

SPRINT 1-PART 2

Date	27 October 2022
Team ID	PNT2022TMID53213
Project Name	Visualizing and Predicting Heart Diseases with an Interactive Dash Board

1)Description:

->It tells about the dataset and their columns and values, and their variation in it.

->To help members of your organization quickly identify datasets that might be useful for them, provide a concise, informative description of your dataset in the dataset's settings. Users will see this description in the info tooltip next to the dataset's name in the datasets hub, as well as on the dataset's details page. Providing a meaningful description helps foster dataset reuse. For instance, based on a dataset's description, users may decide to explore reports that are based on the dataset, or to create their own reports based on the dataset.

The screenshot shows a Jupyter Notebook titled 'Sprint1_part2.ipynb' in a Google Colab environment. The notebook contains the following code and output:

```
import cufflinks as cf
import matplotlib.pyplot as plt
import seaborn as sns

df=pd.read_csv('Heart_Disease_Prediction.csv')
df.head()
```

The output of the code is a table with 15 columns: Age, Sex, Chest pain type, BP, Cholesterol, FBS over 120, EKG results, Max HR, Exercise angina, ST depression, Slope of ST, Number of vessels fluro, Thallium, and Heart Disease. The table displays the first 5 rows of data.

	Age	Sex	Chest pain type	BP	Cholesterol	FBS over 120	EKG results	Max HR	Exercise angina	ST depression	Slope of ST	Number of vessels fluro	Thallium	Heart Disease
0	70	1	4	130	322	0	2	109	0	2.4	2	3	3	Presence
1	67	0	3	115	564	0	2	160	0	1.6	2	0	7	Absence
2	57	1	2	124	261	0	0	141	0	0.3	1	0	7	Presence
3	64	1	4	128	263	0	0	105	1	0.2	2	1	7	Absence
4	74	0	2	120	269	0	2	121	1	0.2	1	1	3	Absence

Below the table, the notebook shows the following code and output:

```
info = ["age", "1: male, 0: female", "chest pain type, 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4: asymptomatic", "resting blood pressure", "serv"]

for i in range(len(info)):
    print(df.columns[i],":",info[i])
```

The output of the code is:

```
age: 1: male, 0: female
chest pain type: 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4: asymptomatic
resting blood pressure:
serv:
```

Inbox (120) - haribala X WhatsApp X Heart-Disease-Predic X Data Cleaning and Pre X Sprint1_part2.ipynb X Get Started: 3 Ways to X

colab.research.google.com/drive/1zq8EmprG08C6_W7glvM7brixIV_-l2B#scrollTo=KzDtrQPSZZkn

Sprint1_part2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
[6] info = ["age", "1: male, 0: female", "chest pain type, 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4: asymptomatic", "resting blood pressure", "serum cholesterol in mg/dl", "fasting blood sugar > 120 mg/dl", "resting electrocardiographic results (values 0,1,2)", "maximum heart rate achieved", "exercise induced angina", "oldpeak = ST depression induced by exercise relative to rest", "the slope of the peak exercise ST segment", "number of major vessels (0-3) colored by flourosopy", "thal: 3 = normal; 6 = fixed defect; 7 = reversible defect"]

for i in range(len(info)):
    print(df.columns[i]+":\t\t\t"+info[i])
```

Age: age
Sex: 1: male, 0: female
Chest pain type: chest pain type, 1: typical angina, 2: atypical angina, 3: non-anginal pain, 4: asymptomatic
BP: resting blood pressure
Cholesterol: serum cholesterol in mg/dl
FBS over 120: fasting blood sugar > 120 mg/dl
EKG results: resting electrocardiographic results (values 0,1,2)
Max HR: maximum heart rate achieved
Exercise angina: exercise induced angina
ST depression: oldpeak = ST depression induced by exercise relative to rest
Slope of ST: the slope of the peak exercise ST segment
Number of vessels fluoro: number of major vessels (0-3) colored by flourosopy
Thallium: thal: 3 = normal; 6 = fixed defect; 7 = reversible defect

```
[8] df.groupby('Heart Disease').size()
```

Heart Disease
Absence 150
Presence 120
dtype: int64

completed at 11:10 AM

Inbox (120) - haribala X WhatsApp X Heart-Disease-Predic X Data Cleaning and Pre X Sprint1_part2.ipynb X Get Started: 3 Ways to X

colab.research.google.com/drive/1zq8EmprG08C6_W7glvM7brixIV_-l2B#scrollTo=KzDtrQPSZZkn

Sprint1_part2.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
[10] df.groupby('Heart Disease').sum()
```

	Age	Sex	Chest pain type	BP	Cholesterol	FBS over 120	EKG results	Max HR	Exercise angina	ST depression	Slope of ST	Number of vessels fluoro	Thallium
Heart Disease													
Absence	7906	83	423	19330	36632	23	129	23750	23	93.4	210	43	568
Presence	6791	100	434	16133	30776	17	147	16663	66	190.1	218	138	700

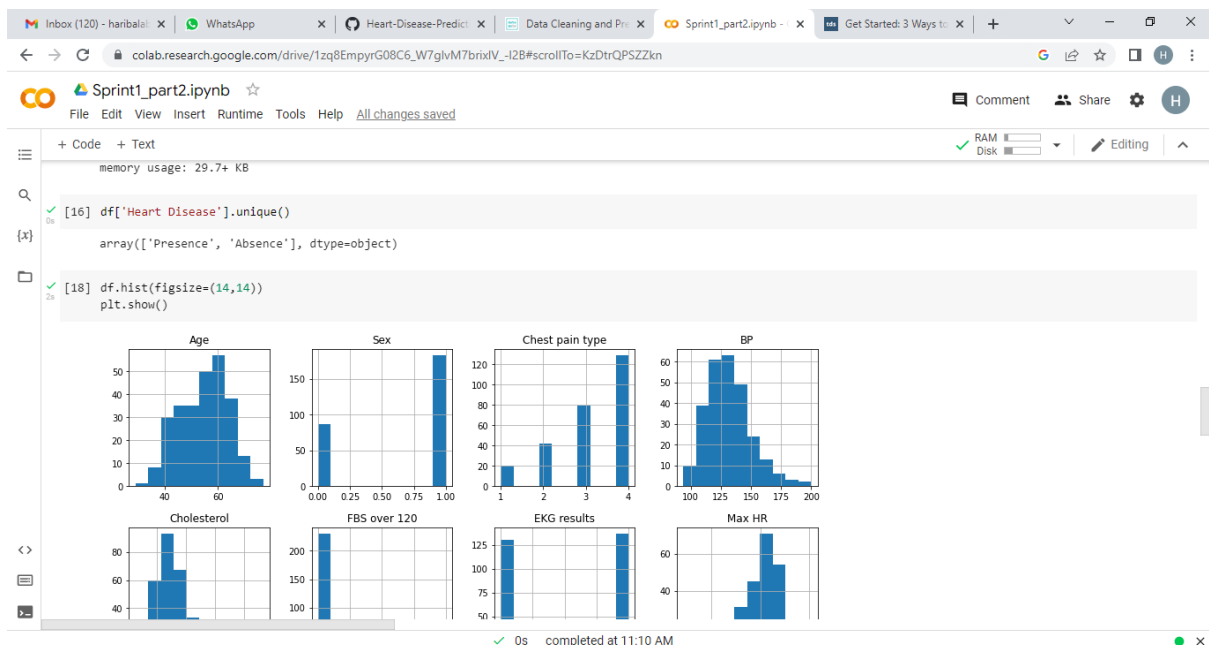
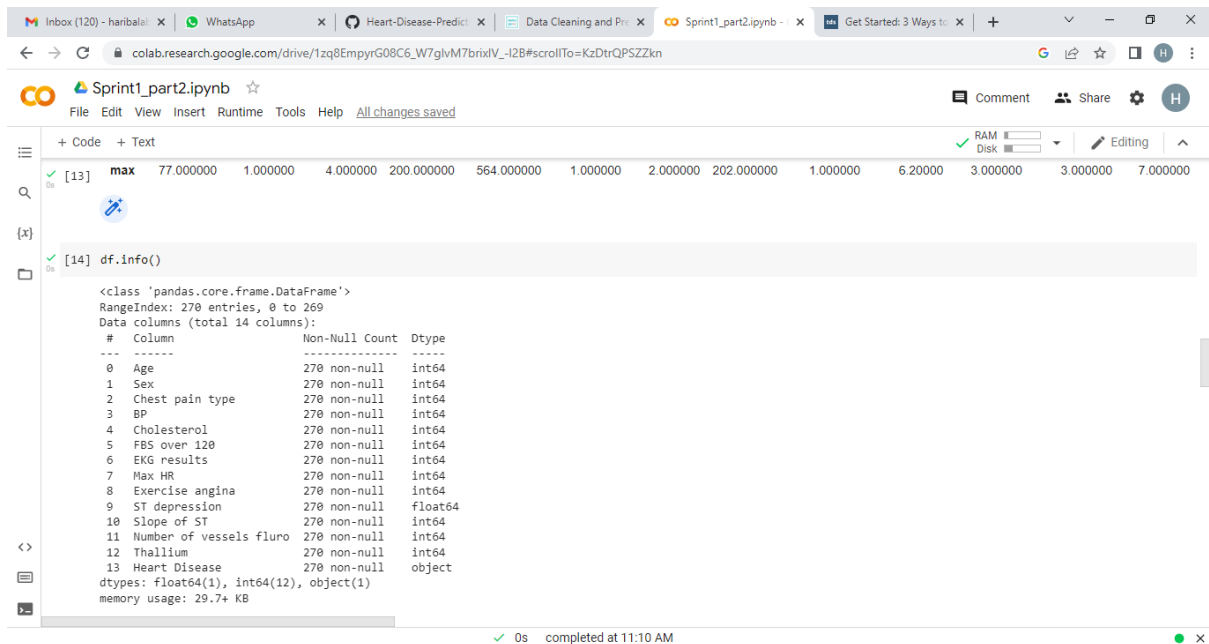
```
[12] df.shape
```

(270, 14)

```
[13] df.describe()
```

	Age	Sex	Chest pain type	BP	Cholesterol	FBS over 120	EKG results	Max HR	Exercise angina	ST depression	Slope of ST	Number of vessels fluoro	Thallium
count	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000	270.000000

completed at 11:10 AM



2)Preprocessing:

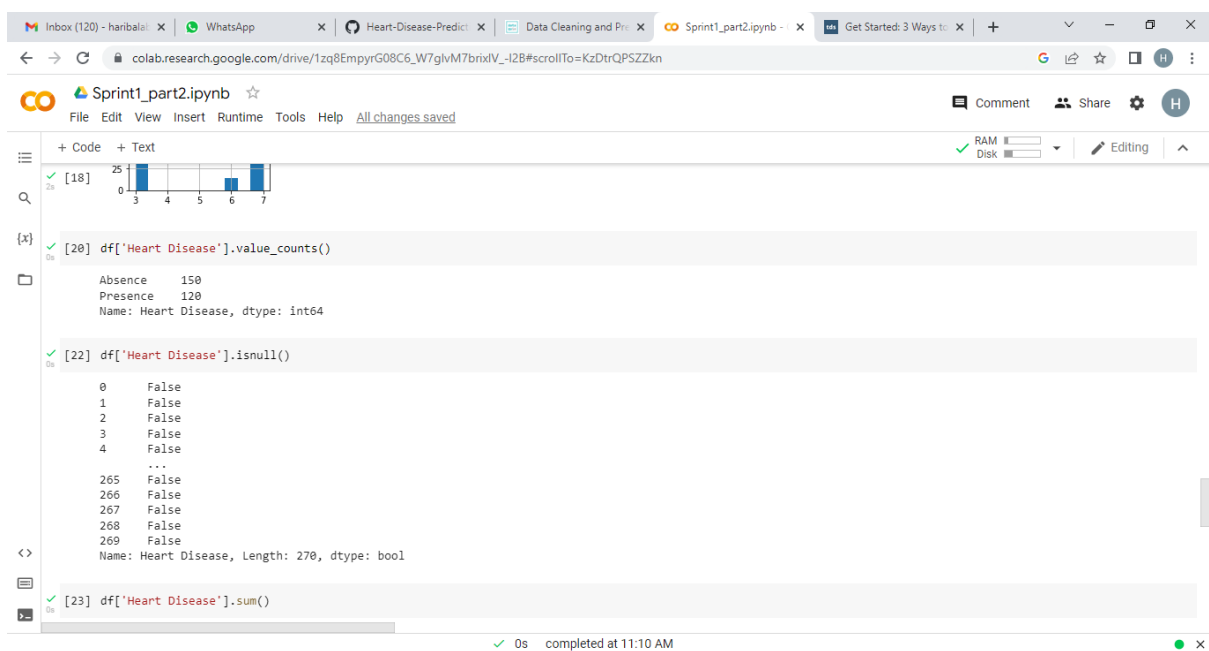
->Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

->A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

Steps:

- -> **Getting the dataset**
- **Importing libraries**
- **Importing datasets**
- **Finding Missing Data**
- **Encoding Categorical Data**
- **Splitting dataset into training and test set**
- **Feature scaling**



The screenshot shows a Jupyter Notebook titled 'Sprint1_part2.ipynb' in a web browser. The notebook contains three cells:

- Cell [23]: A long string of 'Presence' and 'Absence' values, likely representing a single row of data.
- Cell [24]: `df['Heart Disease'].unique()` resulting in `array(['Presence', 'Absence'], dtype=object)`.
- Cell [25]: `df.isnull().sum()` resulting in a summary of missing values for various features.

The output of cell [25] is as follows:

Feature	Count
Age	0
Sex	0
Chest pain type	0
BP	0
Cholesterol	0
FBS over 120	0
EKG results	0
Max HR	0
Exercise angina	0
ST depression	0
Slope of ST	0
Number of vessels fluro	0
Thallium	0
Heart Disease	0
dtype: int64	

The notebook interface shows it was completed at 11:10 AM.

3)Cleaning:

->Data scientists spend a large amount of their time cleaning datasets and getting them down to a form with which they can work. In fact, a lot of data scientists argue that the initial steps of obtaining and cleaning data constitute 80% of the job.

Therefore, if you are just stepping into this field or [planning to step into this field](#), it is important to be able to deal with messy data, whether that means missing values, inconsistent formatting, malformed records, or nonsensical outliers.

->Cleaning is not needed as this dataset already ready to solve the problem.