FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERIG
Department of Computer Engineering

Experiment 1 - Collecting, Cleaning and Transforming Healthcare Data for a Specific Disease

1. Course Details:

Academic Year	2023 - 24	Estimated Time	Experiment No. 1 – 02 Hours
Course & Semester	B.E. – Sem. VII	Subject Name	Data Science for Health and Social Care Lab
Experiment Type	Software Performance	Subject Code	HDSSBL701

Name of Student	Atharva Prashant Pawar	**Roll No.**	9427
Date of Performance.:		**Date of Submission.:**	
CO Mapping	HDSSBL701.1 Identify sources of data and methods for collecting, sharing and analyzing Healthcare data.		

Aim: Collecting, Cleaning, Integrating, and Transforming Healthcare Data for a Specific Disease:

Objective: The objective of this lab experiment is to familiarize students with the process of collecting, cleaning, integrating, and transforming healthcare data related to a specific disease. Students will gain hands-on experience in working with real-world healthcare datasets and preparing the data for analysis and AI applications.

Materials:

- Data analysis software (e.g., Python, R, or any preferred tool)
- Healthcare dataset(s) related to the chosen disease (e.g., public datasets, research datasets, or simulated data)

Procedure:

1. Choose a Specific Disease: Select a specific disease as the focus of the lab experiment. Consider diseases that have publicly available datasets or research data that can be accessed for analysis.

   Examples of diseases could include diabetes, cardiovascular disease, cancer, respiratory disorders, etc.

2. Data Collection: Identify and collect relevant healthcare data related to the chosen disease. Explore public data repositories, research databases, or other reliable sources to gather datasets that contain patient information, medical records, lab results, diagnostic codes, treatment data, and any other relevant variables. Ensure compliance with ethical guidelines and data protection regulations.

3. Data Cleaning: Clean the collected data to ensure its quality and reliability. This process may involve handling missing values, removing duplicates, standardizing formats, correcting errors, and addressing other data quality issues. Document the steps taken during the cleaning process.

4. Data Integration: Integrate multiple datasets if available or necessary. This step involves combining data from different sources that share common variables or patient identifiers. Apply appropriate techniques to merge the datasets while maintaining data integrity and ensuring consistent representations.

5. Data Transformation: Transform the integrated data into a suitable format for analysis and AI applications. This may involve feature engineering, scaling, normalization, encoding categorical variables, and creating derived variables. Document the transformations applied and their rationale.

6. Exploratory Data Analysis (EDA): Perform exploratory data analysis to gain insights into the dataset and the relationships between variables. Use visualizations, statistical summaries, and other techniques to understand the distribution of data, identify patterns, and uncover any interesting findings.

7. Summary and Documentation: Summarize the entire data preparation process, including data collection, cleaning, integration, and transformation. Document the steps taken, the challenges encountered, and the decisions made during each stage. Include any observations or insights gained from the exploratory data analysis.

Optional: Predictive Modeling: As an extension to the lab experiment, students can apply predictive modeling techniques using the prepared dataset. This can involve training machine learning models to predict disease outcomes, identify risk factors, or estimate treatment effectiveness. Students can evaluate the performance of the models and interpret their results.

Note: It is important to adhere to ethical guidelines and ensure the privacy and confidentiality of patient data throughout the lab experiment. Use de-identified or simulated datasets whenever possible to avoid any privacy concerns.

Data Repositories and Platforms: There are various data repositories and platforms where researchers and organizations share datasets. Some popular ones include:

- Kaggle
- UCI Machine Learning Repository
- Data.gov
- NIH National Library of Medicine

Result:

Dataset : heart_disease_data

Team: Atharva Pawar (9427), Aditya, Harsh

Data Science : Exp-1

In [27]:
```
# !pip install seaborn
```

In [28]:
```
# dataset link:
#     https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset
```

Importing the Dependencies

In [29]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import seaborn as sns
```

Data Collection and Processing

In [30]:
```
# loading the csv data to a Pandas DataFrame
heart_data = pd.read_csv('heart2.csv')
```

In [31]:
```
heart_data.shape
```

Out[31]:
```
(1025, 14)
```

In [3]:
```
print(heart_data)
```

```
      age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  \
0      52    1   0       125   212    0        1      168      0      1.0
1      53    1   0       140   203    1        0      155      1      3.1
2      70    1   0       145   174    0        1      125      1      2.6
3      61    1   0       148   203    0        1      161      0      0.0
4      62    0   0       138   294    1        1      106      0      1.9
...   ...  ...  ..       ...   ...  ...      ...      ...    ...      ...
1020   59    1   1       140   221    0        1      164      1      0.0
1021   60    1   0       125   258    0        0      141      1      2.8
1022   47    1   0       110   275    0        0      118      1      1.0
1023   50    0   0       110   254    0        0      159      0      0.0
1024   54    1   0       120   188    0        1      113      0      1.4

      slope  ca  thal  target
0         2   2     3       0
1         0   0     3       0
2         0   0     3       0
3         2   1     3       0
4         1   3     2       0
...     ...  ..   ...     ...
1020      2   0     2       1
1021      1   1     3       0
1022      1   1     2       0
1023      2   0     2       1
1024      1   1     3       0

[1025 rows x 14 columns]
```

In [4]:
```
# print first 10 rows of the dataset
heart_data.head(10)
```

Out[4]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
5	58	0	0	100	248	0	0	122	0	1.0	1	0	2	1
6	58	1	0	114	318	0	2	140	0	4.4	0	3	1	0
7	55	1	0	160	289	0	0	145	1	0.8	1	1	3	0
8	46	1	0	120	249	0	0	144	0	0.8	2	0	3	0
9	54	1	0	122	286	0	0	116	1	3.2	1	2	2	0

SEX: 0 = female 1 = male

In [5]:
```
# print last 10 rows of the dataset
heart_data.tail(10)
```

Out[5]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
1015	58	1	0	128	216	0	0	131	1	2.2	1	3	3	0
1016	65	1	3	138	282	1	0	174	0	1.4	1	1	2	0
1017	53	1	0	123	282	0	1	95	1	2.0	1	2	3	0
1018	41	1	0	110	172	0	0	158	0	0.0	2	0	3	0
1019	47	1	0	112	204	0	1	143	0	0.1	2	0	2	1
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

data info

age: Age of the patient in years (ranging from 29 to 77).

sex: Sex of the patient (0: female, 1: male).

cp: Chest pain type (0 to 3), indicating different levels of chest pain experienced by the patient.

trestbps: Resting blood pressure in mm Hg.

chol: Serum cholesterol level in mg/dL.

fbs: Fasting blood sugar level > 120 mg/dL (1: true, 0: false).

restecg: Resting electrocardiographic results (0 to 2), indicating different types of ECG results.

thalach: Maximum heart rate achieved during exercise.

exang: Exercise-induced angina (1: yes, 0: no).

oldpeak: ST depression induced by exercise relative to rest.

slope: Slope of the peak exercise ST segment (0 to 2).

ca: Number of major vessels (0 to 4) colored by fluoroscopy.

thal: Thalassemia (3 types - 1, 2, 3).

target: The target variable indicating the presence of heart disease (1: yes, 0: no).

In [6]:
```
# Explore the basic statistics of the dataset
print("Basic Statistics:")
print(heart_data.describe())
```

```
Basic Statistics:
               age          sex           cp     trestbps        chol  \
count  1025.000000  1025.000000  1025.000000  1025.000000  1025.00000
mean     54.434146     0.695610     0.942439   131.611707   246.00000
std       9.072290     0.460373     1.029641    17.516718    51.59251
min      29.000000     0.000000     0.000000    94.000000   126.00000
25%      48.000000     0.000000     0.000000   120.000000   211.00000
50%      56.000000     1.000000     1.000000   130.000000   240.00000
75%      61.000000     1.000000     2.000000   140.000000   275.00000
max      77.000000     1.000000     3.000000   200.000000   564.00000

               fbs      restecg      thalach        exang      oldpeak  \
count  1025.000000  1025.000000  1025.000000  1025.000000  1025.000000
mean      0.149268     0.529756   149.114146     0.336585     1.071512
std       0.356527     0.527878    23.005724     0.472772     1.175053
min       0.000000     0.000000    71.000000     0.000000     0.000000
25%       0.000000     0.000000   132.000000     0.000000     0.000000
50%       0.000000     1.000000   152.000000     0.000000     0.800000
75%       0.000000     1.000000   166.000000     1.000000     1.800000
max       1.000000     2.000000   202.000000     1.000000     6.200000

             slope           ca         thal       target
count  1025.000000  1025.000000  1025.000000  1025.000000
mean      1.385366     0.754146     2.323902     0.513171
std       0.617755     1.030798     0.620660     0.500070
min       0.000000     0.000000     0.000000     0.000000
25%       1.000000     0.000000     2.000000     0.000000
50%       1.000000     0.000000     2.000000     1.000000
75%       2.000000     1.000000     3.000000     1.000000
max       2.000000     4.000000     3.000000     1.000000
```

In [20]:
```
# statistical measures about the data
# heart_data.describe()
```

In [8]:
```
# Explore the distribution of target classes
print("\nTarget Class Distribution:")
print(heart_data['target'].value_counts())
```

```
Target Class Distribution:
1    526
0    499
Name: target, dtype: int64
```

1 --> Defective Heart

0 --> Healthy Heart

In [9]:
```
# number of rows and columns in the dataset
heart_data.shape
```

Out[9]: (1025, 14)

In [10]:
```
# getting some info about the data
heart_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1025 non-null   int64
 1   sex       1025 non-null   int64
 2   cp        1025 non-null   int64
 3   trestbps  1025 non-null   int64
 4   chol      1025 non-null   int64
 5   fbs       1025 non-null   int64
 6   restecg   1025 non-null   int64
 7   thalach   1025 non-null   int64
 8   exang     1025 non-null   int64
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64
 11  ca        1025 non-null   int64
 12  thal      1025 non-null   int64
 13  target    1025 non-null   int64
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

In [11]:
```
# checking for missing values
```

In [11]:
```
# checking for missing values
heart_data.isnull().sum()
```

Out[11]:
```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

In [12]:
```
# checking the distribution of Target Variable
heart_data['target'].value_counts()
```

Out[12]:
```
1    526
0    499
Name: target, dtype: int64
```

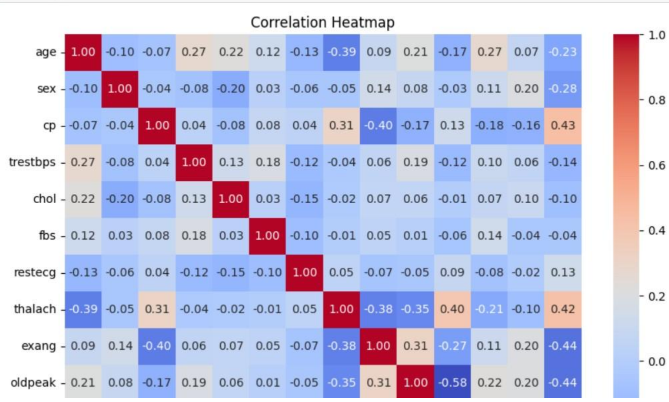
1 --> Defective Heart

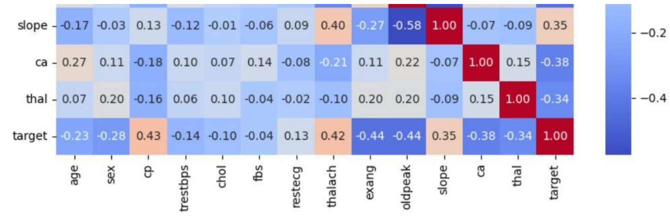
0 --> Healthy Heart

In [32]:
```
# Correlation matrix
print("\nCorrelation Matrix:")
correlation_matrix = heart_data.corr()
print(correlation_matrix)
```

```

Correlation Matrix:
               age       sex        cp  trestbps      chol       fbs  \
age       1.000000 -0.103240 -0.071966  0.271121  0.219823  0.121243
sex      -0.103240  1.000000 -0.041119 -0.078974 -0.198258  0.027200
cp       -0.071966 -0.041119  1.000000  0.038177 -0.081641  0.079294
trestbps  0.271121 -0.078974  0.038177  1.000000  0.127977  0.181767
chol      0.219823 -0.198258 -0.081641  0.127977  1.000000  0.026917
fbs       0.121243  0.027200  0.079294  0.181767  0.026917  1.000000
restecg  -0.132696 -0.055117  0.043581 -0.123794 -0.147410 -0.104051
thalach  -0.390227 -0.049365  0.306839 -0.039264 -0.021772 -0.008866
exang     0.088163  0.139157 -0.401513  0.061197  0.067382  0.049261
oldpeak   0.208137  0.084687 -0.174733  0.187434  0.064880  0.010859
slope    -0.169105 -0.026666  0.131633 -0.120445 -0.014248 -0.061902
ca        0.271551  0.111729 -0.176206  0.104554  0.074259  0.137156
thal      0.072297  0.198424 -0.163341  0.059276  0.100244 -0.042177
target   -0.229324 -0.279501  0.434854 -0.138772 -0.099966 -0.041164

           restecg   thalach     exang   oldpeak     slope        ca  \
age      -0.132696 -0.390227  0.088163  0.208137 -0.169105  0.271551
sex      -0.055117 -0.049365  0.139157  0.084687 -0.026666  0.111729
cp        0.043581  0.306839 -0.401513 -0.174733  0.131633 -0.176206
trestbps -0.123794 -0.039264  0.061197  0.187434 -0.120445  0.104554
chol     -0.147410 -0.021772  0.067382  0.064880 -0.014248  0.074259
fbs      -0.104051 -0.008866  0.049261  0.010859 -0.061902  0.137156
restecg   1.000000  0.048411 -0.065606 -0.050114  0.086086 -0.078072
thalach   0.048411  1.000000 -0.380281 -0.349796  0.395308 -0.207888
exang    -0.065606 -0.380281  1.000000  0.310844 -0.267335  0.107849
oldpeak  -0.050114 -0.349796  0.310844  1.000000 -0.575189  0.221816
slope     0.086086  0.395308 -0.267335 -0.575189  1.000000 -0.073440
ca       -0.078072 -0.207888  0.107849  0.221816 -0.073440  1.000000
thal     -0.020504 -0.098068  0.197201  0.202672 -0.094090  0.149014
target    0.134468  0.422895 -0.438029 -0.438441  0.345512 -0.382085

              thal    target
age       0.072297 -0.229324
sex       0.198424 -0.279501
cp       -0.163341  0.434854
trestbps  0.059276 -0.138772
chol      0.100244 -0.099966
fbs      -0.042177 -0.041164
restecg  -0.020504  0.134468
thalach  -0.098068  0.422895
exang     0.197201 -0.438029
oldpeak   0.202672 -0.438441
slope    -0.094090  0.345512
ca        0.149014 -0.382085
thal      1.000000 -0.337838
target   -0.337838  1.000000
```

In [33]:
```
# Heatmap of correlation matrix
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```




slope
ca
thal
target
age
sex
cp
trestbps
chol
fbs
restecg
thalach
exang
oldpeak
slope
ca
thal
target
−0.2
−0.4