        0
2        0
3        1
4        0
        ..
69995    1
69996    0
69997    1
69998    0
69999    0
Name: gender, Length: 70000, dtype: int64
```

Fig 4: Screenshot of Label Encoding for "gender" attribute

The LabelEncoding for the 'gender' attribute involves transforming the categorical variable 'gender' into numerical values for ease of processing in machine learning models. In the transformed series, values have been encoded as 1 and 0, with 1 typically representing one category (e.g., male) and 0 representing the other category (e.g., female). This encoding simplifies the representation of gender, allowing algorithms to better interpret and utilize this information.

# Converting ‘age’ from days to year:



```
dataframe['age'] = (dataframe['age']/365).round(0)
```

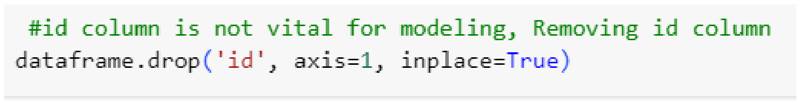
```
dataframe['age']
```

```
0        50.0
1        55.0
2        52.0
3        48.0
4        48.0
         ...
69995    53.0
69996    62.0
69997    52.0
69998    61.0
69999    56.0
Name: age, Length: 70000, dtype: float64
```

Fig 5: Screenshot of converting ‘age’ from days to year

In the process of enhancing the interpretability of our dataset, we converted the 'age' attribute from its original representation in days to years. This transformation involved dividing the existing 'age' values, initially measured in days, by 365 and rounding the result. The resulting 'age' series now provides a more intuitive representation of individuals' ages in years, making it conducive to analysis and interpretation.

#Removing ‘id’ column:

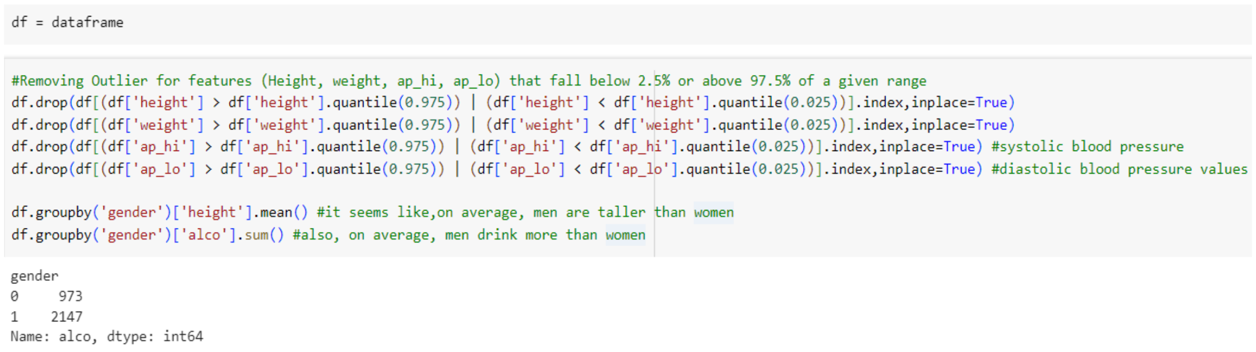


```
 #id column is not vital for modeling, Removing id column
dataframe.drop('id', axis=1, inplace=True)
```

Fig 6: Screenshot of removing ‘id’ column

The 'id' column typically serves as a unique identifier for each entry but does not contribute to the predictive capabilities of models. We excluded the 'id' column from our dataset using the drop method.

# Removing Outliers:



```
df = dataframe
```

```
#Removing Outlier for features (Height, weight, ap_hi, ap_lo) that fall below 2.5% or above 97.5% of a given range
df.drop(df[(df['height'] > df['height'].quantile(0.975)) | (df['height'] < df['height'].quantile(0.025))].index,inplace=True)
df.drop(df[(df['weight'] > df['weight'].quantile(0.975)) | (df['weight'] < df['weight'].quantile(0.025))].index,inplace=True)
df.drop(df[(df['ap_hi'] > df['ap_hi'].quantile(0.975)) | (df['ap_hi'] < df['ap_hi'].quantile(0.025))].index,inplace=True) #systolic blood pressure
df.drop(df[(df['ap_lo'] > df['ap_lo'].quantile(0.975)) | (df['ap_lo'] < df['ap_lo'].quantile(0.025))].index,inplace=True) #diastolic blood pressure values

df.groupby('gender')['height'].mean() #it seems like,on average, men are taller than women
df.groupby('gender')['alco'].sum() #also, on average, men drink more than women
```

```
gender
0     973
1    2147
Name: alco, dtype: int64
```

Fig 7: Screenshot of removing outliers

In our dataset, we've addressed outliers in specific features such as 'height,' 'weight,' 'ap_hi' (systolic blood pressure), and 'ap_lo' (diastolic blood pressure). Outliers were identified by comparing values to the 2.5% and 97.5% quantiles of each feature's distribution, and entries outside this range were removed. This process ensures a more representative dataset, suitable for robust analysis and modeling by mitigating the impact of extreme values. Additionally, we explored gender-related patterns through two group-by operations. The first found that, on average, men are taller than women, as indicated by the mean height for each gender. The second operation showed that men tend to consume more alcohol than women, providing insights into gender-specific trends. These observations enhance our understanding of the dataset, offering valuable considerations for further analyses or modeling strategies that account for gender-related patterns.

#Correlation Matrix:

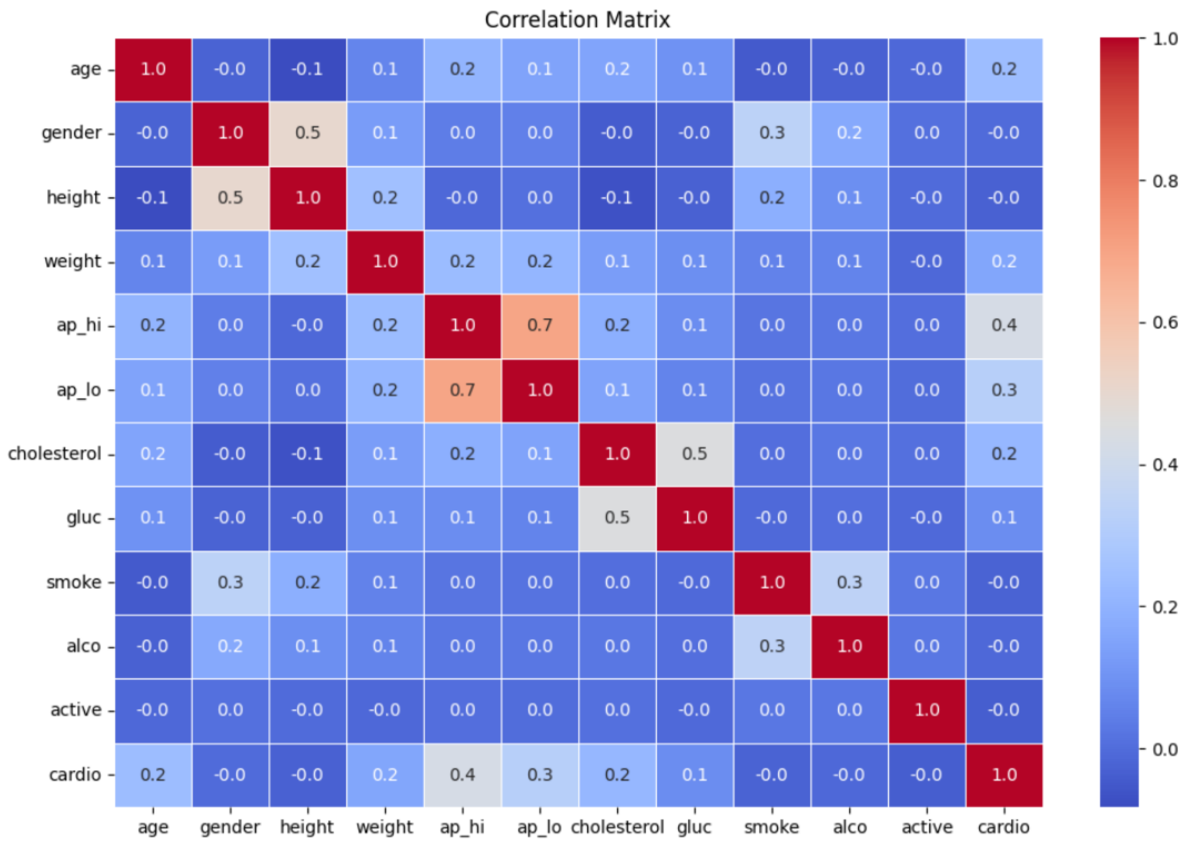


Fig 8: Correlation matrix among the features

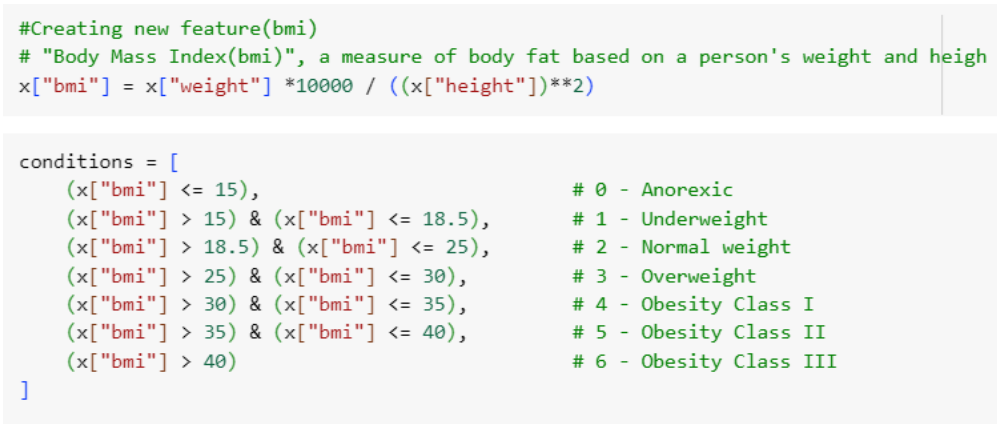
We visualized the importance of features in our dataset using a correlation matrix, which quantifies and illustrates the relationships between variables. Each cell in the matrix indicates the correlation coefficient between two variables, ranging from -1 to 1. A positive correlation (close to 1) means that as one variable increases, the other tends to increase, while a negative correlation (close to -1) indicates that as one variable increases, the other tends to decrease. In our correlation matrix, we observed a strong positive correlation (0.7) between 'ap_hi' (systolic blood pressure) and 'ap_lo' (diastolic blood pressure).

Furthermore, 'cholesterol' and 'glucose' levels exhibited a positive correlation of 0.5, and a similar correlation was observed between 'gender' and 'height'. These findings from the correlation matrix provide valuable insights into the interconnectedness of variables, aiding in the identification of significant relationships and informing subsequent steps in our data analysis and modeling processes.

#Feature Engineering (Feature Extraction):

Feature extraction involves selecting or transforming a dataset's input features to create a more effective and informative representation for building predictive models. In our initial step, we divided our dataset into an independent variable (x) and a dependent variable or target variable (y), with 'cardio' as our target. The remaining 11 features served as independent variables. Subsequently, we enhanced the dataset by introducing four new independent features:

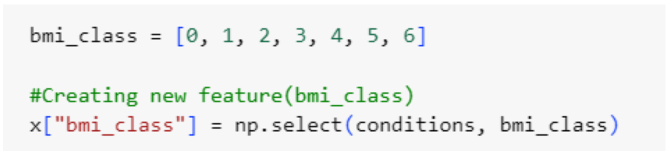
- ‘bmi’ (Body Mass Index), a measure of body fat based on weight and height.
- 'bmi_class' to categorize BMI;
- 'bp_cat' to categorize blood pressure; and
- 'pulse_press,' representing the difference between systolic and diastolic blood pressure.



```
#Creating new feature(bmi)
# "Body Mass Index(bmi)", a measure of body fat based on a person's weight and heigh
x["bmi"] = x["weight"] *10000 / ((x["height"])**2)
```

```
conditions = [
    (x["bmi"] <= 15),                          # 0 - Anorexic
    (x["bmi"] > 15) & (x["bmi"] <= 18.5),      # 1 - Underweight
    (x["bmi"] > 18.5) & (x["bmi"] <= 25),      # 2 - Normal weight
    (x["bmi"] > 25) & (x["bmi"] <= 30),        # 3 - Overweight
    (x["bmi"] > 30) & (x["bmi"] <= 35),        # 4 - Obesity Class I
    (x["bmi"] > 35) & (x["bmi"] <= 40),        # 5 - Obesity Class II
    (x["bmi"] > 40)                            # 6 - Obesity Class III
]
```

Fig 9: Screenshot of creating new feature (bmi)

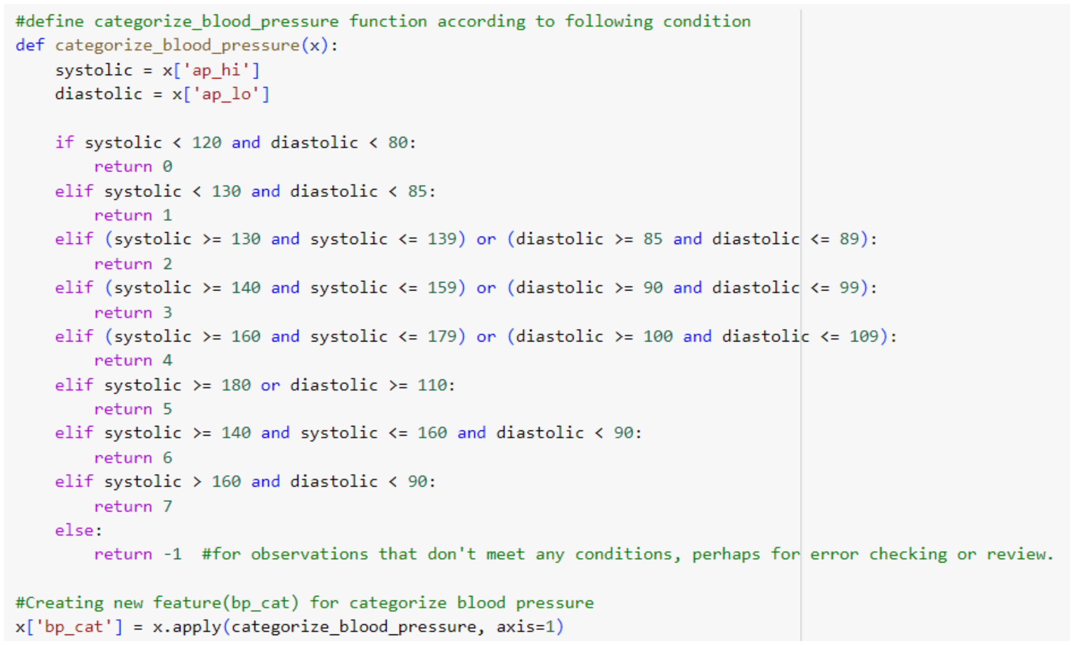


```
bmi_class = [0, 1, 2, 3, 4, 5, 6]

#Creating new feature(bmi_class)
x["bmi_class"] = np.select(conditions, bmi_class)
```

Fig 10: Screenshot of creating new feature (bmi_class)

The first addition is the "Body Mass Index (BMI)," calculated as the ratio of an individual's weight to the square of their height. Subsequently, we categorized BMI into seven classes, ranging from Anorexic to Obesity Class III, providing a comprehensive representation of body weight status.

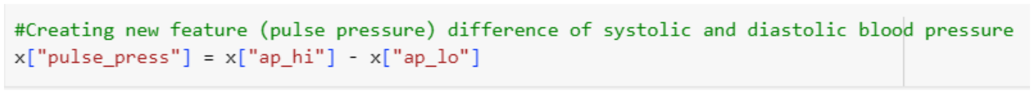


```python
#define categorize_blood_pressure function according to following condition
def categorize_blood_pressure(x):
    systolic = x['ap_hi']
    diastolic = x['ap_lo']

    if systolic < 120 and diastolic < 80:
        return 0
    elif systolic < 130 and diastolic < 85:
        return 1
    elif (systolic >= 130 and systolic <= 139) or (diastolic >= 85 and diastolic <= 89):
        return 2
    elif (systolic >= 140 and systolic <= 159) or (diastolic >= 90 and diastolic <= 99):
        return 3
    elif (systolic >= 160 and systolic <= 179) or (diastolic >= 100 and diastolic <= 109):
        return 4
    elif systolic >= 180 or diastolic >= 110:
        return 5
    elif systolic >= 140 and systolic <= 160 and diastolic < 90:
        return 6
    elif systolic > 160 and diastolic < 90:
        return 7
    else:
        return -1  #for observations that don't meet any conditions, perhaps for error checking or review.

#Creating new feature(bp_cat) for categorize blood pressure
x['bp_cat'] = x.apply(categorize_blood_pressure, axis=1)
```

Fig 11: Screenshot of creating new feature (bp_cat)

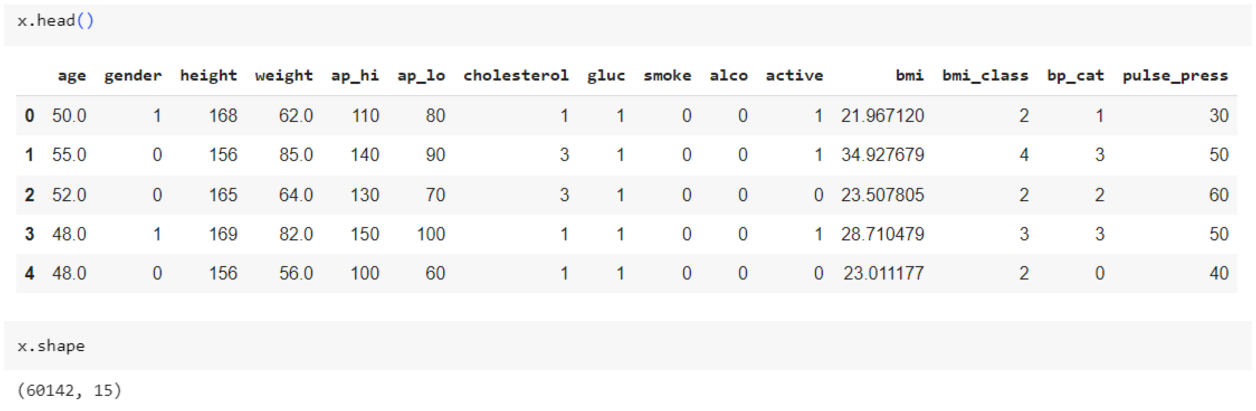


```python
#Creating new feature (pulse pressure) difference of systolic and diastolic blood pressure
x["pulse_press"] = x["ap_hi"] - x["ap_lo"]
```

Fig 12: Screenshot of creating new feature (pulse_press)

Additionally, we introduced a function to categorize blood pressure ('bp_cat') based on predefined conditions, incorporating both systolic and diastolic values. The categories range from normal blood pressure to severe hypertension. Finally, we computed the "pulse pressure," representing the difference between systolic and diastolic blood pressure. This strategic feature creation aimed to enrich the dataset with additional information, ultimately enhancing its utility for predictive modeling.

Now, our dataset has 15 independent features including new features.



```python
x.head()
```

	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active	bmi	bmi_class	bp_cat	pulse_press
0	50.0	1	168	62.0	110	80	1	1	0	0	1	21.967120	2	1	30
1	55.0	0	156	85.0	140	90	3	1	0	0	1	34.927679	4	3	50
2	52.0	0	165	64.0	130	70	3	1	0	0	0	23.507805	2	2	60
3	48.0	1	169	82.0	150	100	1	1	0	0	1	28.710479	3	3	50
4	48.0	0	156	56.0	100	60	1	1	0	0	0	23.011177	2	0	40

```python
x.shape
```

```
(60142, 15)
```

Fig 13: The shape of dataset after creating new features.

# Feature Engineering (dimensionality reduction):

Now, our dataset comprised 15 independent features. To implement dimensionality reduction, we applied Principal Component Analysis (PCA) after standardizing the data, ensuring uniform scales across features. This standardization is crucial for PCA, which relies on the covariance structure and can be influenced by varying scales. The main goal of PCA was to transform the dataset from 15 dimensions to a more efficient 2-dimensional representation while preserving essential information from the original features. The resulting two principal components, representing linear combinations of the original features, capture the maximum variance directions in the data. These components are orthogonal, and their corresponding eigenvalues indicate the explained variance. By selecting the top two components, we retained a substantial proportion of overall variance, significantly reducing dimensionality. This reduction offers advantages such as faster computation and training, aiding visualization, and interpretation. The 2-dimensional dataset maintains the essence of the original features, providing a more manageable input for subsequent modeling or analysis. The reduced-dimension representation via PCA enhances efficiency, interpretability, and overall performance of machine learning models on our dataset.

```
Text(0, 0.5, 'Second principle component')
```

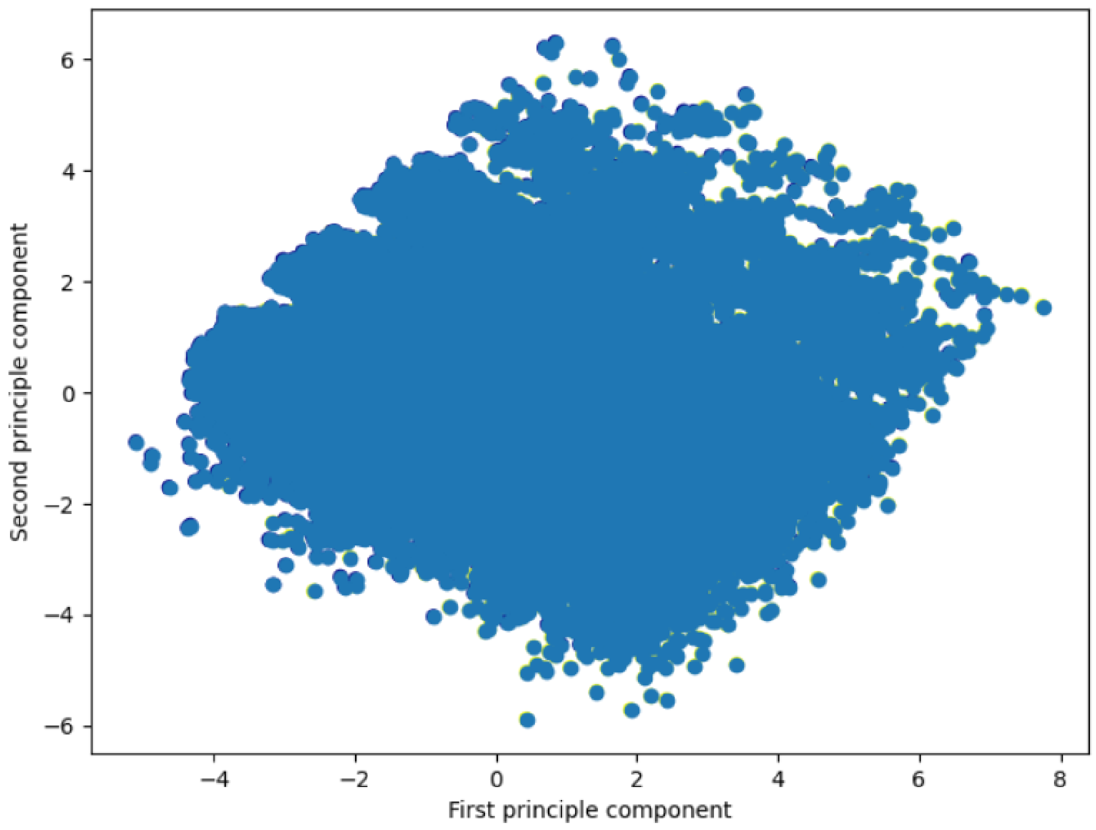
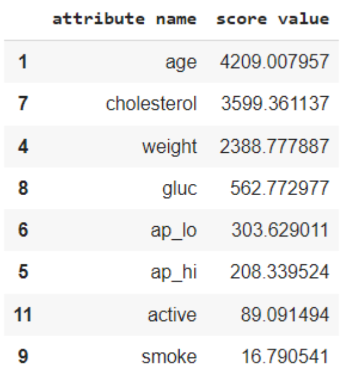


Fig 14: Dimensionality reduction using PCA.

Technique Used

After dimensionality reduction, we proceeded to split our dataset into training data (80%) and test data (20%). We assigned the variables xtrain and ytrain for training, and xtest and ytest for testing purposes. Subsequently, we applied several supervised classifiers, including RandomForest, DecisionTree, Support Vector Machine (SVM), and K-Nearest Neighbors (KNN). Notably, the highest accuracy of 71% was achieved by the Support Vector Machine Classifier, while the other classifiers exhibited varying accuracies, with RandomForest at 66%, DecisionTree at 62%, and KNN at 68%.

To further refine our model, we incorporated a feature selection technique. After evaluating the feature importance scores, we identified the top 8 features among the initial 12 independent features. Subsequently, we removed the less influential 4 features (id, gender, height, alco) from our dataset.



	attribute name	score value
1	age	4209.007957
7	cholesterol	3599.361137
4	weight	2388.777887
8	gluc	562.772977
6	ap_lo	303.629011
5	ap_hi	208.339524
11	active	89.091494
9	smoke	16.790541

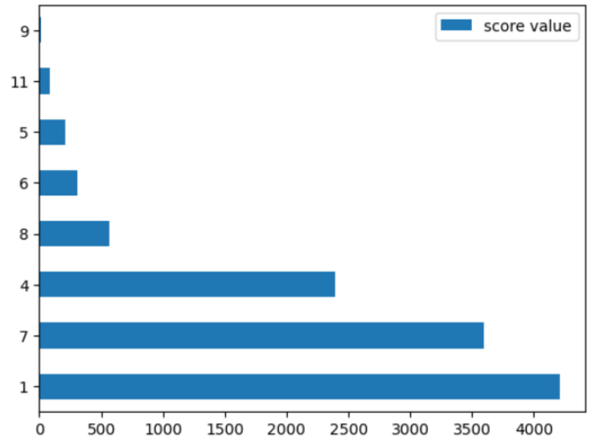


Fig 15: Top features and bar chart using "Feature Selection Technique".

Following this feature selection, we once again split our dataset into training and test sets, applying the same classifier algorithms. Surprisingly, the results shifted after feature selection. The KNN classifier now demonstrated the highest accuracy of 69%, tied with RandomForest. However, other classifiers showed different performance metrics: DecisionTree accuracy increased to 63%, and Support Vector Machine accuracy notably dropped to 50%, marking the lowest accuracy among the classifiers. This outcome underscores the impact of feature selection on model performance.

Classifier	Accuracy (after using PCA)	Accuracy (after using Feature Selection)
Decision Tree	62%	63%
Random Forest	66%	69%
SVM	71%	50%
KNN	68%	69%

Table: Accuracy of the various classifier

The choice of a specific model depends on various factors, such as the nature of the data, the characteristics of the features, and the desired outcome. The reasons for selecting these models and how features were treated in the modeling process are describing below:

RandomForest: RandomForest is an ensemble learning method that builds multiple decision trees and merges their predictions. It is robust, handles complex relationships well, and is less prone to overfitting. It is particularly effective when dealing with datasets that have a mix of categorical and numerical features, as it can handle both types effectively. It's also robust to outliers and irrelevant features, making it suitable for datasets with diverse characteristics. RandomForest inherently assesses feature importance during the training process, allowing for the identification of influential features. This information was later used in the feature selection step to refine the model.

DecisionTree: Decision trees are simple, interpretable models that can capture non-linear relationships in the data. They're useful for understanding feature importance. They are suitable when the relationships within the data are not strictly linear. They can capture complex patterns and interactions between features. They inherently consider feature importance when making splits. However, in the feature selection step, less informative features were identified and removed to refine the model.

Support Vector Machine (SVM): SVM is effective in high-dimensional spaces, making it suitable for datasets with many features. It's also powerful in capturing complex relationships between features. It is well-suited for datasets with a large number of features and can handle non-linear relationships effectively through the use of kernel functions. SVM performance was impacted by feature selection, indicating that certain features were crucial for its accuracy. After feature selection, SVM accuracy notably dropped, suggesting that the removed features might have contained valuable information for SVM's decision boundaries.

K-Nearest Neighbors (KNN): KNN is a simple and effective algorithm that relies on the similarity of data points. It's useful when local patterns in the data are important. It can perform well when the underlying patterns in the data exhibit local similarities. It's sensitive to the scale of features, so standardizing the data is crucial. It initially demonstrated competitive accuracy. After feature selection, its accuracy improved, indicating that the removal of certain features contributed positively to its performance.

Feature Treatment in the Modeling Process: The features in the dataset underwent several treatments to enhance the modeling process. Initially, exploratory data analysis (EDA) was conducted to understand the characteristics of each feature, including visualizing key statistics and patterns. The 'gender' feature was label-encoded to convert categorical values into numerical ones, simplifying the representation for machine learning algorithms. Feature engineering was a crucial step where new features were introduced to enrich the dataset. 'BMI' (Body Mass Index), 'bmi_class' (BMI categories), 'bp_cat' (blood pressure categories), and 'pulse_press' (difference between systolic and diastolic blood pressure) were added to provide additional information for predictive modeling. Subsequently, dimensionality reduction through PCA was applied to reduce the dataset from 15 dimensions to 2 dimensions while preserving essential information. This reduction aimed to enhance efficiency, interpretability, and overall model performance. The resulting two principal components retained a substantial proportion of overall variance, making the dataset more manageable for subsequent analysis. After applying supervised classifiers, including SVM, RandomForest, DecisionTree, and KNN, the SVM classifier stood out with the highest accuracy of 71%. However, a feature selection step was introduced to further refine the model. Surprisingly, the SVM classifier's accuracy notably dropped to 50% after feature selection, while KNN and RandomForest showed the highest accuracy of 69%. This outcome emphasizes the impact of feature selection on model performance and highlights the iterative nature of model development. The choice of the SVM classifier was initially based on its performance on the entire set of features, but the efficacy changed after feature selection. This illustrates the importance of experimenting with different combinations of features and algorithms to optimize predictive accuracy in the context of your specific dataset.

Evaluation Metric

In evaluating machine learning classifiers for the cardiovascular disease prediction task, we employed key metrics such as the confusion matrix and classification report. These metrics offer a comprehensive insight into the model's predictive performance, encompassing accuracy, precision, recall, F1-score, and support. The Support Vector Machine (SVM) demonstrated the highest accuracy score at 71%, leading us to consider it as the preferred model for the task.

Confusion matrix: The confusion matrix provided is a 2x2 matrix that summarizes the performance of a classification model on a set of test data. Each row of the matrix represents the instances in an actual class, while each column represents the instances in a predicted class.

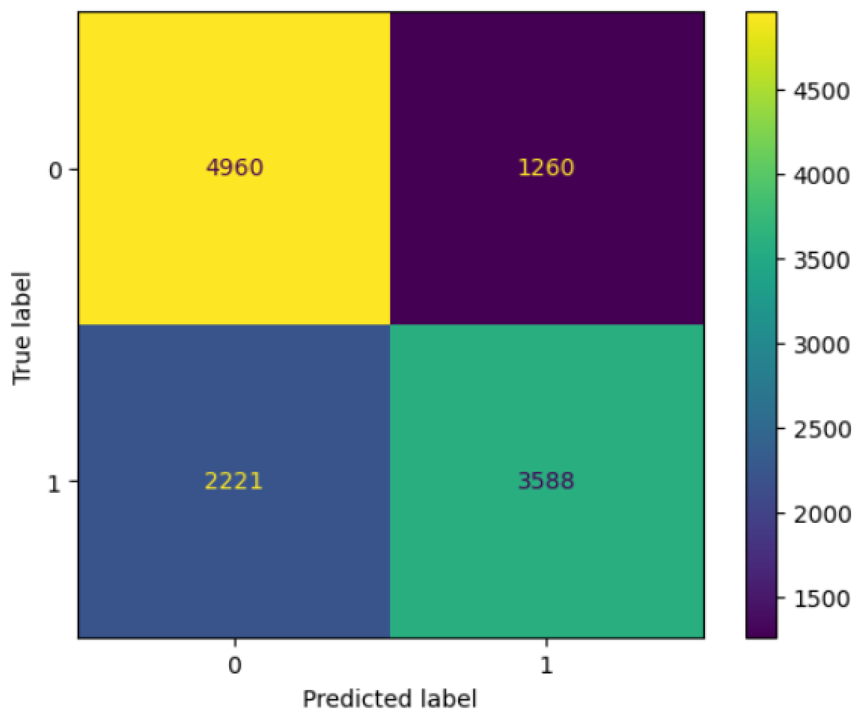


Fig 16: Confusion Matrix for Support Vector Machine

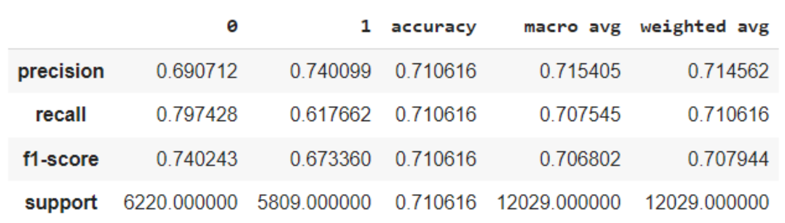
The matrix is as follows:

- True Positive (TP): 3588, These are cases where the model correctly predicted the presence of cardiovascular disease (CVD).
- False Positive (FP): 1260, These are cases where the model incorrectly predicted the presence of CVD. In other words, the model predicted positive, but the actual condition is negative.
- True Negative (TN): 4960, These are cases where the model correctly predicted the absence of cardiovascular disease.
- False Negative (FN): 2221, These are cases where the model incorrectly predicted the absence of CVD. In other words, the model predicted negative, but the actual condition is positive.

Classification Report: The values of accuracy, precision, recall, and F1- score were calculated as follows.

Accuracy = (TN + TP) /(TN + TP + FN + FP) × 100%
Precision = *TP /(TP + FP)*× 100%
Recall = *TP/(TP + FN)*× 100%
F1-score = 2(Precision*Recall)/(Precision + Recall) ×100%

Here, TP = True Positive, TN = True Negative, FP = False Positive, and FN = False Negative.



	0	1	accuracy	macro avg	weighted avg
precision	0.690712	0.740099	0.710616	0.715405	0.714562
recall	0.797428	0.617662	0.710616	0.707545	0.710616
f1-score	0.740243	0.673360	0.710616	0.706802	0.707944
support	6220.000000	5809.000000	0.710616	12029.000000	12029.000000

```
<Axes: >
```

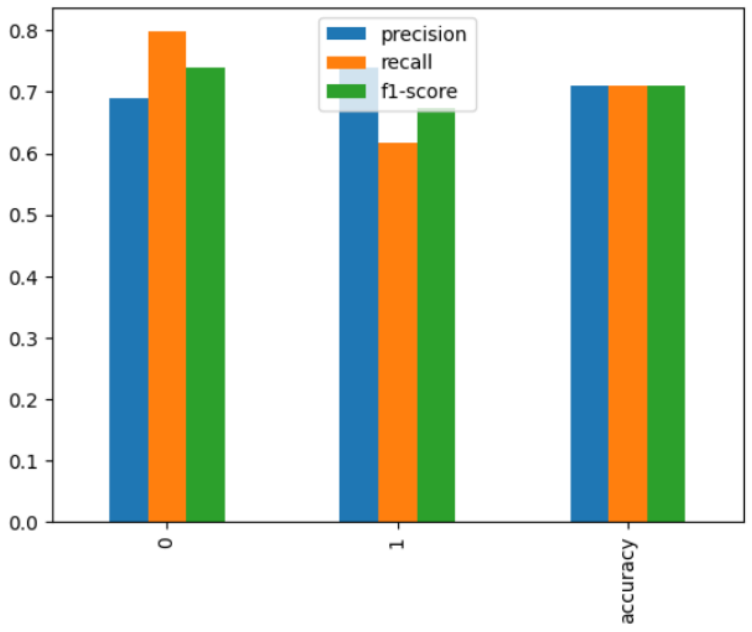


Fig 17: Classification Report

From Fig -17, we demonstrated the classification report as follows:

- Precision: Precision is the ratio of correctly predicted positive observations to the total predicted positives. For class 0 (no CVD), precision is 0.691, and for class 1 (CVD), precision is 0.740. This indicates the proportion of correctly predicted instances for each class among all instances predicted as belonging to that class.
- Recall (Sensitivity): Recall is the ratio of correctly predicted positive observations to the total actual positives. For class 0, recall is 0.797, and for class 1, recall is 0.618. This measures the ability of the model to capture all the relevant instances of each class.
- F1-Score: The F1-Score is the harmonic mean of precision and recall. It provides a balance between precision and recall. For class 0, F1-Score is 0.740, and for class 1, F1-Score is 0.673.
- Support: The number of actual occurrences of each class in the specified dataset. For class 0, there are 6220 instances, and for class 1, there are 5809 instances.
- Accuracy: Overall accuracy of the model is 0.711, indicating the proportion of correctly predicted instances (both true positives and true negatives) among all instances.
- Macro Avg: The macro average calculates the average performance across all classes. The macro average precision, recall, and F1-Score are 0.715, 0.708, and 0.707, respectively.
- Weighted Avg: The weighted average considers the contribution of each class based on its representation in the dataset. The weighted average precision, recall, and F1-Score are 0.715, 0.711, and 0.708, respectively.

Why These Metrics:

- Precision: It is crucial when the cost of false positives is high. In the context of healthcare, high precision means that when the model predicts the presence of cardiovascular disease, it is often correct.
- Recall: It is important when the cost of false negatives is high. In healthcare, a high recall indicates that the model is good at capturing instances of cardiovascular disease, minimizing false negatives.
- F1-score: It provides a balance between precision and recall. It is especially useful when there is an uneven class distribution.
- Support: It gives the number of occurrences of each class, providing context for the performance metrics.

Parameters of the Technique (Support Vector Machine - SVM): SVM is a powerful classifier that works well for both linear and non-linear relationships in data. The accuracy achieved by SVM is reported to be 71%. The specific parameters of the SVM model, such as the choice of kernel, regularization parameters (C), and kernel-specific parameters. These parameters are essential for fine-tuning the model's performance.

Learning in Course project

In this project, we learned the importance of thorough exploratory data analysis (EDA) to understand the characteristics of the dataset comprehensively. The initial examination ensured data completeness, absence of duplicates, and the identification of potential areas for preprocessing. We leveraged label encoding for categorical variables, such as 'gender,' and conducted feature engineering to introduce informative variables like BMI, blood pressure categories, and pulse pressure. Dimensionality reduction using Principal Component Analysis (PCA) was applied to enhance efficiency and interpretability. Model evaluation revealed the Support Vector Machine (SVM) as the most accurate classifier at 71%, highlighting the significance of feature selection. The confusion matrix and classification report provided a detailed assessment of the model's performance, emphasizing precision, recall, and F1-score in the context of cardiovascular disease prediction. Notably, the iterative nature of model development was underscored, demonstrating the impact of feature selection on classifier performance and reinforcing the need for experimentation to optimize predictive accuracy. Overall, this project showcased the importance of a systematic approach, from data exploration and preprocessing to model development and evaluation, in building an effective predictive model for cardiovascular disease.