

**NAME : Nitesh Yadav**

**Subject : DATA SCIENCE**

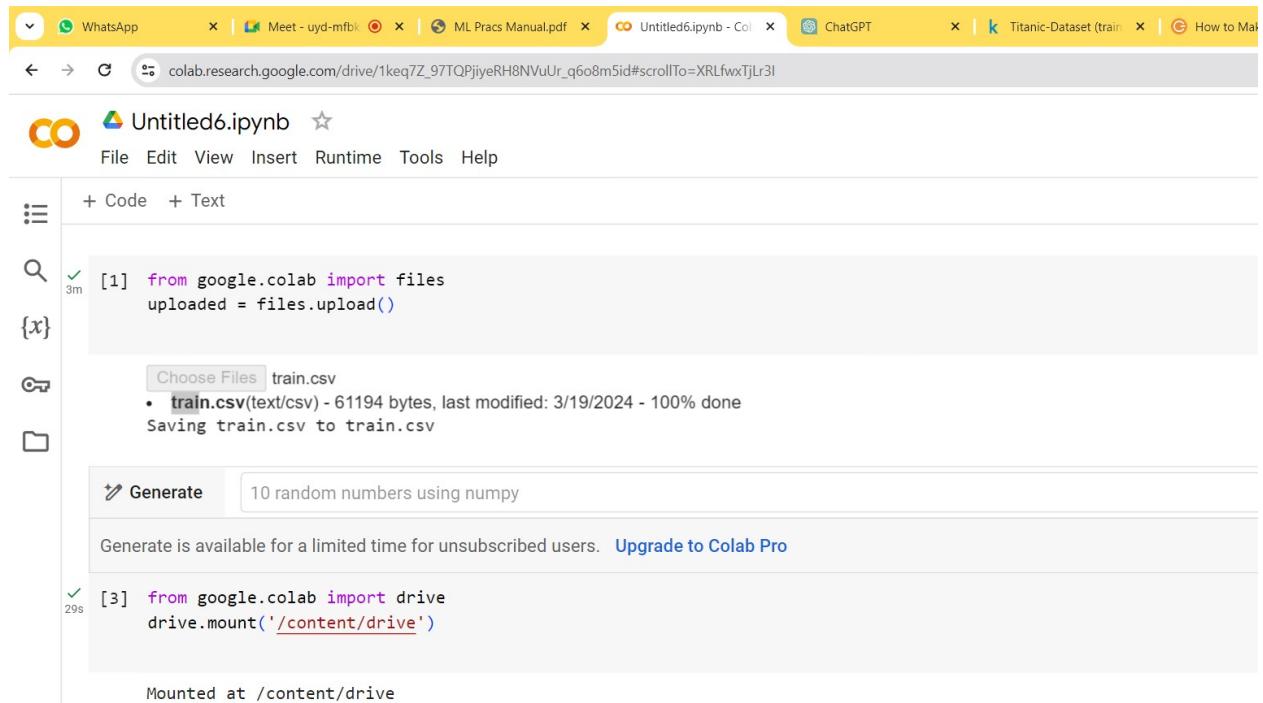
**Course :TYIT**

**Seat NO. : 310921060**

# Practical No. 1

## Q1. Importing data

To import a dataset into a Google Colab notebook, you have several options depending on where your dataset is located and the format it's in. Here's how you can import datasets from different sources:



The screenshot shows a Google Colab notebook interface. The title bar indicates the notebook is titled "Untitled6.ipynb". The code cell [1] contains the following Python code:

```
[1] from google.colab import files  
uploaded = files.upload()
```

Below the code cell, there is a "Choose Files" button with a "train.csv" placeholder. A message indicates the file has been uploaded: "• train.csv(text/csv) - 61194 bytes, last modified: 3/19/2024 - 100% done" and "Saving train.csv to train.csv".

The sidebar on the left shows a file tree with a single item: "train.csv". Below the sidebar, there is a "Generate" button followed by the text "10 random numbers using numpy". A note at the bottom says "Generate is available for a limited time for unsubscribed users. [Upgrade to Colab Pro](#)".

The code cell [3] contains the following Python code:

```
[3] from google.colab import drive  
drive.mount('/content/drive')
```

A message at the bottom of the cell area says "Mounted at /content/drive".

### 1. Upload from Local File System:

- If the dataset is on your local machine, you can upload it directly to your Colab notebook. You can use the following code to upload a file:

```
```python  
from google.colab import files  
uploaded = files.upload()  
```
```

- This will prompt you to select files from your local machine and upload them to the current working directory in your Colab notebook.

## 2. Mount Google Drive:

- If your dataset is stored in Google Drive, you can mount your Google Drive in the notebook and access the files. Follow these steps:

```
```python  
from google.colab import drive  
drive.mount('/content/drive')  
```
```

```

- After running this code, you'll need to authenticate and provide an authorization code. Once mounted, you can access your files under `/content/drive/My Drive/`.

### 3. Direct Download from URL:

- If the dataset is available via a direct URL, you can download it using `wget` or `curl`. For example:

```python

!wget <URL>

```

or

```python

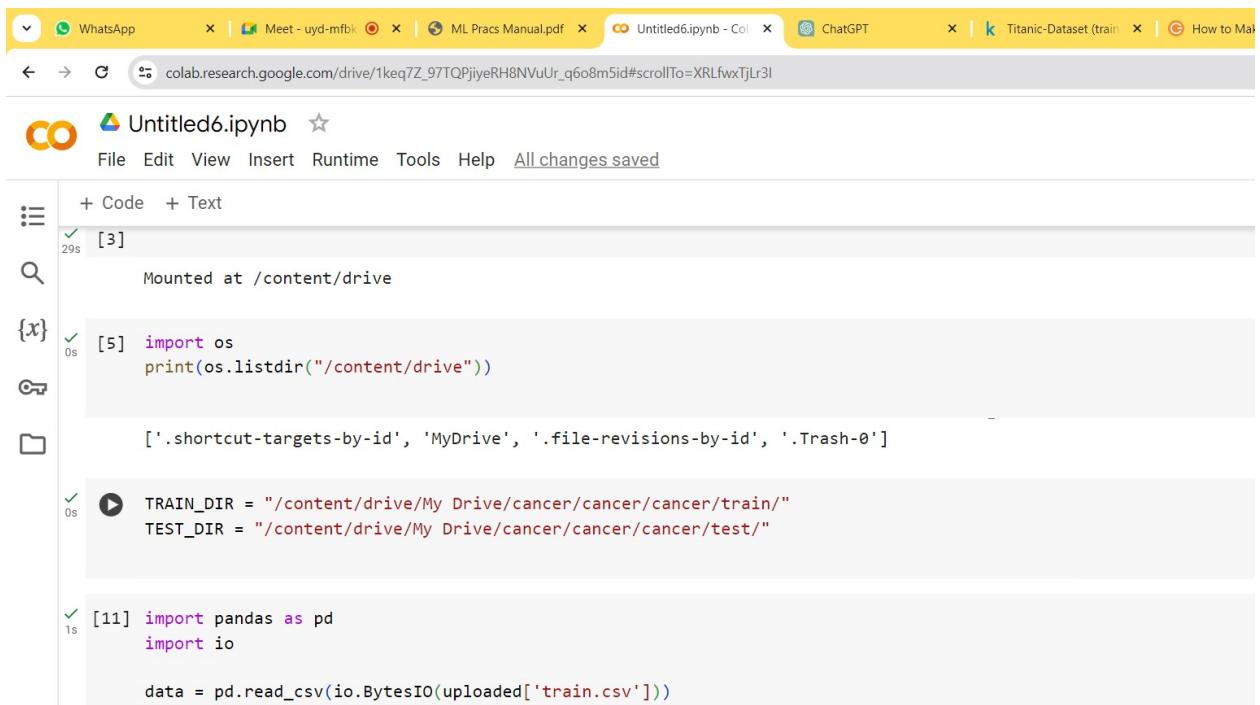
!curl -O <URL>

```

- Replace `<URL>` with the actual URL of the dataset.

### 4. Using APIs or Libraries:

- If your dataset is available through APIs or specific libraries like `torchvision` (for image datasets), `tensorflow\_datasets` (for TensorFlow datasets), or `scikit-learn` (for built-in datasets), you can directly load it using their respective methods.



The screenshot shows a Google Colab interface with the following details:

- Header:** WhatsApp, Meet - uyd-mfbk, ML Pracs Manual.pdf, Untitled6.ipynb - Colab, ChatGPT, Titanic-Dataset (train), How to Ma...
- Title Bar:** colab.research.google.com/drive/1keq7Z\_97TQPjiyeRH8NVuUr\_q6o8m5id#scrollTo=XRLfwxTjLr3I
- Toolbar:** File, Edit, View, Insert, Runtime, Tools, Help, All changes saved
- Sidebar:** + Code, + Text, search icon, {x} icon, refresh icon, folder icon.
- Code Cells:**
  - [3] Mounted at /content/drive
  - [5] 

```
import os
print(os.listdir("/content/drive"))
```

['.shortcut-targets-by-id', 'MyDrive', '.file-revisions-by-id', '.Trash-0']
  - [6] 

```
TRAIN_DIR = "/content/drive/My Drive/cancer/cancer/cancer/train/"
TEST_DIR = "/content/drive/My Drive/cancer/cancer/test/"
```
  - [11] 

```
import pandas as pd
import io

data = pd.read_csv(io.BytesIO(uploaded['train.csv']))
```

Choose the method that suits your dataset's location and format, and then proceed with further processing or analysis as needed in your Colab notebook.

## Practical No. 2

To summarize the dataset in colab notebook

Import Necessary Libraries: Make sure you have libraries like Pandas, NumPy, and any visualization libraries installed.

The screenshot shows a Google Colab notebook interface. The code cell at the top imports pandas, numpy, matplotlib.pyplot, and seaborn. The next cell reads a CSV file named 'train.csv' into a DataFrame. The subsequent cell contains four print statements: df.head(), df.info(), df.describe(), and df.isnull().sum(). The output of df.describe() and df.isnull().sum() is shown below:

```
[12] import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
[13] df = pd.read_csv('train.csv')
[14] print(df.head())
print(df.info())
print(df.describe())
print(df.isnull().sum())
PassengerId  Survived  Pclass \
0            1         0    3
1            2         1    1
2            3         1    3
3            4         1    1
4            5         0    3

Name      Sex   Age  SibSp \
0  Braund, Mr. Owen Harris   male  22.0     1
1  Cumings, Mrs. John Bradley (Florence Briggs Th... female  38.0     1
2  Heikkinen, Miss. Laina   female  26.0     0
3  Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35.0     1
4          Allen, Mr. William Henry   male  35.0     0
```

Load the Dataset: Load your dataset into a Pandas DataFrame.

Exploratory Data Analysis (EDA): Perform basic exploratory data analysis to understand the structure, content, and summary statistics of your dataset. Some common tasks include:

WhatsApp | Access Denied | Meet - uyd- | ML Pracs Manual | Untitled6.ipynb | ChatGPT | Titanic-Dataset | How to

colab.research.google.com/drive/1keq7Z\_97TQPjiyeRH8NVuUr\_q6o8m5id#scrollTo=BtinwPtFN8uv

### Untitled6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
0s
0s
{x}
print(df.head())
print(df.info())
print(df.describe())
print(df.isnull().sum())

Parch      Ticket      Fare Cabin Embarked
0         0     A/5 21171   7.2500   NaN      S
1         0       PC 17599  71.2833   C85      C
2         0  STON/O2. 3101282   7.9250   NaN      S
3         0    113803  53.1000  C123      S
4         0    373450  8.0500   NaN      S
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column      Non-Null Count  Dtype  
---  -- 
 0   PassengerId 891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object  
 4   Sex          891 non-null    object  
 5   Age          714 non-null    float64 
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object  
 9   Fare          891 non-null    float64 
 10  Cabin        204 non-null    object  
 11  Embarked     889 non-null    object  

```

WhatsApp | Access Denied | Meet - uyd- | ML Pracs Manual | Untitled6.ipynb | ChatGPT | Titanic-Dataset | How to

colab.research.google.com/drive/1keq7Z\_97TQPjiyeRH8NVuUr\_q6o8m5id#scrollTo=BtinwPtFN8uv

### Untitled6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
0s
0s
{x}
6   SibSp      891 non-null    int64
7   Parch      891 non-null    int64
8   Ticket     891 non-null    object
9   Fare        891 non-null    float64
10  Cabin       204 non-null    object
11  Embarked    889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
None
PassengerId  Survived  Pclass      Age      SibSp \
count    891.000000  891.000000  891.000000  714.000000  891.000000
mean     446.000000   0.383838  2.308642  29.699118   0.523008
std      257.353842   0.486592  0.836071 14.526497  1.102743
min      1.000000   0.000000  1.000000  0.420000  0.000000
25%    223.500000   0.000000  2.000000 20.125000  0.000000
50%    446.000000   0.000000  3.000000 28.000000  0.000000
75%    668.500000   1.000000  3.000000 38.000000  1.000000
max     891.000000   1.000000  3.000000 80.000000  8.000000

Parch      Fare
count    891.000000  891.000000
mean     0.381594  32.204208
std      0.806057  49.693429
min      0.000000  0.000000
25%    0.000000  7.918400
50%    0.000000 14.454200
75%    0.000000 31.000000
max     6.000000 512.329200
PassengerId  0
```

WhatsApp | Access Denied | Meet - uyd- | ML Pracs Manual | Untitled6.ipynb | ChatGPT | Titanic-Dataset | How to

colab.research.google.com/drive/1keq7Z\_97TQPjiyeRH8NVuUr\_q6o8m5id#scrollTo=BtinwPtFN8uv

### Untitled6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
Age      1//  
Sibsp    0  
Parch    0  
Ticket   0  
Fare     0  
Cabin    687  
Embarked 2  
dtype: int64
```

[x] 0s print(df.describe())

```
PassengerId  Survived  Pclass  Age  SibSp \  
count  891.000000  891.000000  891.000000  714.000000  891.000000  
mean   446.000000  0.383838  2.308642  29.699118  0.523008  
std    257.353842  0.486592  0.836071  14.526497  1.102743  
min    1.000000  0.000000  1.000000  0.420000  0.000000  
25%   223.500000  0.000000  2.000000  20.125000  0.000000  
50%   446.000000  0.000000  3.000000  28.000000  0.000000  
75%   668.500000  1.000000  3.000000  38.000000  1.000000  
max   891.000000  1.000000  3.000000  80.000000  8.000000  
  
Parch  Fare  
count  891.000000  891.000000  
mean   0.381594  32.204208  
std    0.806057  49.693429  
min    0.000000  0.000000  
25%   0.000000  7.910400
```

WhatsApp | Access Denied | Meet - uyd- | ML Pracs Manual | Untitled6.ipynb | ChatGPT | Titanic-Dataset | How to

colab.research.google.com/drive/1keq7Z\_97TQPjiyeRH8NVuUr\_q6o8m5id#scrollTo=BtinwPtFN8uv

### Untitled6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

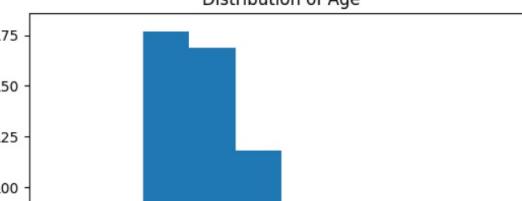
```
[17] print(df['Survived'].value_counts())
```

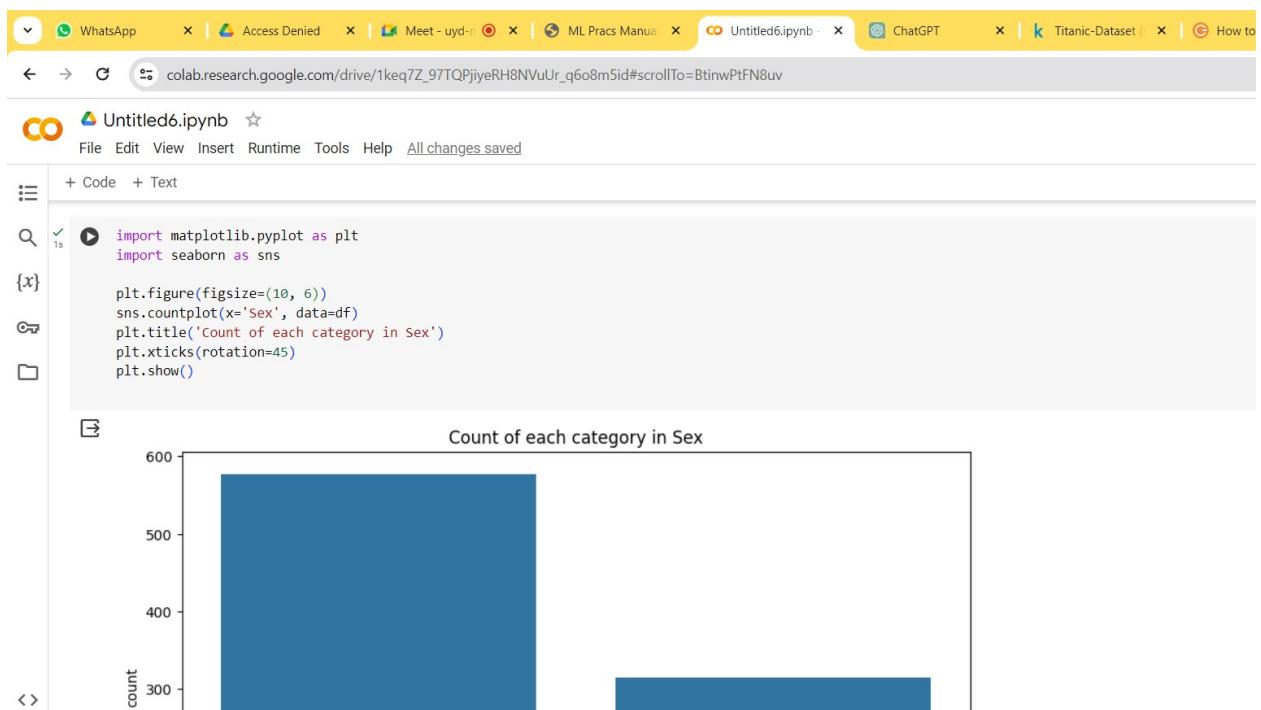
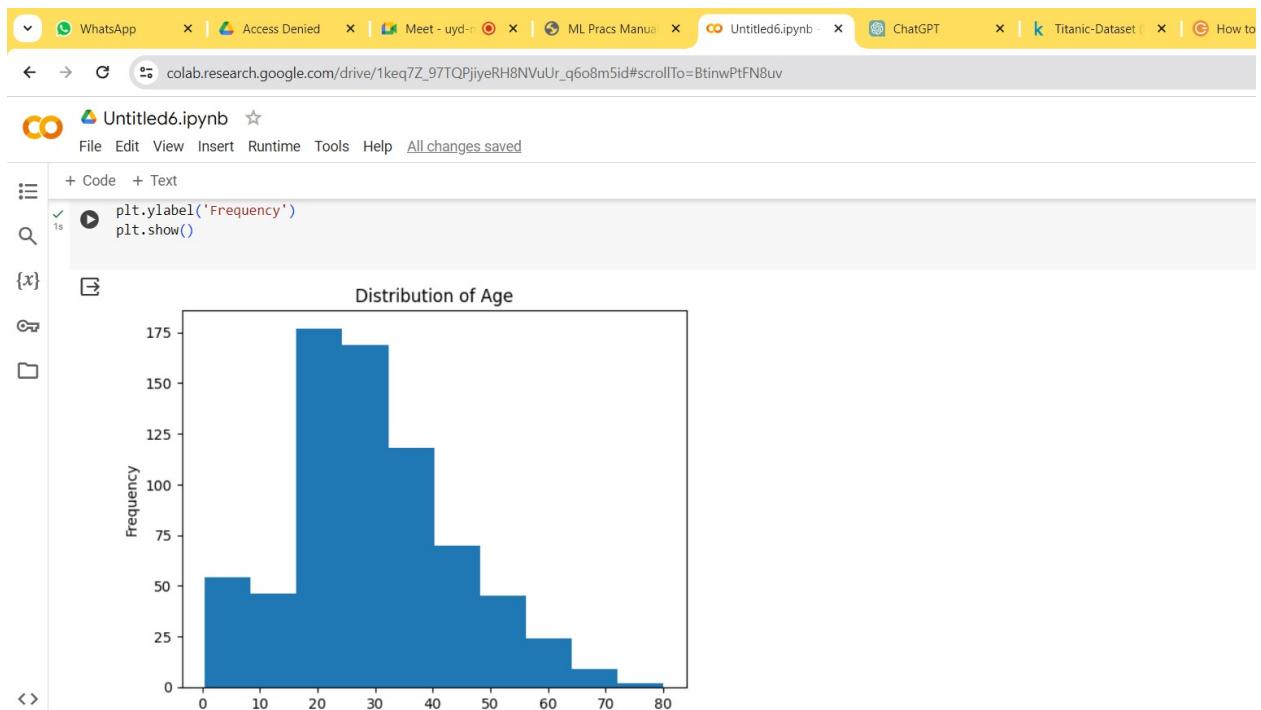
{x} 0s

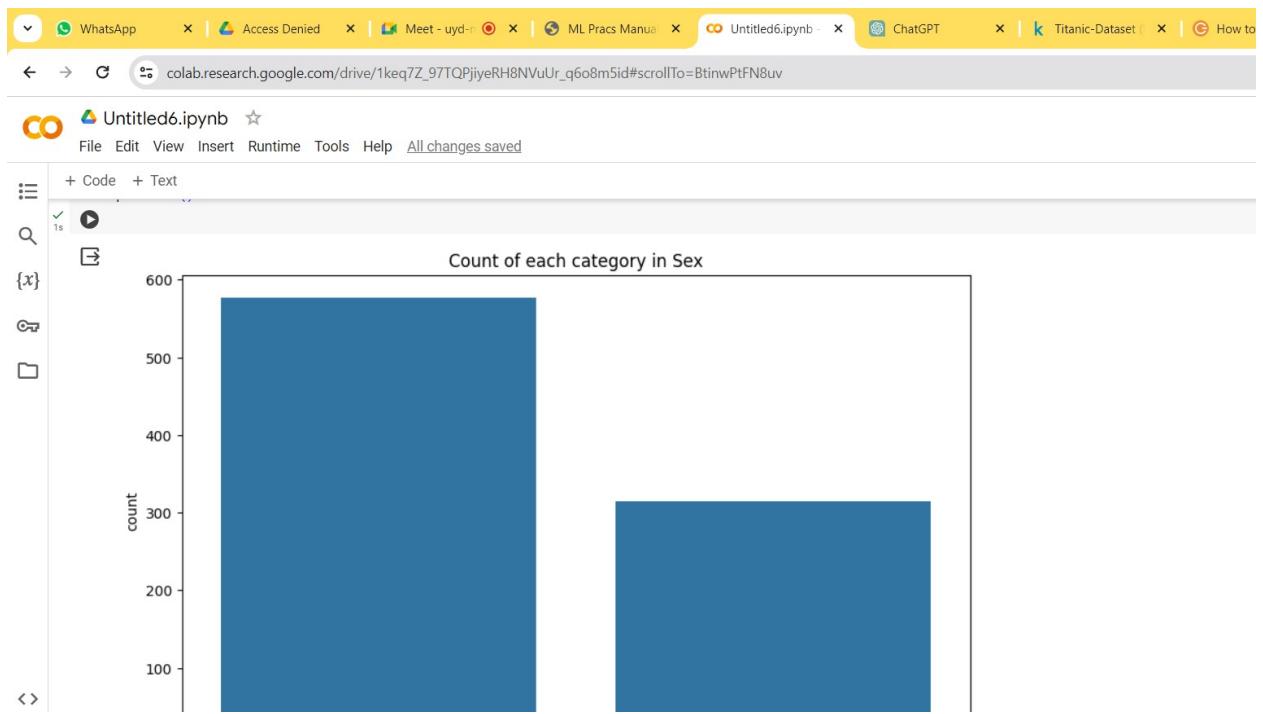
```
0    549  
1    342  
Name: Survived, dtype: int64
```

[18] 1s plt.hist(df['Age'])  
plt.title('Distribution of Age')  
plt.xlabel('Age')  
plt.ylabel('Frequency')  
plt.show()

Distribution of Age







## Practical No. 3

To perform Feature Scaling on the dataset

Untitled6.ipynb

```
[33]: import pandas as pd
      from sklearn.preprocessing import MinMaxScaler, StandardScaler
{x}
[34]: from google.colab import files
      uploaded = files.upload()

[35]: Choose File: train.csv
      • train.csv(text/csv) - 61194 bytes, last modified: 3/19/2024 - 100% done
      Saving train.csv to train (1).csv

[36]: print(uploaded.keys())

dict_keys(['train (1).csv'])

[37]: import pandas as pd
      import io

# Assuming you've already uploaded the file 'train (1).csv'
df = pd.read_csv(io.BytesIO(uploaded['train (1).csv']))
```

Untitled6.ipynb

```
[40]: import pandas as pd
      import io
{x}
[41]: # Assuming you've already uploaded the file 'train (1).csv'
      df = pd.read_csv(io.BytesIO(uploaded['train (1).csv']))

[42]: print("Size of the dataset:", df.shape)

Size of the dataset: (891, 12)

[43]: print("Null values in the dataset:", df.isnull().sum().sum())

Null values in the dataset: 866

[44]: print(df.head())
      PassengerId  Survived  Pclass \
0             1        0       3
1             2        1       1
2             3        1       3
```

WhatsApp | Access Denied | Meet - uyd- | ML Pracs Manual | Untitled6.ipynb | ChatGPT | Titanic-Dataset | How to

colab.research.google.com/drive/1keq7Z\_97TQPjiyeRH8NVuUr\_q6o8m5id#scrollTo=JF\_ipP-6R0Yi

### Untitled6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

Null values in the dataset: 866

```
✓ 0s [x] 0s print(df.head())
{x}
[43] df
  PassengerId  Survived  Pclass \
  0            1         0      3
  1            2         1      1
  2            3         1      3
  3            4         1      1
  4            5         0      3

  Name     Sex   Age  SibSp \
  0             Braund, Mr. Owen Harris    male  22.0      1
  1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
  2                Heikkinen, Miss. Laina  female  26.0      0
  3        Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
  4           Allen, Mr. William Henry    male  35.0      0

   Parch      Ticket     Fare Cabin Embarked
  0     0       A/5 21171  7.2500   NaN      S
  1     0       PC 17599  71.2833   C85      C
  2     0      STON/O2. 3101282  7.9250   NaN      S
  3     0      113803  53.1000  C123      S
  4     0      373450  8.0500   NaN      S
```

<> ✓ 0s [44] features\_to\_scale = ['feature1', 'feature2', 'feature3'] # Specify the features to scale

WhatsApp | Access Denied | Meet - uyd- | ML Pracs Manual | Untitled6.ipynb | ChatGPT | Titanic-Dataset | How to

colab.research.google.com/drive/1keq7Z\_97TQPjiyeRH8NVuUr\_q6o8m5id#scrollTo=JF\_ipP-6R0Yi

### Untitled6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
✓ 0s [43] df
  1     0      PC 17599  71.2833   C85      C
  2     0      STON/O2. 3101282  7.9250   NaN      S
  3     0      113803  53.1000  C123      S
  4     0      373450  8.0500   NaN      S
{x}
✓ 0s [44] features_to_scale = ['feature1', 'feature2', 'feature3'] # Specify the features to scale
  ✓ 0s [46] scaler_minmax = MinMaxScaler()
  ✓ 0s [47] df_scaled_minmax = df.copy()
  ✓ 0s [49] scaler_standard = StandardScaler()
  ✓ 0s [51] df_scaled_standard = df.copy()
  ✓ 0s [52] # Print column names of df_scaled_standard
            print(df_scaled_standard.columns)
```

WhatsApp | Access Denied | Meet - uyd- | ML Pracs Manual | Untitled6.ipynb | ChatGPT | Titanic-Dataset | How to

colab.research.google.com/drive/1keq7Z\_97TQPjiyeRH8NVuUr\_q6o8m5id#scrollTo=JF\_ipP-6R0Yi

Untitled6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
[52] print(df_scaled_standard.columns)
```

{x} [52]

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',  
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],  
      dtype='object')
```

0s # View the structure of df\_scaled\_standard  
print(df\_scaled\_standard.head())

0s

```
PassengerId  Survived  Pclass \
0            1         0      3
1            2         1      1
2            3         1      3
3            4         1      1
4            5         0      3

Name      Sex   Age  SibSp \
0  Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
2  Heikkinen, Miss. Laina    female  26.0      0
3  Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4  Allen, Mr. William Henry    male  35.0      0

Parch      Ticket     Fare Cabin Embarked
0            0    A/5 21171  7.2500   NaN        S
```

WhatsApp | Access Denied | Meet - uyd- | ML Pracs Manual | Untitled6.ipynb | ChatGPT | Titanic-Dataset | How to

colab.research.google.com/drive/1keq7Z\_97TQPjiyeRH8NVuUr\_q6o8m5id#scrollTo=JF\_ipP-6R0Yi

Untitled6.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
0s
```

{x} [52]

```
# Step 1: Verify Column Names
print(features_to_scale)
print(df_scaled_standard.columns)

# Step 2: Check DataFrame Columns
print(df_scaled_standard.columns)

# Step 3: Inspect DataFrame
print(df_scaled_standard.head())
```

0s

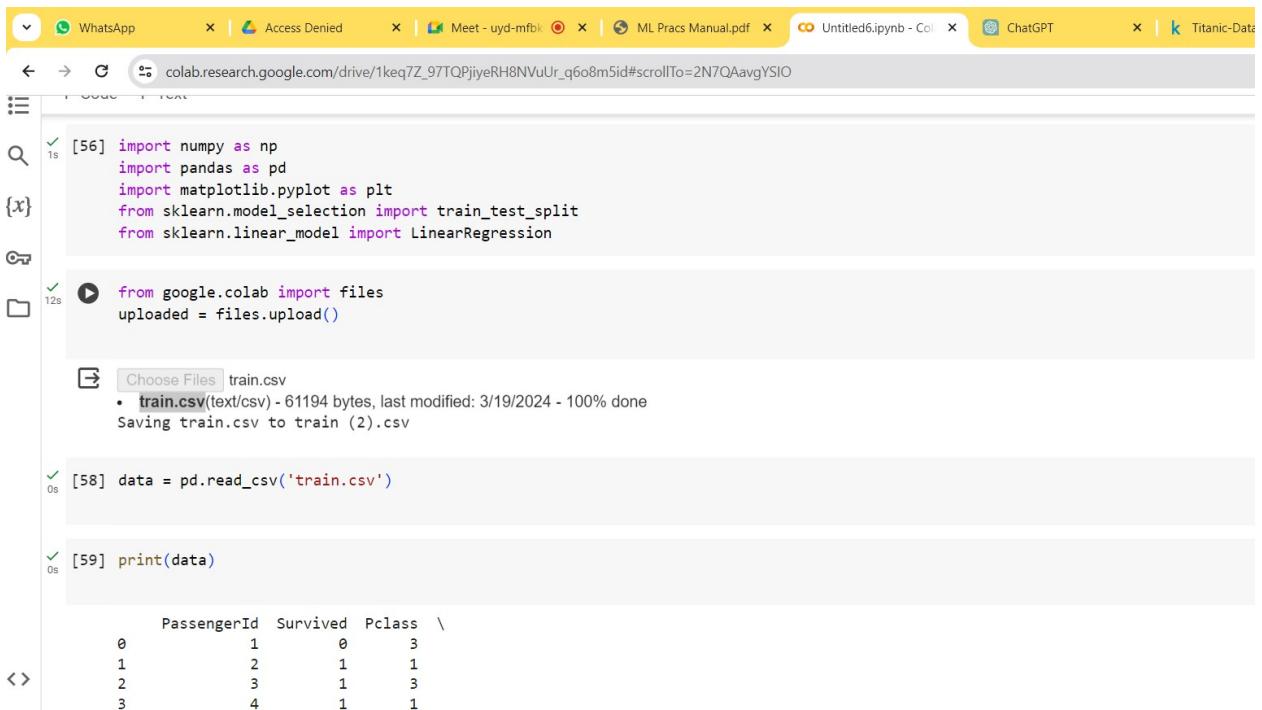
```
['feature1', 'feature2', 'feature3']
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',  
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],  
      dtype='object')
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',  
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],  
      dtype='object')
```

0s

```
PassengerId  Survived  Pclass \
0            1         0      3
1            2         1      1
2            3         1      3
3            4         1      1
4            5         0      3
```

# Practical No. 4

To perform linear regression using google



The screenshot shows a Google Colab notebook interface. The code cell [56] imports numpy, pandas, matplotlib.pyplot, train\_test\_split, and LinearRegression from sklearn. The code cell [125] uses google.colab.files.upload() to upload a CSV file named 'train.csv'. The code cell [58] reads the CSV file into a DataFrame named 'data'. The code cell [59] prints the first few rows of the DataFrame.

```
[56]: import numpy as np
       import pandas as pd
       import matplotlib.pyplot as plt
       from sklearn.model_selection import train_test_split
       from sklearn.linear_model import LinearRegression

[125]: from google.colab import files
       uploaded = files.upload()

[58]: data = pd.read_csv('train.csv')

[59]: print(data)
```

PassengerId	Survived	Pclass
0	1	3
1	2	1
2	3	1
3	4	1

```
WhatsApp x | Access Denied x | Meet - uyd-mfbk x | ML Pracs Manual.pdf x | Untitled6.ipynb - Col x | ChatGPT x | k Titanic-Data x

colab.research.google.com/drive/1keq7Z_97TQPjiyeRH8NVuUr_q6o8m5id#scrollTo=2N7QAavgYSIO
```

[59]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Cabin	Embarked
0	886	0	211536	13.0000	NaN	S			
1	887	0	112053	30.0000	B42	S			
2	888	2	W./C. 6607	23.4500	NaN	S			
3	889	0	111369	30.0000	C148	C			
4	890	0	370376	7.7500	NaN	Q			

[891 rows x 12 columns]

# Check if column names exist in data  
print("Column names in data:", data.columns)

# Print the first few rows of data to inspect its structure  
print(data.head())

Column names in data: Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],  
dtype='object')

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp
0	1	0	Braund, Mr. Owen Harris	male	22.0	1
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th... Heikkinen, Miss. Laina	female	38.0	1 0
2	3	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1
3	4	1	Allen, Mr. William Henry	male	35.0	0
4	5	0				

```
WhatsApp x | Access Denied x | Meet - uyd-mfbk x | ML Pracs Manual.pdf x | Untitled6.ipynb - Col x | ChatGPT x | k Titanic-Data x

colab.research.google.com/drive/1keq7Z_97TQPjiyeRH8NVuUr_q6o8m5id#scrollTo=2N7QAavgYSIO
```

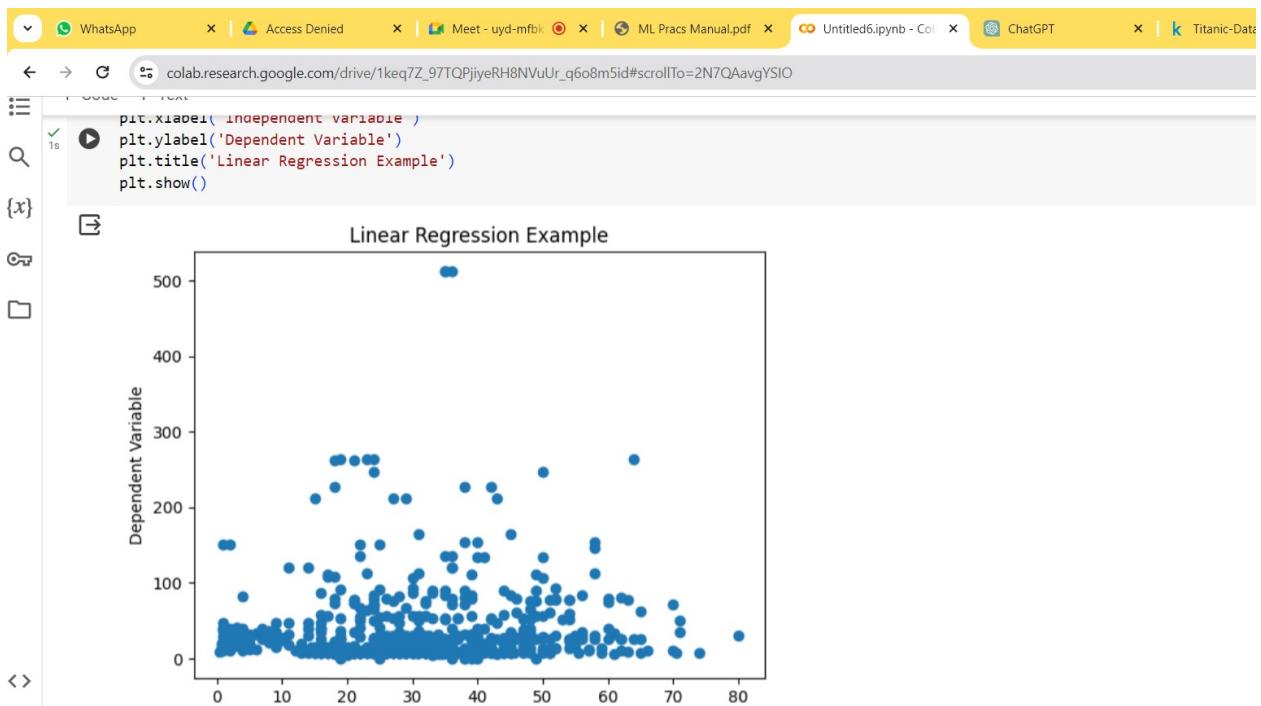
[61]:

	PassengerId	Survived	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	886	0	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	887	0	Cumings, Mrs. John Bradley (Florence Briggs Th... Heikkinen, Miss. Laina	female	38.0	1 0	0	PC 17599	71.2833	C85	C
2	888	2	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	STON/O2. 3101282	7.9250	NaN	S
3	889	0	Allen, Mr. William Henry	male	35.0	0	0	113803	53.1000	C123	S
4	890	0					0	373450	8.0500	NaN	S

[62]: X = data['Age'].values.reshape(-1, 1)  
Y = data['Fare'].values

plt.scatter(X, Y)  
plt.xlabel('Independent Variable')  
plt.ylabel('Dependent Variable')  
plt.title('Linear Regression Example')  
plt.show()

Linear Regression Example



```
[63] 0s [64] model = LinearRegression()
[65] from sklearn.model_selection import train_test_split
      from sklearn.linear_model import LinearRegression
      from sklearn.impute import SimpleImputer

      # Impute missing values in X
      imputer = SimpleImputer(strategy='mean')
      X_imputed = imputer.fit_transform(X)

      # Split the data into training and testing sets
      X_train, X_test, Y_train, Y_test = train_test_split(X_imputed, Y, test_size=0.2, random_state=0)

      # Initialize and fit the Linear Regression model
      model = LinearRegression()
      model.fit(X_train, Y_train)
```

LinearRegression  
LinearRegression()

WhatsApp | Access Denied | Meet - uyd-mfbk | ML Pracs Manual.pdf | Untitled6.ipynb - Col | ChatGPT | Titanic-Data

colab.research.google.com/drive/1keq7Z\_97TQPjiyeRH8NVuUr\_q6o8m5id#scrollTo=2N7QAavgYSIO

Generate 10 random numbers using numpy

{x}

Generate is available for a limited time for unsubscribed users. [Upgrade to Colab Pro](#)

```
# Step 6: Initializing X and Y variables
X = data['Age'].values.reshape(-1, 1) # Independent variable
Y = data['Fare'].values # Dependent variable

# Step 7: Handle missing values
imputer = SimpleImputer(strategy='mean')
X = imputer.fit_transform(X)

# Step 8: Plotting the graph
plt.scatter(X, Y)
plt.xlabel('Age')
plt.ylabel('Fare')
plt.title('Linear Regression Example')
plt.show()

# Step 9: Creating the object for the regression
model = LinearRegression()

# Step 10: Training the dataset
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=0)
model.fit(X_train, Y_train)
```

WhatsApp | Access Denied | Meet - uyd-mfbk | ML Pracs Manual.pdf | Untitled6.ipynb - Col | ChatGPT | Titanic-Data

colab.research.google.com/drive/1keq7Z\_97TQPjiyeRH8NVuUr\_q6o8m5id#scrollTo=2N7QAavgYSIO

```
plt.ylabel('Fare')
plt.title('Linear Regression Example')
plt.show()

# Step 9: Creating the object for the regression
model = LinearRegression()

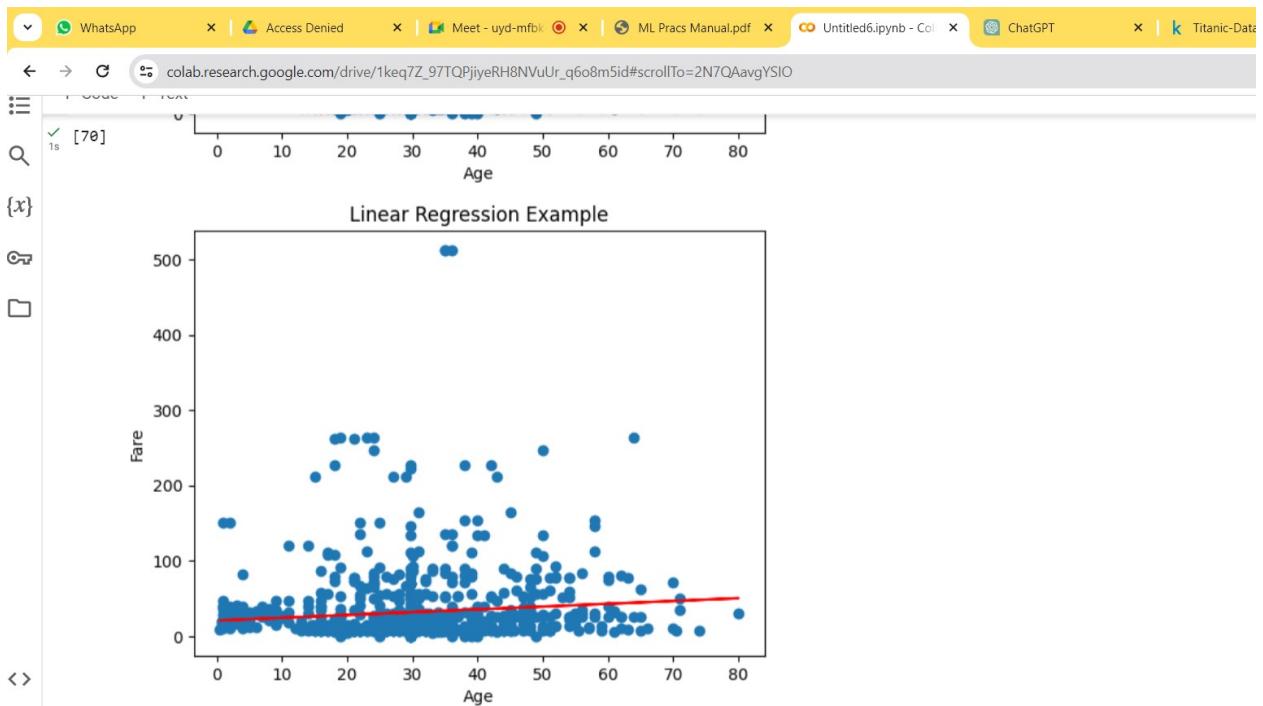
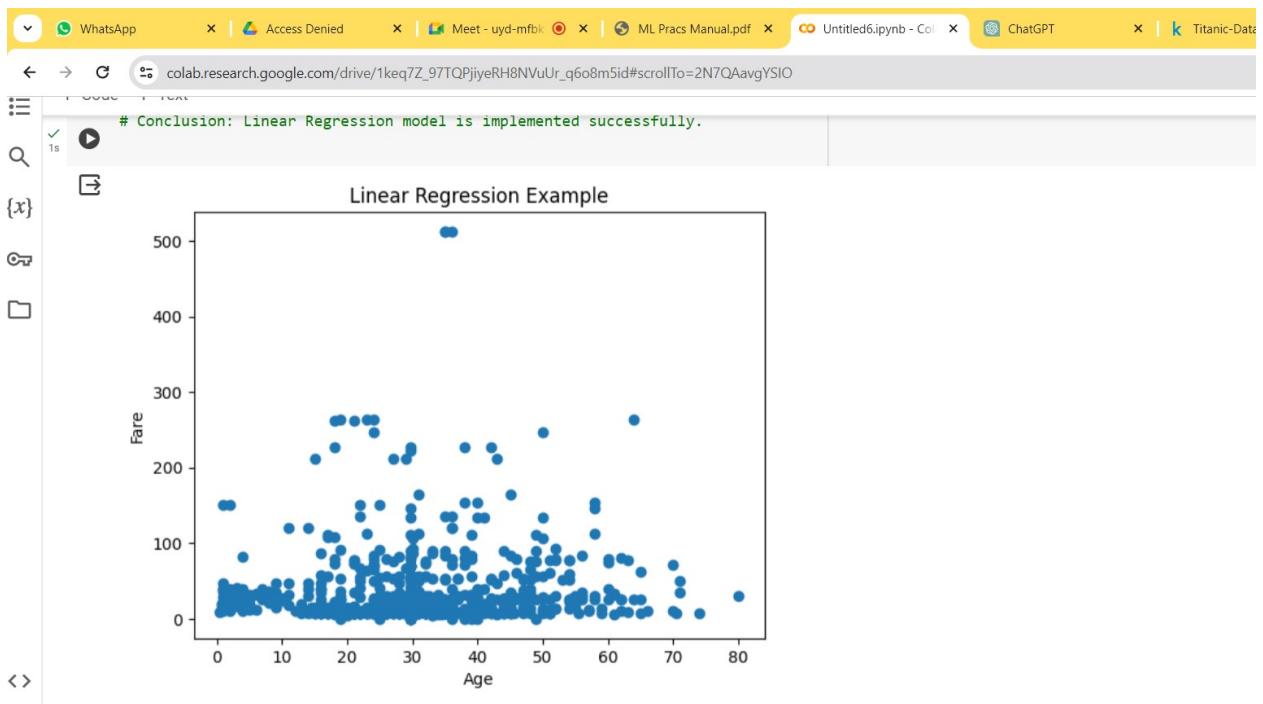
# Step 10: Training the dataset
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=0)
model.fit(X_train, Y_train)

# Step 11: Plotting the regression line
plt.scatter(X, Y)
plt.plot(X, model.predict(X), color='red')
plt.xlabel('Age')
plt.ylabel('Fare')
plt.title('Linear Regression Example')
plt.show()

# Conclusion: Linear Regression model is implemented successfully.
```

Linear Regression Example





## Practical No. 5

# To perform Multiple Linear Regression using google colab

The screenshot shows a Google Colab notebook titled "Untitled6.ipynb". The code cell [71] imports pandas, numpy, and various sklearn modules. The code cell [72] reads the "train.csv" dataset. The code cell [73] prints the first few rows of the dataset, showing columns like PassengerId, Survived, Pclass, Name, Sex, Age, and SibSp.

```
[71]: import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
import matplotlib.pyplot as plt

[72]: data = pd.read_csv('train.csv')

[73]: print("Dataset:")
print(data.head())

Dataset:
   PassengerId  Survived  Pclass \
0              1         0      3
1              2         1      1
2              3         1      3
3              4         1      1
4              5         0      3

  Name     Sex   Age  SibSp \
0    Braund, Mr. Owen Harris       male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
```

The screenshot shows a Google Colab notebook titled "Untitled6.ipynb". The code cell [73] prints the first five rows of the dataset. The code cell [74] checks for null data using the isnull() method. The code cell [75] prints the summary statistics of the dataset using the describe() method.

```
[73]: data.head()

[74]: if data.isnull().sum().sum() == 0:
    print("No null data found")
else:
    print("Null data found")

[75]: print("Summary Statistics:")
print(data.describe())

Summary Statistics:
   PassengerId  Survived  Pclass     Age  SibSp \
count    891.000000  891.000000  891.000000  714.000000  891.000000
mean     446.000000   0.383838   2.308642  29.699118   0.523008
std      257.353842   0.486592   0.836071  14.526497   1.102743
min      1.000000   0.000000   1.000000   0.420000   0.000000
25%    223.500000   0.000000   2.000000  20.125000   0.000000
50%    446.000000   0.000000   3.000000  28.000000   0.000000
75%    668.500000   1.000000   3.000000  38.000000   1.000000
max    834.000000  100.000000  3.000000  80.000000  15.000000
```

```
[75] 25%    0.000000  7.910400
[75] 50%    0.000000  14.454200
[75] 75%    0.000000  31.000000
[75] max    6.000000  512.329200

# Check if 'target_column_name' exists in data
print("'target_column_name' in data:", 'target_column_name' in data.columns)

# Print the first few rows of data to inspect its structure
print(data.head())

'target_column_name' in data: False
PassengerId Survived Pclass \
0            1        0      3
1            2        1      1
2            3        1      3
3            4        1      1
4            5        0      3

Name      Sex   Age SibSp \
0 Braund, Mr. Owen Harris male  22.0     1
1 Cumings, Mrs. John Bradley (Florence Briggs Th... female 38.0     1
2 Heikkinen, Miss. Laina female 26.0     0
3 Futrelle, Mrs. Jacques Heath (Lily May Peel) female 35.0     1
4 Allen, Mr. William Henry male  35.0     0
```

```
import pandas as pd

# Assuming 'target_column_name' is the name of your target variable
column_names = ['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked']
data = pd.read_csv('train.csv', names=column_names)

# Verify if 'Survived' exists in data.columns
print("Survived" in data.columns:, 'Survived' in data.columns)

'Survived' in data.columns: True

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

# Assuming you have already trained your model and named it 'model'
# Predict the target variable using the trained model and test data
Y_pred = model.predict(X_test)

# Calculate the R^2 score
r2 = r2_score(Y_test, Y_pred)
print("The r2 score is {:.2f}% which is good.".format(r2 * 100))

# Plotting the result (optional)
```



```

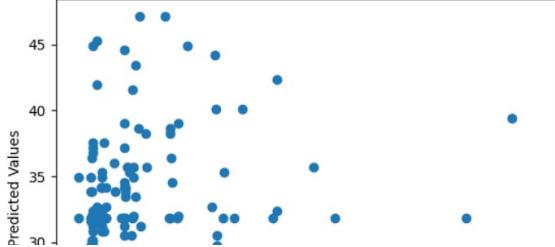
r2 = r2_score(Y_test, Y_pred)
print("The r2 score is {:.2f}% which is good.".format(r2 * 100))

# Plotting the result (optional)
plt.scatter(Y_test, Y_pred)
plt.xlabel("Actual Values")
plt.ylabel("Predicted Values")
plt.title("Actual vs Predicted Values")
plt.show()

```

The r2 score is 1.68% which is good.

Actual vs Predicted Values



## Practical No. 6

To perform Multiple Logistic Regression using google colab

ML Pracs Manual.pdf - Google | DS1.ipynb - Colaboratory | ChatGPT

colab.research.google.com/drive/1ND4WJUH05zdUHhOhGzImWjpkvDP4UcE#scrollTo=\_B\_5Y9AQMKVf

DS1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 10:57 PM

+ Code + Text

```
[ ]
```

{x}

```
[ ] model = LogisticRegression()
model.fit(x_train, y_train)
```

LogisticRegression

```
[ ] y_pred = model.predict(x_test)
```

print(classification\_report(y\_test,y\_pred))

	precision	recall	f1-score	support
0	0.80	0.98	0.88	41
1	0.90	0.47	0.62	19
accuracy			0.82	60
macro avg	0.85	0.72	0.75	60
weighted avg	0.83	0.82	0.80	60

< >

ML Pracs Manual.pdf - Google | DS1.ipynb - Colaboratory | ChatGPT

colab.research.google.com/drive/1ND4WJUH05zdUHhOhGzImWjpkvDP4UcE#scrollTo=\_B\_5Y9AQMKVf

DS1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 10:57 PM

+ Code + Text

```
v      v.vv  v.vv  v.vv  "1
1      0.90  0.47  0.62  19
```

{x}

```
accuracy
macro avg
weighted avg
```

accuracy

```
[[40  1]
 [10  9]]
0.8166666666666667
```

```
mylist = []

cm = confusion_matrix(y_test, y_pred)
ac = accuracy_score(y_test, y_pred)
mylist.append(ac)
print(cm)
print(ac)
```

```
[[40  1]
 [10  9]]
0.8166666666666667
```

```
print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
0	0.80	0.98	0.88	41
1	0.90	0.47	0.62	19

< >

The screenshot shows a Google Colab interface with three tabs at the top: 'ML Pracs Manual.pdf - Google' (closed), 'DS1.ipynb - Colaboratory' (active), and 'ChatGPT' (closed). The URL in the address bar is 'colab.research.google.com/drive/1ND4WJUH05zdUHhOhGzImWjpkvDP4UcE#scrollTo=\_B\_5Y9AQMKVf'. The notebook content includes:

File Edit View Insert Runtime Tools Help Last saved at 10:57 PM

+ Code + Text

Code cell content:

```
[ ] mylist = []
cm = confusion_matrix(y_test, y_pred)
ac = accuracy_score(y_test, y_pred)
mylist.append(ac)
print(cm)
print(ac)

[[40  1]
 [10  9]]
0.8166666666666667
```

Output cell content (with a play icon):

```
print(classification_report(y_test,y_pred))
```

Output of the classification report:

	precision	recall	f1-score	support
0	0.80	0.98	0.88	41
1	0.90	0.47	0.62	19

## Practical No. 7

To build Decision Tree Model using google

The screenshot shows a Google Colab notebook titled "DS1.ipynb". The code cell contains imports for pandas, numpy, matplotlib.pyplot, and seaborn, followed by imports for LinearRegression, LogisticRegression, SVC, confusion\_matrix, classification\_report, accuracy\_score, train\_test\_split, and DecisionTreeClassifier. A second code cell shows the command to mount Google Drive, resulting in the message "Mounted at /content/drive". A third code cell starts with "import pandas as pd".

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# import models

from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.svm import SVC

from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier

from google.colab import drive
drive.mount('/content/drive')

Mounted at /content/drive

import pandas as pd
```

The screenshot shows a Google Colab notebook titled "DS1.ipynb". The code cell contains the command "drive.mount('/content/drive')", resulting in the message "Mounted at /content/drive". A second code cell starts with "import pandas as pd" and reads a CSV file named "heart\_failure\_clinical\_records\_dataset.csv". Subsequent code cells handle feature selection, splitting the data into training and testing sets, and printing the shapes of the resulting datasets.

```
drive.mount('/content/drive')

import pandas as pd
df=pd.read_csv('/content/drive/MyDrive/dataset/heart_failure_clinical_records_dataset.csv')
df.head()

x = df.drop("DEATH_EVENT", axis=1)
y = df['DEATH_EVENT']

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

x = scaler.fit_transform(x)
y = scaler.fit_transform(y.values.reshape(-1,1))
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, shuffle = True, random_state = 42, stratify = y)

print(f"The Shape of x_train : {x_train.shape}")
print(f"The Shape of x_test : {x_test.shape}")
print(f"The Shape of y_train : {y_train.shape}")
```

The screenshot shows a Google Colab interface with three tabs at the top: 'ML Pracs Manual.pdf - Google' (closed), 'DS1.ipynb - Colaboratory' (active), and another 'DS1.ipynb - Colaboratory' tab. The URL in the address bar is 'colab.research.google.com/drive/1ND4WJUH05zdUHhOhGzImWjpkvDP4UcE#scrollTo=xP01OF--APgw'. The notebook content is as follows:

```
[ ] print(classification_report(y_test,y_pred))

{x}
[ ] print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

from sklearn.model_selection import GridSearchCV

param_grid = {
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
}

grid_search = GridSearchCV(estimator=classifier, param_grid=param_grid, cv=3, n_jobs=-1, verbose=2)
grid_search.fit(x_train, y_train)

print("Best parameters:", grid_search.best_params_)

# Use the best parameters to make predictions
best_model = grid_search.best_estimator_
y_pred_optimized = best_model.predict(x_test)
```

## Practical No. 8

To build Suport Vector Machine model (SVM) using google colab

ML Pracs Manual.pdf - Google | DS1.ipynb - Colaboratory | ChatGPT

colab.research.google.com/drive/1ND4WJUH05zdUHhOhGzZlmWjpkvDP4UcE#scrollTo=791Kq6WjLroW

DS1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 10:57 PM

+ Code + Text

Dataset used : <https://www.kaggle.com/datasets/andrewmd/heart-failure-clinical-data/data>

```
[ ]
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# import models

from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
```

```
[ ]
```

```
from google.colab import drive
```

ML Pracs Manual.pdf - Google | DS1.ipynb - Colaboratory | ChatGPT

colab.research.google.com/drive/1ND4WJUH05zdUHhOhGzZlmWjpkvDP4UcE#scrollTo=791Kq6WjLroW

DS1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 10:57 PM

+ Code + Text

```
[ ]
```

```
from google.colab import drive
drive.mount('/content/drive')
```

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force\_remount=True).

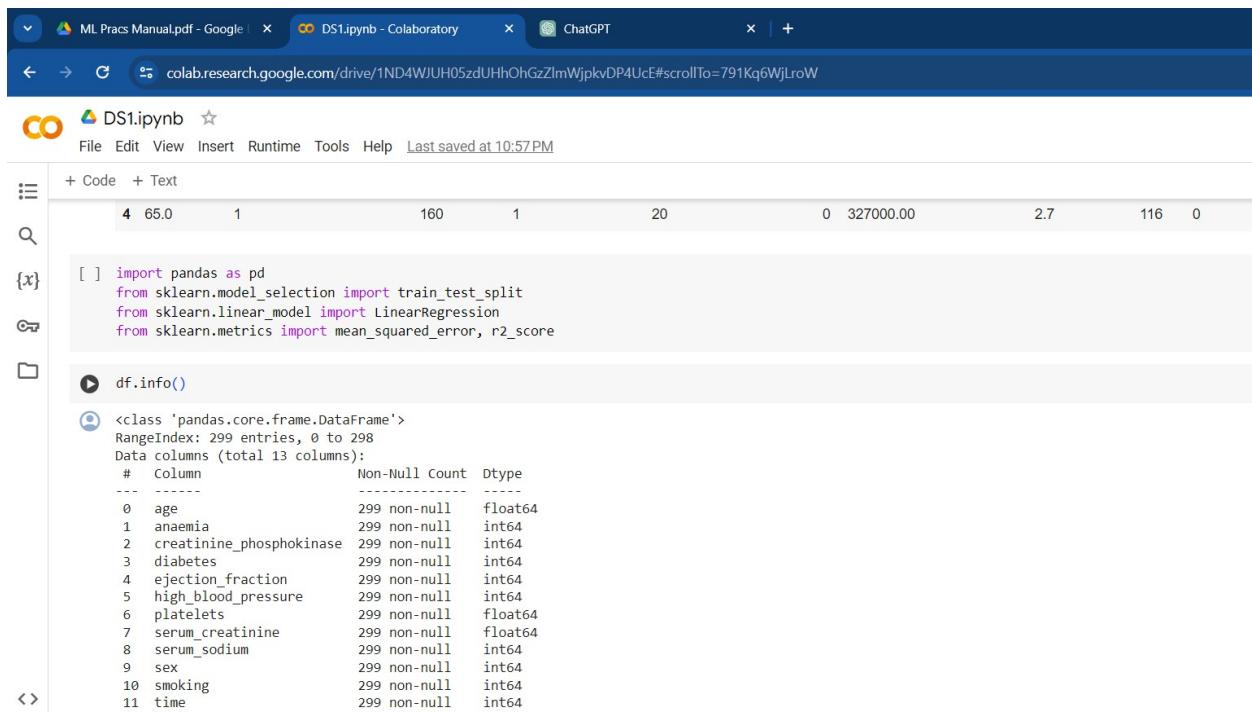
```
[ ]
```

```
import pandas as pd
df=pd.read_csv('/content/drive/MyDrive/dataset/heart_failure_clinical_records_dataset.csv')
df.head()
```

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking
0	75.0	0	582	0	20	1	265000.00	1.9	130	1	
1	55.0	0	7861	0	38	0	263358.03	1.1	136	1	
2	65.0	0	146	0	20	0	162000.00	1.3	129	1	
3	50.0	1	111	0	20	0	210000.00	1.9	137	1	
4	65.0	1	160	1	20	0	327000.00	2.7	116	0	

```
[ ]
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
```



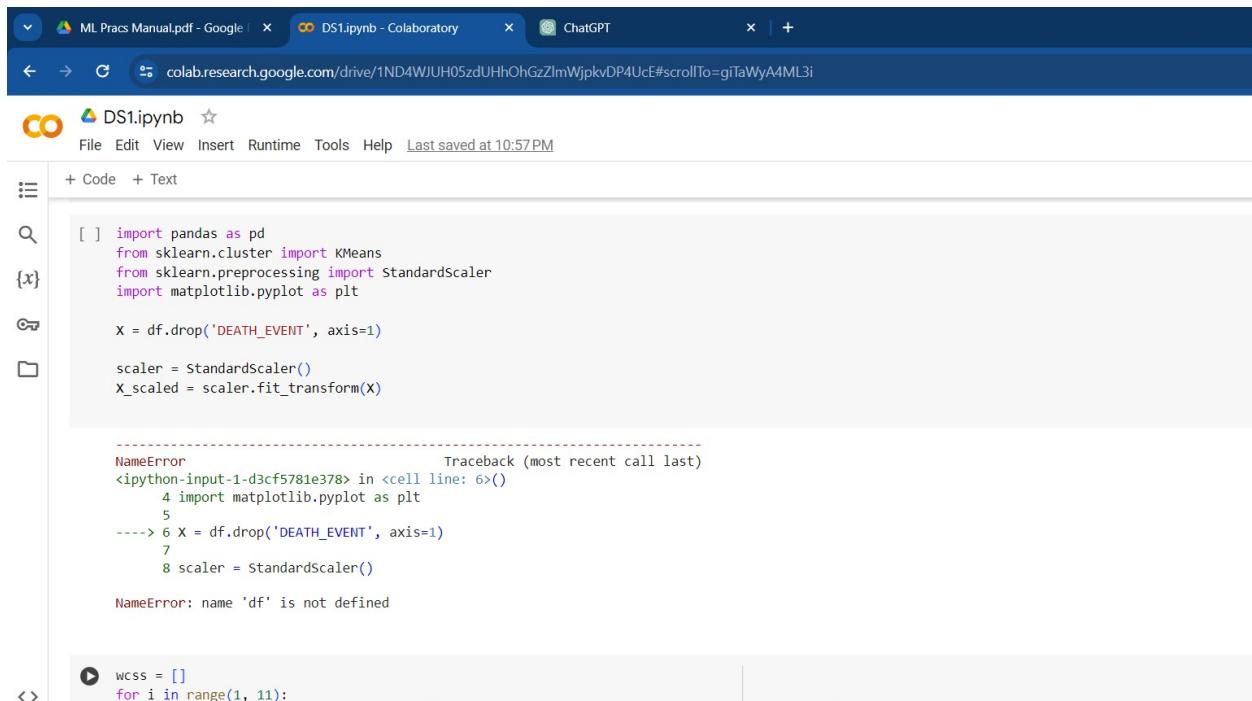
```
[ ] import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

{x} df.info()

[ ] <class 'pandas.core.frame.DataFrame'>
RangeIndex: 299 entries, 0 to 298
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   age              299 non-null    float64
 1   anaemia          299 non-null    int64  
 2   creatinine_phosphokinase 299 non-null  int64  
 3   diabetes          299 non-null    int64  
 4   ejection_fraction 299 non-null    int64  
 5   high_blood_pressure 299 non-null    int64  
 6   platelets         299 non-null    float64
 7   serum_creatinine 299 non-null    float64
 8   serum_sodium      299 non-null    int64  
 9   sex               299 non-null    int64  
 10  smoking           299 non-null    int64  
 11  time              299 non-null    int64
```

## Practical No. 9

### perform clustering using KMeans clustering



```
[ ] import pandas as pd
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt

{x} X = df.drop('DEATH_EVENT', axis=1)

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

NameError: name 'df' is not defined

{ } wcss = []
for i in range(1, 11):
```

ML Pracs Manual.pdf - Google | DS1.ipynb - Colaboratory | ChatGPT

colab.research.google.com/drive/1ND4WJUH05zdUHhOhGzImWjpkvDP4UcE#scrollTo=giTaWyA4ML3i

### DS1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 10:57 PM

```
+ Code + Text
    7
    8 scaler = StandardScaler()

NameError: name 'df' is not defined

{x}
[ ] wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10, random_state=42)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)

plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()

k = 3 # example value
kmeans = KMeans(n_clusters=k, init='k-means++', max_iter=300, n_init=10, random_state=42)
y_kmeans = kmeans.fit_predict(X_scaled)
```

ML Pracs Manual.pdf - Google | DS1.ipynb - Colaboratory | ChatGPT

colab.research.google.com/drive/1ND4WJUH05zdUHhOhGzImWjpkvDP4UcE#scrollTo=giTaWyA4ML3i

### DS1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 10:57 PM

```
+ Code + Text
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)

plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()

k = 3 # example value
kmeans = KMeans(n_clusters=k, init='k-means++', max_iter=300, n_init=10, random_state=42)
y_kmeans = kmeans.fit_predict(X_scaled)

from sklearn.decomposition import PCA

pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

plt.scatter(X_pca[y_kmeans == 0, 0], X_pca[y_kmeans == 0, 1], s = 100, c = 'red', label = 'Cluster 1')
plt.scatter(X_pca[y_kmeans == 1, 0], X_pca[y_kmeans == 1, 1], s = 100, c = 'blue', label = 'Cluster 2')
plt.scatter(X_pca[y_kmeans == 2, 0], X_pca[y_kmeans == 2, 1], s = 100, c = 'green', label = 'Cluster 3')
plt.title('Clusters of patients')
plt.xlabel('PCA 1')
```

# Practical No. 10 Aim:

## To perform reinforcement learning

The screenshot shows a Google Colab interface with the notebook titled "DS1.ipynb". The code cell contains the following content:

```
Taxi(v3)
[ ]
Taxi(v3) game:
- The Taxi-v3 environment is a classic grid-world problem.
- The goal is for a taxi agent to pick up passengers and drop them off at specified locations.
- The agent navigates a 5x5 grid world with passengers, destinations, and walls.
- The task involves learning an optimal policy to maximize rewards while obeying traffic rules.

[ ] !pip install gym

Requirement already satisfied: gym in /usr/local/lib/python3.10/dist-packages (0.25.2)
Requirement already satisfied: numpy>=1.18.0 in /usr/local/lib/python3.10/dist-packages (from gym) (1.25.2)
Requirement already satisfied:云pickle>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from gym) (2.2.1)
Requirement already satisfied: gym-notices>=0.0.4 in /usr/local/lib/python3.10/dist-packages (from gym) (0.0.8)
```

The screenshot shows a Google Colab notebook titled "DS1.ipynb". The code cell contains the following Python script:

```
[ ] !pip install gym

Requirement already satisfied: gym in /usr/local/lib/python3.10/dist-packages (0.25.2)
Requirement already satisfied: numpy>=1.18.0 in /usr/local/lib/python3.10/dist-packages (from gym) (1.25.2)
Requirement already satisfied:云pickle>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from gym) (2.2.1)
Requirement already satisfied: gym-notices>=0.0.4 in /usr/local/lib/python3.10/dist-packages (from gym) (0.0.8)

[ ] import gym
import numpy as np

# Create the Taxi-v3 environment
env = gym.make('Taxi-v3')

# Initialize Q-table with zeros
num_states = env.observation_space.n
num_actions = env.action_space.n
Q_table = np.zeros((num_states, num_actions))

# Q-learning parameters
alpha = 0.1 # Learning rate
```

The screenshot shows the continuation of the Google Colab notebook "DS1.ipynb". The code cell contains the following Python script:

```
num_actions = env.action_space.n
Q_table = np.zeros((num_states, num_actions))

# Q-learning parameters
alpha = 0.1 # Learning rate
gamma = 0.6 # Discount factor
epsilon = 0.1 # Exploration rate
num_episodes = 1000
max_steps_per_episode = 100

for episode in range(num_episodes):
    state = env.reset()
    done = False
    total_reward = 0

    for step in range(max_steps_per_episode):
        # Exploration-exploitation trade-off
        if np.random.uniform(0, 1) < epsilon:
            action = env.action_space.sample() # Explore
        else:
            action = np.argmax(Q_table[state]) # Exploit

        new_state, reward, done, _ = env.step(action)

        # Update Q-table
```

The screenshot shows a Google Colab notebook titled "DS1.ipynb". The code implements a Q-learning algorithm with epsilon-greedy exploration:

```
if np.random.uniform(0, 1) < epsilon:
    action = env.action_space.sample() # Explore
else:
    action = np.argmax(Q_table[state]) # Exploit

new_state, reward, done, _ = env.step(action)

# Update Q-table
Q_table[state, action] = Q_table[state, action] + alpha * \
    (reward + gamma * np.max(Q_table[new_state])) - Q_table[state, action])

total_reward += reward
state = new_state

if done:
    break

if episode % 100 == 0:
    print(f"Episode {episode}, Total Reward: {total_reward}")
```

The output shows the total reward for each episode from 0 to 500:

```
Episode 0, Total Reward: -244
Episode 100, Total Reward: -118
Episode 200, Total Reward: -136
Episode 300, Total Reward: -109
Episode 400, Total Reward: -145
Episode 500, Total Reward: -64
```

The screenshot shows the same Google Colab notebook "DS1.ipynb". The code now includes a testing loop after the training loop:

```
if episode % 100 == 0:
    print(f"Episode {episode}, Total Reward: {total_reward}")

Episode 0, Total Reward: -244
Episode 100, Total Reward: -118
Episode 200, Total Reward: -136
Episode 300, Total Reward: -109
Episode 400, Total Reward: -145
Episode 500, Total Reward: -64
Episode 600, Total Reward: -73
Episode 700, Total Reward: -145
Episode 800, Total Reward: -93
Episode 900, Total Reward: -136

num_test_episodes = 10

for episode in range(num_test_episodes):
    state = env.reset()
    done = False
    total_reward = 0

    for step in range(max_steps_per_episode):
        action = np.argmax(Q_table[state])
        new_state, reward, done, _ = env.step(action)
        total_reward += reward
```

ML Pracs Manual.pdf - Google | DS1.ipynb - Colaboratory | ChatGPT

colab.research.google.com/drive/1ND4WJUH05zdUHhOhGzImWjpkvDP4UcE#scrollTo=nkmd2-B\_CSx4

DS1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 10:57 PM

+ Code + Text

```
▶ num_test_episodes = 10

for episode in range(num_test_episodes):
    state = env.reset()
    done = False
    total_reward = 0

    for step in range(max_steps_per_episode):
        action = np.argmax(q_table[state])
        new_state, reward, done, _ = env.step(action)
        total_reward += reward
        state = new_state

        if done:
            break

    print(f"Test Episode {episode + 1}, Total Reward: {total_reward}")
```

⌚ Test Episode 1, Total Reward: -100  
Test Episode 2, Total Reward: -100  
Test Episode 3, Total Reward: -100  
Test Episode 4, Total Reward: -100  
Test Episode 5, Total Reward: -100  
Test Episode 6, Total Reward: -100  
Test Episode 7, Total Reward: -100  
Test Episode 8, Total Reward: -100