

# RINEX

## MACHINE LEARNING

### MAJOR PROJECT: 1

### LINEAR AND LOGISTIC REGRESSION

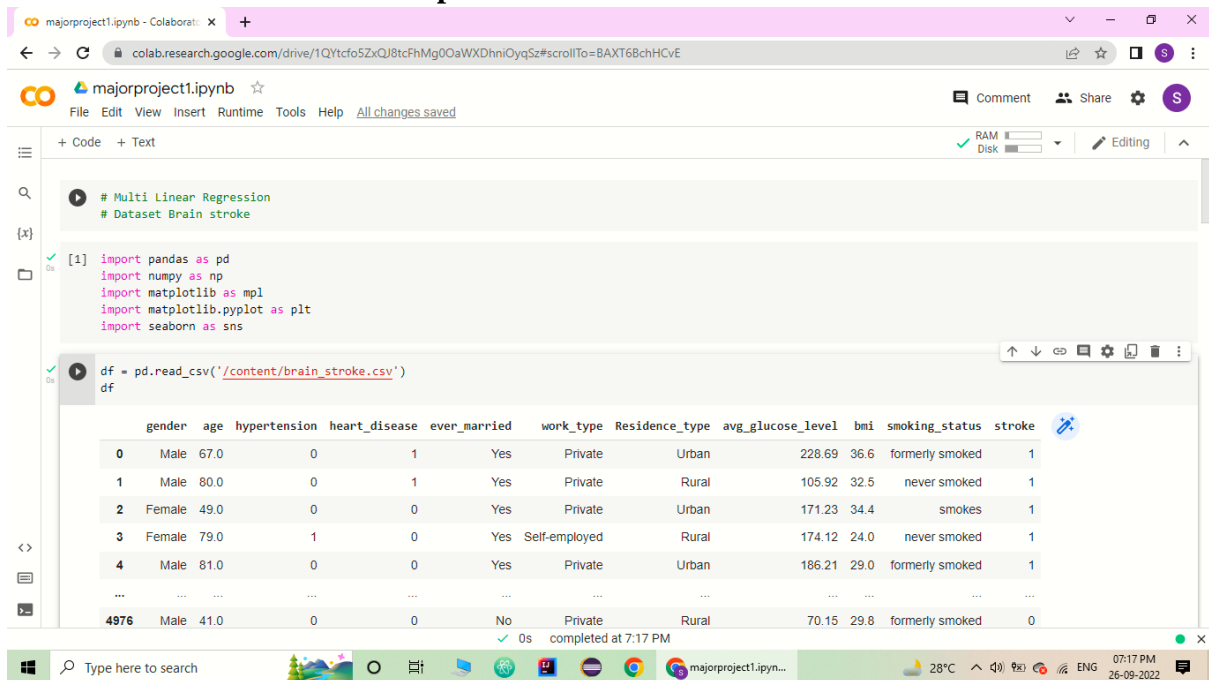
Harsh Singh

Thakur College of Engineering and Technology-(TCET)

B.E. Electronics

Third year-[TE]

#### Screenshots of code and the output:

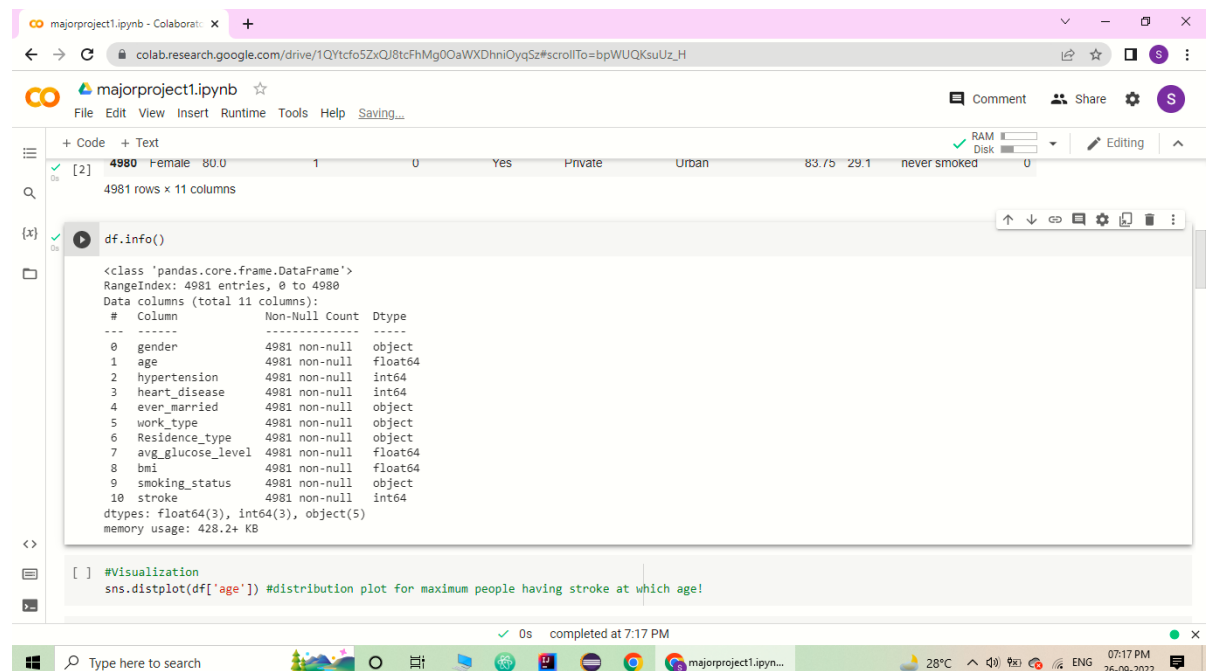


```
# Multi Linear Regression
# Dataset Brain stroke

[1] import pandas as pd
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('/content/brain_stroke.csv')
df
```

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
2	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
3	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1
4	Male	81.0	0	0	Yes	Private	Urban	186.21	29.0	formerly smoked	1
...	...	...	...	...	...	...	...	...	...	...	...
4976	Male	41.0	0	0	No	Private	Rural	70.15	29.8	formerly smoked	0



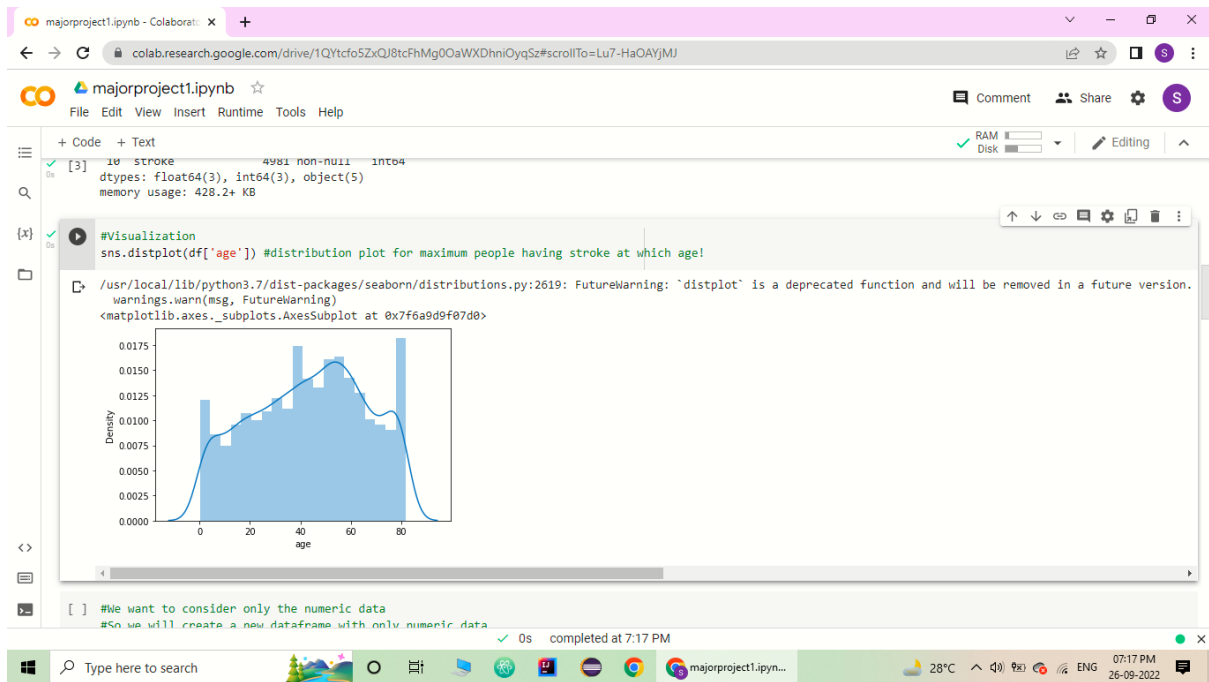
```
[2] 4980 Female 80.0 1 0 Yes Private Urban 83.75 29.1 never smoked 0
4981 rows x 11 columns

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4981 entries, 0 to 4980
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype  
---  --
 0   gender        4981 non-null   object  
 1   age           4981 non-null   float64  
 2   hypertension   4981 non-null   int64  
 3   heart_disease  4981 non-null   int64  
 4   ever_married   4981 non-null   object  
 5   work_type      4981 non-null   object  
 6   Residence_type  4981 non-null   object  
 7   avg_glucose_level  4981 non-null  float64  
 8   bmi           4981 non-null   float64  
 9   smoking_status  4981 non-null  object  
10   stroke        4981 non-null   int64  
dtypes: float64(3), int64(3), object(5)
memory usage: 428.2+ KB
```

```
[ ] #Visualization
sns.distplot(df['age']) #distribution plot for maximum people having stroke at which age!
```

# RINEX



majorproject1.ipynb - Collaborator: X

colab.research.google.com/drive/1QYtcf05ZxQJ8tcFhMg0OaWXDhniOyqSz#scrollTo=Y04mlb1harEA

majorproject1.ipynb

File Edit View Insert Runtime Tools Help

+ Code + Text

```
#We want to consider only the numeric data  
#So we will create a new dataframe with only numeric data  
df_numeric = df.select_dtypes(include = ['float64', 'int64'])  
df_numeric
```

	age	hypertension	heart_disease	avg_glucose_level	bmi	stroke
0	67.0	0	1	228.69	36.6	1
1	80.0	0	1	105.92	32.5	1
2	49.0	0	0	171.23	34.4	1
3	79.0	1	0	174.12	24.0	1
4	81.0	0	0	186.21	29.0	1
...	...	...	...	...	...	...
4976	41.0	0	0	70.15	29.8	0
4977	40.0	0	0	191.15	31.1	0
4978	45.0	1	0	95.02	31.8	0
4979	40.0	0	0	83.94	30.0	0
4980	80.0	1	0	83.75	29.1	0

4981 rows x 6 columns

completed at 7:18 PM

Type here to search

28°C

07:18 PM  
26-09-2022

# RINEX

majorproject1.ipynb - Collaborator

colab.research.google.com/drive/1QYtcf05ZxQJ8tcFhMg0OaWXDhniOyqSz#scrollTo=YgbOGPD3dV\_y

majorproject1.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
#I want to know the exact count of How many have got stroke and not got stroke.
#0 - No Stroke
#1 - Stroke
df['stroke'].value_counts()
```

```
0    4733
1     248
Name: stroke, dtype: int64
```

```
df_numeric.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4981 entries, 0 to 4980
Data columns (total 6 columns):
#   Column              Non-Null Count  Dtype  
---  -
0    age                 4981 non-null   float64
1    hypertension        4981 non-null   int64  
2    heart_disease       4981 non-null   int64  
3    avg_glucose_level   4981 non-null   float64
4    bmi                 4981 non-null   float64
5    stroke              4981 non-null   int64  
dtypes: float64(3), int64(3)
memory usage: 233.6 KB
```

```
[ ] #Divide the data into i/p and o/p
#Output: Age at which most of the people get stroke
```

0s completed at 7:18 PM

Type here to search

28°C

07:18 PM 26-09-2022

majorproject1.ipynb - Collaborator

colab.research.google.com/drive/1QYtcf05ZxQJ8tcFhMg0OaWXDhniOyqSz#scrollTo=SFCNpcbVeMcA

majorproject1.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
memory usage: 233.6 KB
```

```
[ ] #Divide the data into i/p and o/p
#Output: Age at which most of the people get stroke
#Input: All the columns except age column.
```

```
x = df_numeric.iloc[:,1:6].values
x
```

```
array([[ 0.,  1., 228.69, 36.6,  1. ],
       [ 0.,  1., 105.92, 32.5,  1. ],
       [ 0.,  0., 171.23, 34.4,  1. ],
       ...,
       [ 1.,  0.,  95.02, 31.8,  0. ],
       [ 0.,  0.,  83.94, 30.,  0. ],
       [ 1.,  0.,  83.75, 29.1,  0. ]])
```

```
[ ] y = df_numeric.iloc[:,0].values
y
```

```
array([67., 80., 49., ..., 45., 40., 80.])
```

```
[ ] #TRAIN and TEST VARIABLES
#sklearn.model_selection - package , train_test_split - library
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,random_state = 0)
```

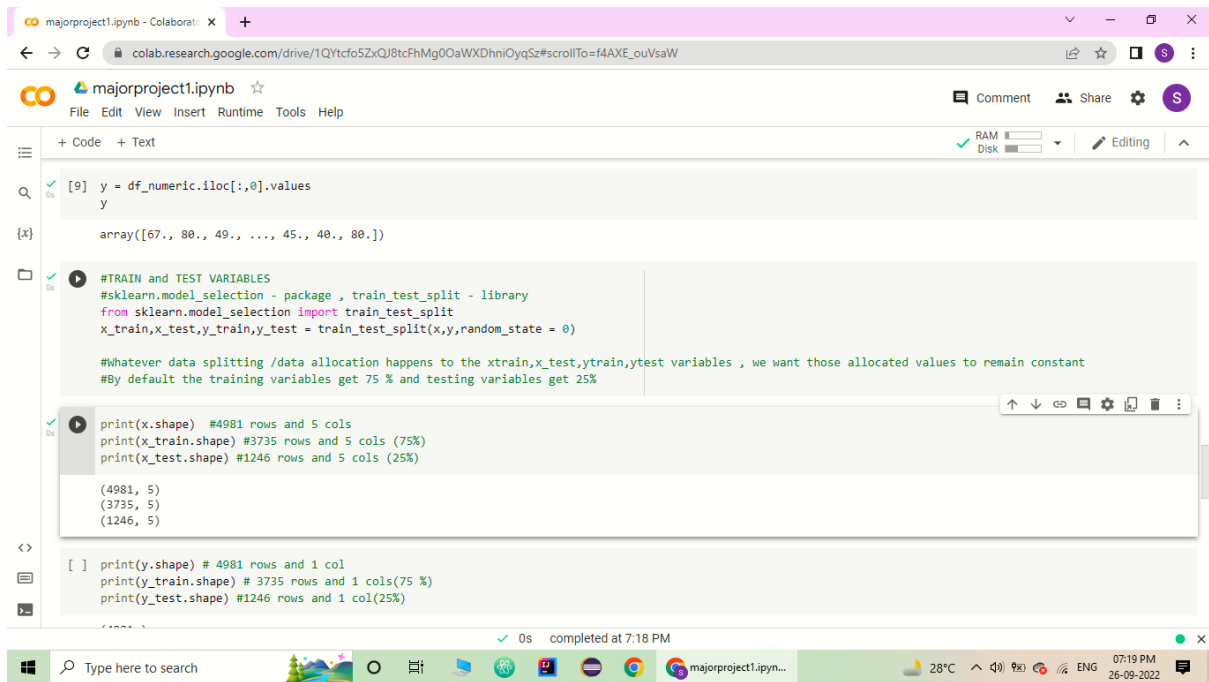
0s completed at 7:18 PM

Type here to search

28°C

07:18 PM 26-09-2022

# RINEX



The screenshot shows a Google Colab notebook titled 'majorproject1.ipynb'. The interface includes a top bar with the Colab logo, file name, and navigation icons. Below the top bar is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. The main workspace has a toolbar with '+ Code' and '+ Text' buttons. The code area contains the following Python code:

```
[9] y = df_numeric.iloc[:,0].values
y
array([67., 80., 49., ..., 45., 40., 80.])
```

```
#TRAIN and TEST VARIABLES
#sklearn.model_selection - package , train_test_split - library
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,random_state = 0)

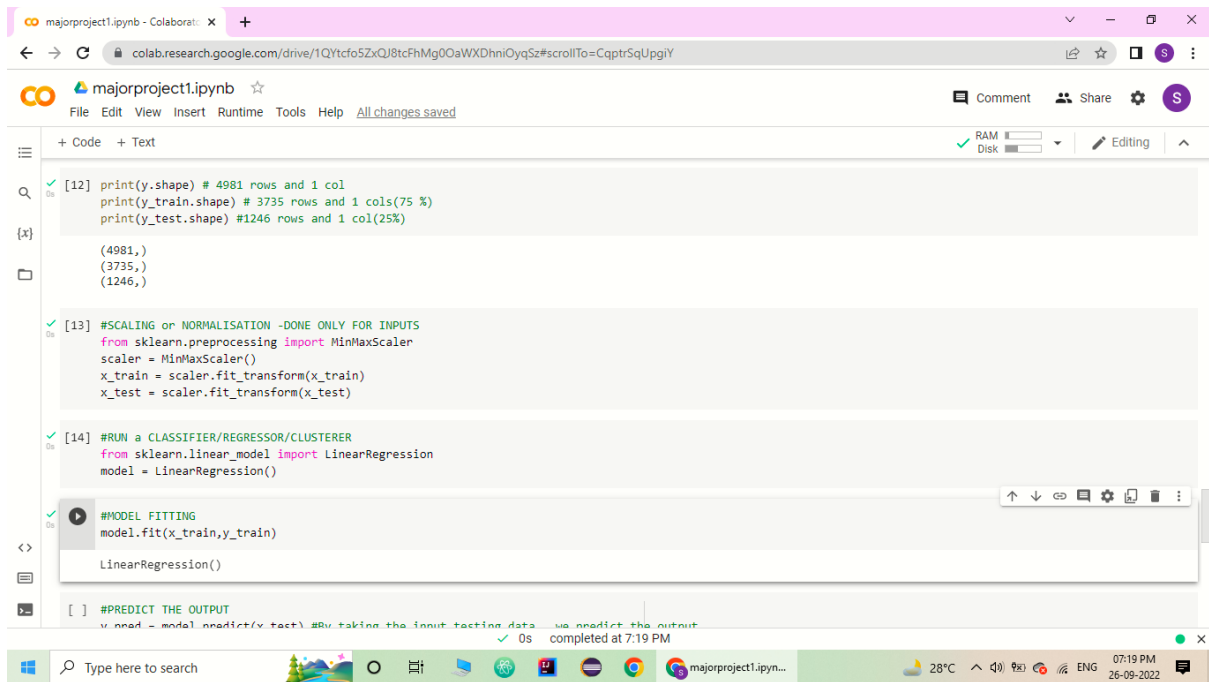
#Whatever data splitting /data allocation happens to the xtrain,x_test,ytrain,ytest variables , we want those allocated values to remain constant
#By default the training variables get 75 % and testing variables get 25%
```

```
print(x.shape) #4981 rows and 5 cols
print(x_train.shape) #3735 rows and 5 cols (75%)
print(x_test.shape) #1246 rows and 5 cols (25%)

(4981, 5)
(3735, 5)
(1246, 5)
```

```
[ ] print(y.shape) # 4981 rows and 1 col
print(y_train.shape) # 3735 rows and 1 cols(75 %)
print(y_test.shape) #1246 rows and 1 col(25%)
```

The output of the code is displayed below the code cells. The status bar at the bottom indicates '0s completed at 7:18 PM'.



The screenshot shows the same Google Colab notebook 'majorproject1.ipynb' at a later stage. The code area contains the following Python code:

```
[12] print(y.shape) # 4981 rows and 1 col
print(y_train.shape) # 3735 rows and 1 cols(75 %)
print(y_test.shape) #1246 rows and 1 col(25%)

(4981,)
(3735,)
(1246,)
```

```
[13] #SCALING or NORMALISATION -DONE ONLY FOR INPUTS
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.fit_transform(x_test)
```

```
[14] #RUN a CLASSIFIER/REGRESSOR/CLUSTERER
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

```
#MODEL FITTING
model.fit(x_train,y_train)

LinearRegression()
```

```
[ ] #PREDICT THE OUTPUT
y_pred = model.predict(x_test) #By taking the input testing data , we predict the output
```

The output of the code is displayed below the code cells. The status bar at the bottom indicates '0s completed at 7:19 PM'.

# RINEX

majorproject1.ipynb - Colaboratory

colab.research.google.com/drive/1QYtcf05ZxQJ8tcFhMg0OaWXDhniOyqSz#scrollTo=Etn9npkqO6x

majorproject1.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
[16] #PREDICT THE OUTPUT
y_pred = model.predict(x_test) #By taking the input testing data , we predict the output
y_pred #PREDICTED VALUES

array([[45.24391547, 51.49240425, 28.84289821, ..., 35.22385774,
        34.87637116, 28.49874727]])

[17] y_test #ACTUAL VALUES

array([40., 78., 14., ..., 80., 18., 2.])

[19] print(x_train[56]) #these are scaled/normalised values

[0.      0.      0.16600499 0.39255014 0.      ]

#INDIVIDUAL PREDICTION
model.predict([x_train[89]])

array([48.76754935])
```

0s completed at 7:20 PM

Type here to search

majorproject1.ipyn...

28°C

ENG 07:20 PM 26-09-2022

majorproject1.ipynb - Colaboratory

D10.ipynb - Colaboratory

BMI Dataset | Kaggle

RINEX5 - JULY - Google Drive

colab.research.google.com/drive/1QYtcf05ZxQJ8tcFhMg0OaWXDhniOyqSz#scrollTo=qEN7XFOWalbl

majorproject1.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
[22] # Logistic Regression
# Dataset -

[32] df1 = pd.read_csv('/content/bmi.csv')
df1
```

	Gender	Height	Weight	Index
0	Male	174	96	4
1	Male	189	87	2
2	Female	185	110	4
3	Female	195	104	3
4	Male	149	61	3
...	...	...	...	...
495	Female	150	153	5
496	Female	184	121	4
497	Female	141	136	5
498	Male	150	95	5
499	Male	173	131	5

0s completed at 9:04 PM

Type here to search

majorproject1.ipyn...

28°C

ENG 09:04 PM 26-09-2022

# RINEX

majorproject1.ipynb - Collaboratory | D10.ipynb - Collaboratory | BMI Dataset | Kaggle | RINEX5 - JULY - Google Drive | +

colab.research.google.com/drive/1QYtcf05ZxQJ8tcFhMg0OaWXDhniOyqSz#scrollTo=qEN7XFOWalbL

majorproject1.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
[30] df.shape
```

(4981, 11)

```
[31] df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype  
---  -
0   Gender      500 non-null    object  
1   Height      500 non-null    int64   
2   Weight      500 non-null    int64   
3   Index       500 non-null    int64   
dtypes: int64(3), object(1)
memory usage: 15.8+ KB
```

```
[ ] #Input - Height and Weight
#Output - Index
```

```
[35] #I want to know the exact count of male and female
df1["Gender"].value_counts()
```

```
Female    255
Male      245
```

0s completed at 9:04 PM

Type here to search

28°C

09:04 PM 26-09-2022

majorproject1.ipynb - Collaboratory | D10.ipynb - Collaboratory | BMI Dataset | Kaggle | RINEX5 - JULY - Google Drive | +

colab.research.google.com/drive/1QYtcf05ZxQJ8tcFhMg0OaWXDhniOyqSz#scrollTo=qEN7XFOWalbL

majorproject1.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

```
[ ] #Input - Height and Weight
#Output - Index
```

```
[35] #I want to know the exact count of male and female
df1["Gender"].value_counts()
```

```
Female    255
Male      245
Name: Gender, dtype: int64
```

```
[83] #divide the data into i/p and o/p
x1 = df1.iloc[:,1:3].values
x1
```

```
[143, 149],
[152, 146],
[186, 128],
[159, 140],
[146, 70],
[176, 121],
[146, 101],
[159, 145],
[162, 157],
[172, 90],
[169, 121],
[182, 50],
[183, 79],
[176, 77],
```

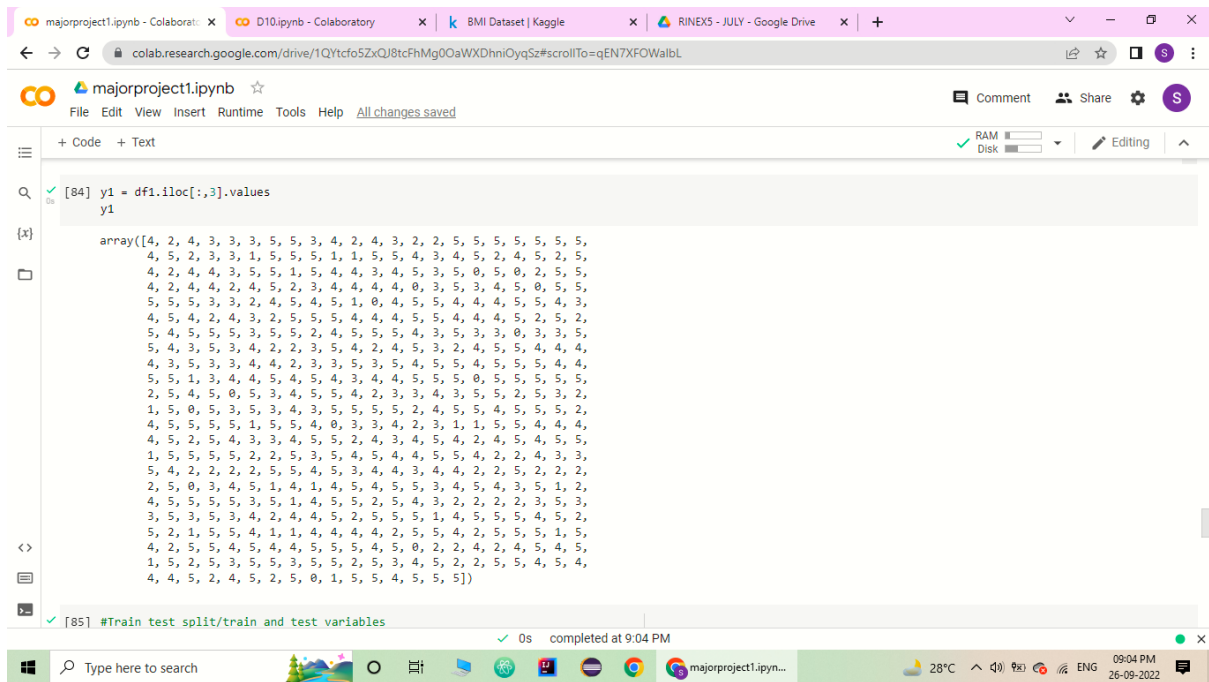
0s completed at 9:04 PM

Type here to search

28°C

09:04 PM 26-09-2022

# RINEX



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

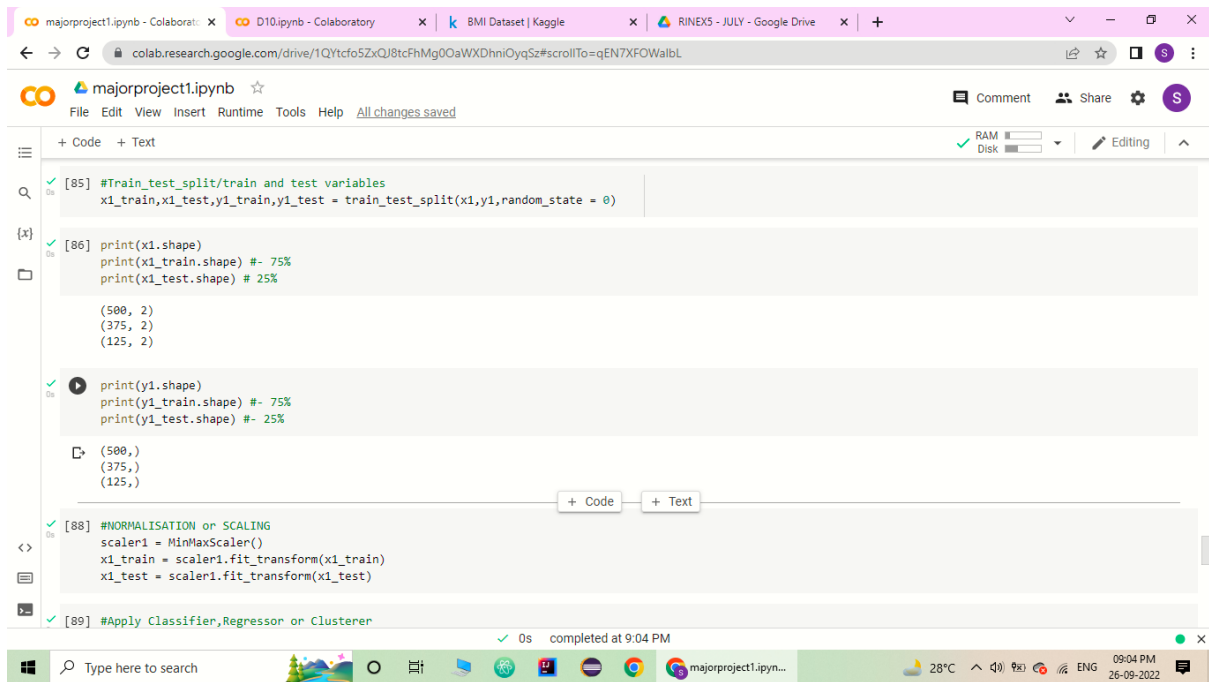
- Browser Tabs:** majorproject1.ipynb - Collaboratory, D10.ipynb - Collaboratory, BMI Dataset | Kaggle, RINEX5 - JULY - Google Drive.
- Address Bar:** colab.research.google.com/drive/1QYtcf0S2xQJ8tcFhMg00aWXDhniOyqSz#scrollTo=qEN7XFOWalbL
- Notebook Interface:** majorproject1.ipynb, File Edit View Insert Runtime Tools Help, All changes saved.
- Code Cell [84]:**

```
y1 = df1.iloc[:,3].values
y1
```

The output is a large array of integers, representing a single column of data from a DataFrame.
- Code Cell [85]:**

```
#Train test split/train and test variables
```

This cell is currently empty.
- Status Bar:** RAM 28°C, Disk, ENG, 09:04 PM 26-09-2022.



The screenshot shows the same Google Colab notebook with the following details:

- Code Cell [85]:**

```
#Train test split/train and test variables
x1_train,x1_test,y1_train,y1_test = train_test_split(x1,y1,random_state = 0)
```
- Code Cell [86]:**

```
print(x1.shape)
print(x1_train.shape) #- 75%
print(x1_test.shape) # 25%
```

The output shows the shapes of the data arrays:

```
(500, 2)
(375, 2)
(125, 2)
```
- Code Cell [87]:**

```
print(y1.shape)
print(y1_train.shape) #- 75%
print(y1_test.shape) #- 25%
```

The output shows the shapes of the target variable arrays:

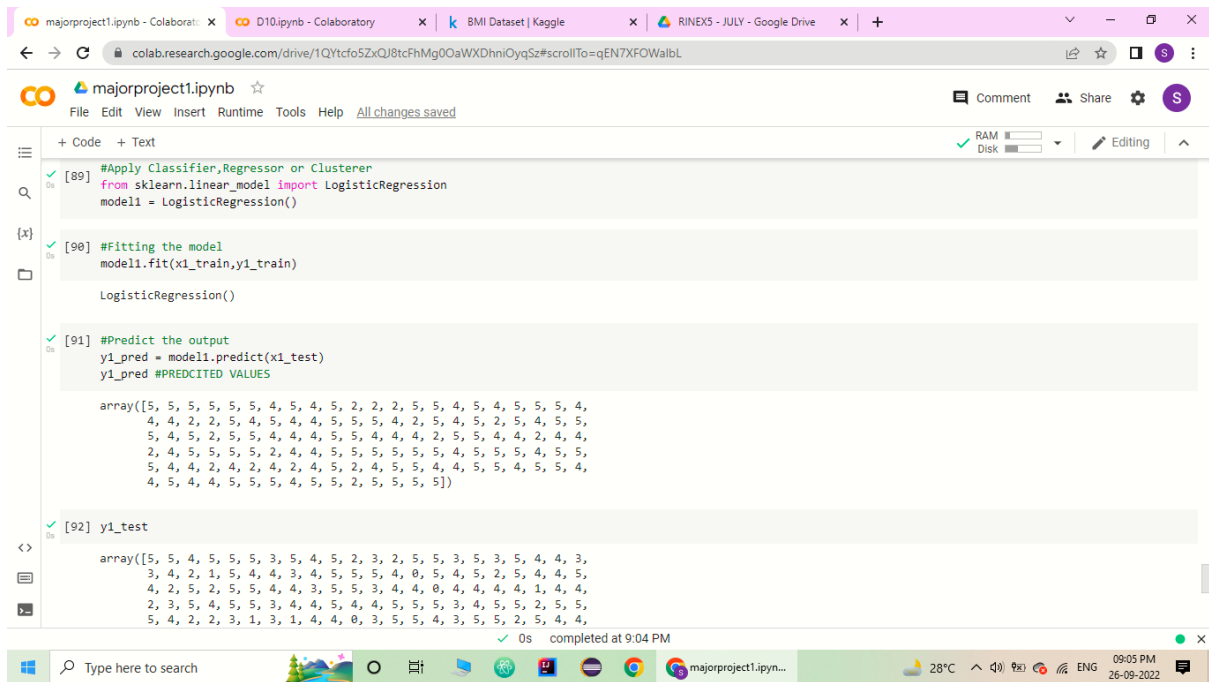
```
(500,)
(375,)
(125,)
```
- Code Cell [88]:**

```
#NORMALISATION or SCALING
scaler1 = MinMaxScaler()
x1_train = scaler1.fit_transform(x1_train)
x1_test = scaler1.fit_transform(x1_test)
```
- Code Cell [89]:**

```
#Apply Classifier,Regressor or Clusterer
```

This cell is currently empty.
- Status Bar:** RAM 28°C, Disk, ENG, 09:04 PM 26-09-2022.

# RINEX



```
[89] #Apply Classifier,Regressor or Clusterer
from sklearn.linear_model import LogisticRegression
model1 = LogisticRegression()

[90] #Fitting the model
model1.fit(x1_train,y1_train)

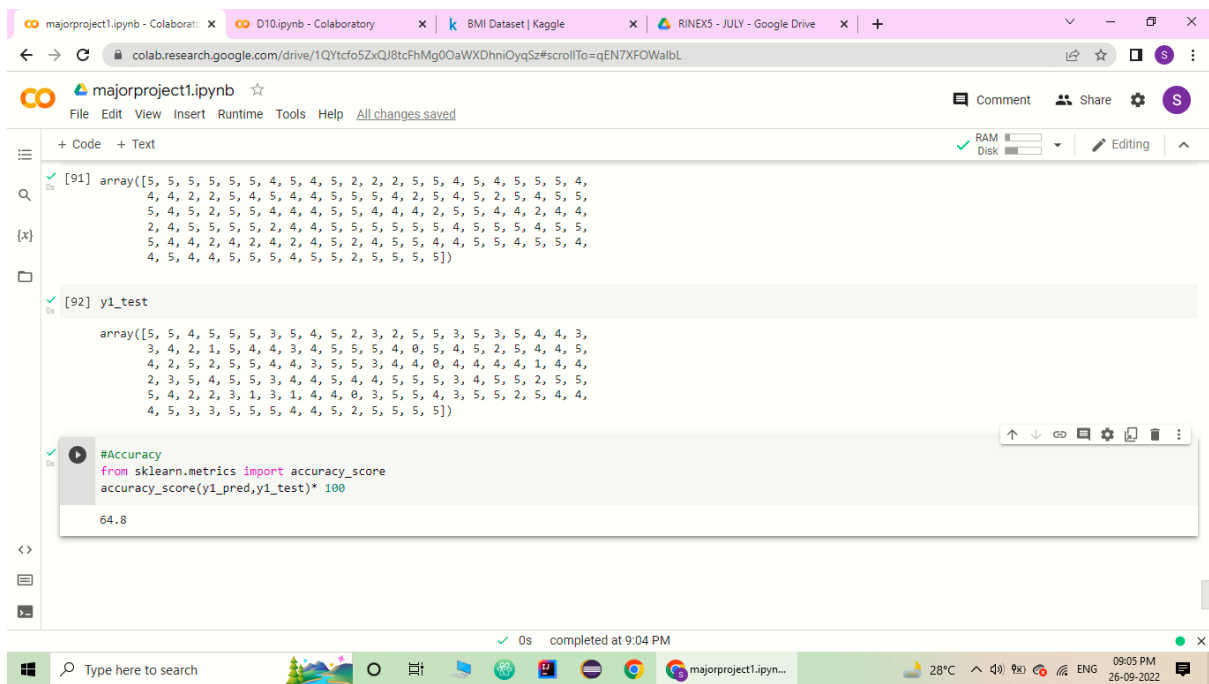
LogisticRegression()

[91] #Predict the output
y1_pred = model1.predict(x1_test)
y1_pred #PREDCITED VALUES

array([[5, 5, 5, 5, 5, 5, 4, 5, 4, 5, 2, 2, 2, 5, 5, 4, 5, 4, 5, 5, 5, 4,
        4, 4, 2, 2, 5, 4, 5, 4, 4, 5, 5, 5, 4, 2, 5, 4, 5, 2, 5, 4, 5, 5,
        5, 4, 5, 2, 5, 5, 4, 4, 4, 5, 5, 4, 4, 4, 2, 5, 5, 4, 4, 2, 4, 4,
        2, 4, 5, 5, 5, 5, 2, 4, 4, 5, 5, 5, 5, 5, 5, 4, 5, 5, 5, 4, 5, 5,
        5, 4, 4, 2, 4, 2, 4, 2, 4, 5, 2, 4, 5, 4, 4, 5, 5, 4, 5, 5, 4,
        4, 5, 4, 4, 5, 5, 5, 4, 5, 5, 2, 5, 5, 5, 5]])

[92] y1_test

array([[5, 5, 4, 5, 5, 5, 3, 5, 4, 5, 2, 3, 2, 5, 5, 3, 5, 3, 5, 4, 4, 3,
        3, 4, 2, 1, 5, 4, 4, 3, 4, 5, 5, 5, 4, 0, 5, 4, 5, 2, 5, 4, 4, 5,
        4, 2, 5, 2, 5, 5, 4, 4, 3, 5, 5, 3, 4, 4, 0, 4, 4, 4, 1, 4, 4,
        2, 3, 5, 4, 5, 5, 3, 4, 4, 5, 4, 4, 5, 5, 3, 4, 5, 2, 5, 5,
        5, 4, 2, 2, 3, 1, 3, 1, 4, 4, 0, 3, 5, 5, 4, 3, 5, 5, 2, 5, 4, 4,
        4, 5, 3, 3, 5, 5, 5, 4, 4, 5, 2, 5, 5, 5, 5]])
```



```
[91] array([[5, 5, 5, 5, 5, 5, 4, 5, 4, 5, 2, 2, 2, 5, 5, 4, 5, 4, 5, 5, 5, 4,
        4, 4, 2, 2, 5, 4, 5, 4, 4, 5, 5, 5, 4, 2, 5, 4, 5, 2, 5, 4, 5, 5,
        5, 4, 5, 2, 5, 5, 4, 4, 4, 5, 5, 4, 4, 4, 2, 5, 5, 4, 4, 2, 4, 4,
        2, 4, 5, 5, 5, 5, 2, 4, 4, 5, 5, 5, 5, 5, 4, 5, 5, 5, 4, 5, 5,
        5, 4, 4, 2, 4, 2, 4, 2, 4, 5, 2, 4, 5, 4, 4, 5, 5, 4, 5, 5, 4,
        4, 5, 4, 4, 5, 5, 5, 4, 5, 5, 2, 5, 5, 5, 5]])

[92] y1_test

array([[5, 5, 4, 5, 5, 5, 3, 5, 4, 5, 2, 3, 2, 5, 5, 3, 5, 3, 5, 4, 4, 3,
        3, 4, 2, 1, 5, 4, 4, 3, 4, 5, 5, 5, 4, 0, 5, 4, 5, 2, 5, 4, 4, 5,
        4, 2, 5, 2, 5, 5, 4, 4, 3, 5, 5, 3, 4, 4, 0, 4, 4, 4, 1, 4, 4,
        2, 3, 5, 4, 5, 5, 3, 4, 4, 5, 4, 4, 5, 5, 3, 4, 5, 2, 5, 5,
        5, 4, 2, 2, 3, 1, 3, 1, 4, 4, 0, 3, 5, 5, 4, 3, 5, 5, 2, 5, 4, 4,
        4, 5, 3, 3, 5, 5, 5, 4, 4, 5, 2, 5, 5, 5, 5]])

#Accuracy
from sklearn.metrics import accuracy_score
accuracy_score(y1_pred,y1_test)* 100

64.8
```

Google Drive Link: [https://drive.google.com/drive/folders/1wNawEyjY5Q-J\\_6Q1PRjcH5h9yjgOP\\_jz](https://drive.google.com/drive/folders/1wNawEyjY5Q-J_6Q1PRjcH5h9yjgOP_jz)