A blue and white logo

Description automatically generated

*Ain Shams University - Faculty of Engineering*

**CSE381 (UG2018) - Introduction to Machine Learning – Fall 24**

**Project**

Mohamed Omar Adel ElBialy

21P0068

Abdelrahman Sherif Abdelaziz

21P0098

Ezz Eldin Alaa Eldin Shalaby

21P0100

Table of Contents

[Milestone 1 3](#_Toc183960076)

[1.Introduction 3](#_Toc183960077)

[2.Libraries 3](#_Toc183960078)

[3.Data preprocessing and cleaning 4](#_Toc183960079)

[3.1 Introduction 4](#_Toc183960080)

[3.2 Process 4](#_Toc183960081)

[3.3 PCA 10](#_Toc183960082)

# Milestone 1

This milestone focuses on the key steps of the machine learning workflow, including Data Exploration and Visualization, Data Cleaning and Processing, as well as training and testing classifiers. Specifically, we will implement Naïve Bayes and Support Vector Machine (SVM) models to predict heart disease outcomes.

## 1.Introduction

Heart disease remains one of the leading causes of morbidity and mortality worldwide, necessitating the development of efficient diagnostic tools to aid in early detection and treatment. In this project, we leverage machine learning (ML) techniques to build a predictive model for identifying heart diseases based on patient data. The goal is to enhance diagnostic accuracy and provide timely interventions, ultimately improving patient outcomes. By utilizing various ML algorithms, this model aims to analyze patterns in health data, offering a reliable and scalable solution to support healthcare professionals in diagnosing and managing heart conditions effectively.

## 2.Libraries

These are the libraries to be used in **Milestone 1:**

A black screen with white text

Description automatically generated

In this project, several essential Python libraries are imported to support data analysis, preprocessing, and model development. **Pandas** is used for data manipulation and analysis, allowing us to load, clean, and preprocess the dataset efficiently. **NumPy** is employed for handling numerical operations and performing mathematical computations. **Matplotlib** and **Seaborn** are utilized for visualizing data and insights through various plots, helping us to understand patterns and distributions in the data. **Scikit-learn** is integral to the project, providing tools like **train\_test\_split** for splitting the data into training and testing sets, **PCA** (Principal Component Analysis) for dimensionality reduction, and multiple scaling techniques such as **StandardScaler** and **MinMaxScaler** to standardize or normalize the features. Lastly, **Scikit-learn** also offers various evaluation metrics to assess the performance of our machine learning models.

## 3.Data preprocessing and cleaning

Kindly note that PCA is done but after data preprocessing.

### 3.1 Introduction

Data preprocessing and cleaning is a critical step in any machine learning pipeline, particularly in healthcare applications where data can be noisy, incomplete, and imbalanced. In this project, we focus on preparing the raw healthcare data for analysis by handling missing values, normalizing features, encoding categorical variables, and addressing class imbalances. These preprocessing steps are essential for ensuring that the model can effectively learn patterns and make accurate predictions related to heart disease diagnosis. Proper preprocessing improves the model's performance and reliability, ultimately leading to better predictive outcomes.

### 3.2 Process

Most of the data analysis steps and procedures are thoroughly explained in the notebook; however, we will briefly review them here to eliminate any ambiguity.

We started up by visualizing the data in form of rows and columns, to get more sense on what we are working on with; get familiar with the data. It seems that we have 918 rows of data, uncleaned and ready to be processed.

We will check for duplicates or missing values in rows, apparently, we don’t have any:

A screenshot of a computer program

Description automatically generated

#### 3.2.1 Data encoding

Before moving on and checking for outliers, we must **encode** all the string/text values in our dataset, to get a better visualization of the data.

We will be using label encoding in this part.

A screenshot of a computer

Description automatically generated

Dataset before encoding

In the notebook, there is more explanation of what each number in the encoding represents and its past value.

A screenshot of a computer screen

Description automatically generated

Dataset after encoding

As we see, now we have a “numerical” dataset now which can be easily manipulated and visualized.

#### 3.2.2 Outliers

Now to identify outliers, we will be using the interquartile method, where we first determine the First Quartile (Q1) and Third Quartile (Q3). Q1 represents the 25th percentile, which means that 25% of the data lies below this value, while Q3 represents the 75th percentile, meaning 75% of the data lies below it. These quartiles are computed by finding the median of the lower half of the data for Q1 and the median of the upper half of the data for Q3. Once these quartiles are computed, we can calculate the Interquartile Range (IQR), which is the difference between Q3 and Q1, and represents the range where the middle 50% of the data lies. The next step is to determine the outlier boundaries, which are calculated by applying a factor of 1.5 times the IQR to both the lower and upper quartiles. Specifically, the lower boundary is found by subtracting 1.5 times the IQR from Q1, while the upper boundary is determined by adding 1.5 times the IQR to Q3. Any data point that falls outside of these boundaries, either below the lower bound or above the upper bound, is considered an outlier. Once outliers are identified, they can be removed from the dataset to ensure a more accurate analysis or model-building process.

We will firstly be visualizing the data with boxplots, pie charts and histograms. Boxplots are very helpful in visualizing the IQRs. Note that the pie charts aren’t that helpful in presenting features with lots of continues values in them, for example:

A pie chart with numbers and numbers

Description automatically generated

Below is an example for the visualization of the pie charts:

A pie chart of sex

Description automatically generatedA pie chart of pain type

Description automatically generatedA pie chart with numbers and a number of percentages

Description automatically generated

Below is an example for the visualization of the histograms:

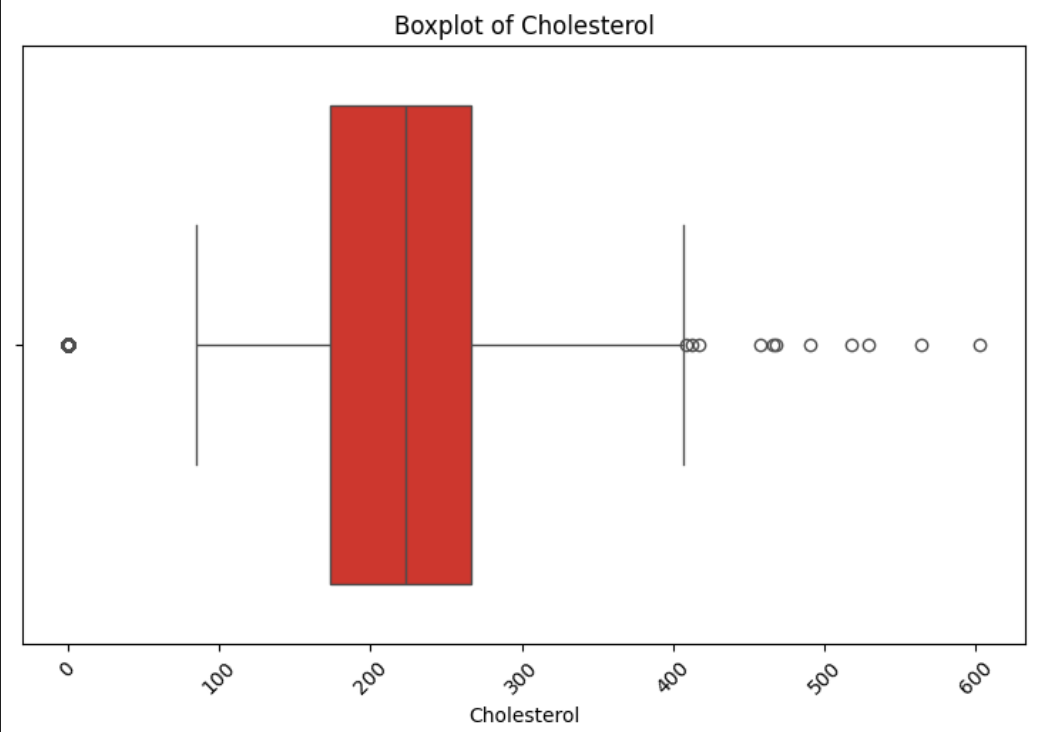
A graph with red lines

Description automatically generatedA diagram of a distribution of cholesterol

Description automatically generated

Note that in a discrete data set, histograms are of no great use.

Below is an example for the visualization of boxplots:

A graph with a red rectangular object

Description automatically generatedA graph with a red rectangle

Description automatically generated

After further analysis in the visualization of the data, we concluded that we had 4 features with possible outliers in them.

1. “RestingBp”

Because as we see here:

A graph of a box plot

Description automatically generated

There’s plenty of outliers outside Q1 and Q3.

1. “Cholesterol”

A diagram of a distribution of cholesterol

Description automatically generated

Same goes here + we have about 170 data rows with 0 cholesterol, which is nonsense.

1. “Maxhr”

A red rectangular object with numbers

Description automatically generated

1. “Old peak”

A graph of a function

Description automatically generated with medium confidence

Now the new data set with no outliers:

A screenshot of a computer screen

Description automatically generated

We now have 702 data rows.

Notice that we dropped 216 rows of data, most of them were the 0 cholesterol rows, and the rest are simple outliers.

In the notebook, you’ll find the new cleaned visualization of the data.

### 3.3 PCA