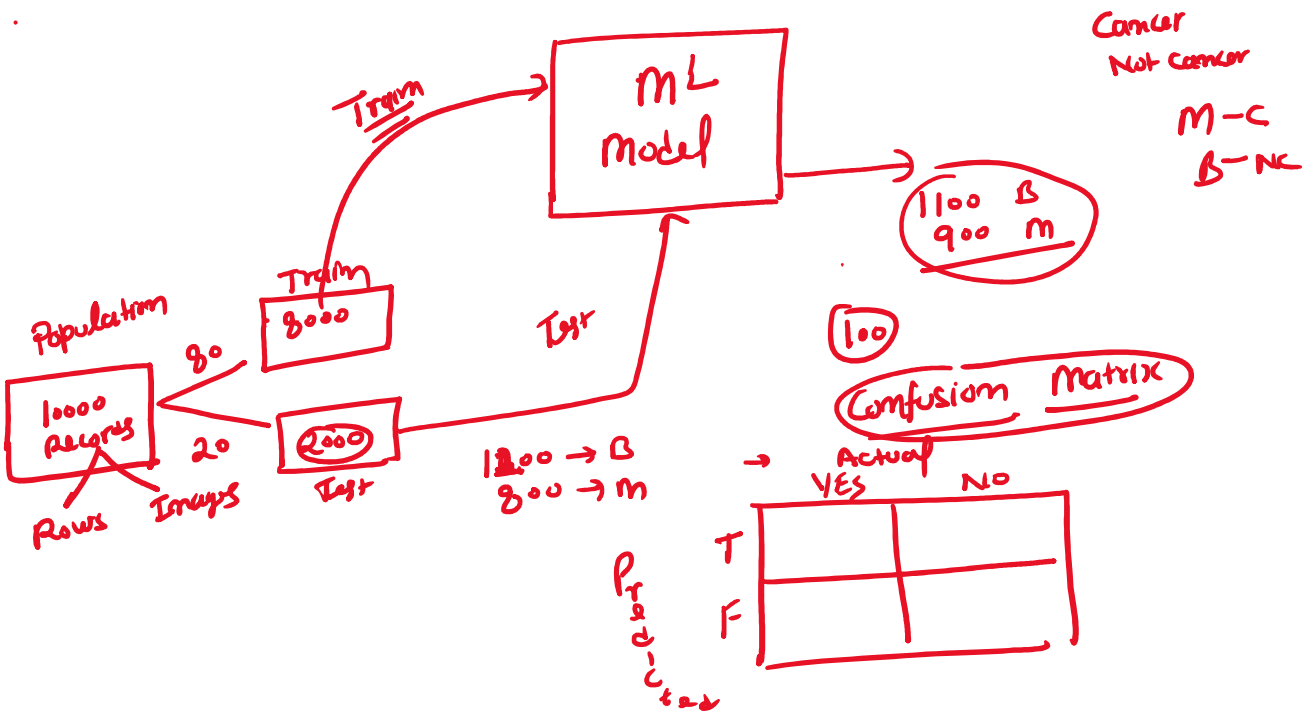
02 April 2024 21:19

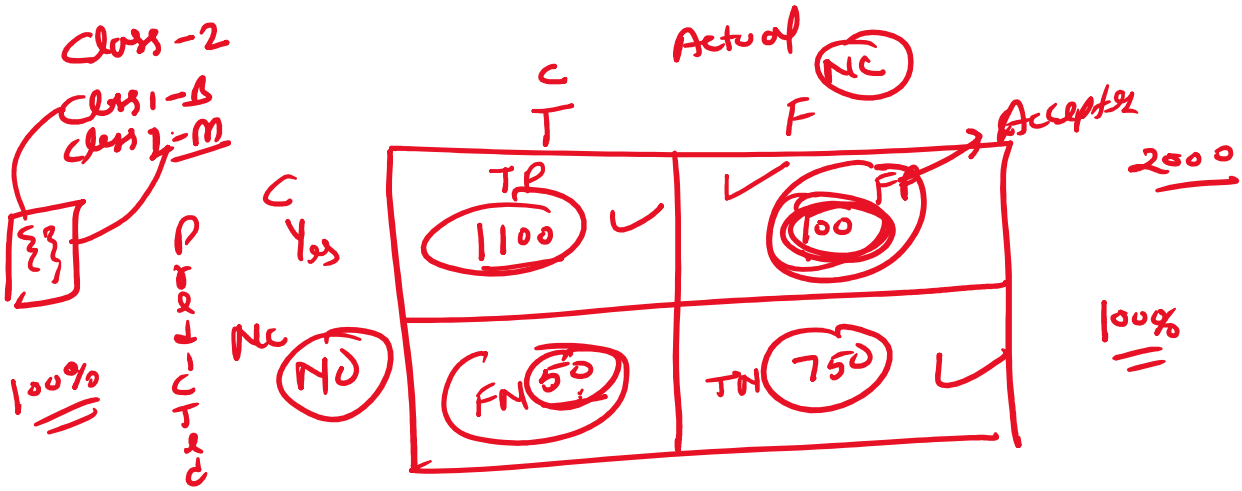
Day-2

Intro. to Machine Learning

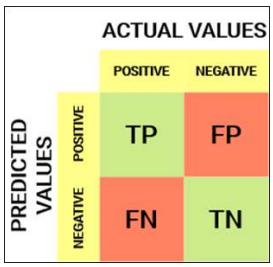
Python Module

Pydata





$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} = \frac{1100+750}{2000} = \frac{1850}{2000} = 92.5\%$$

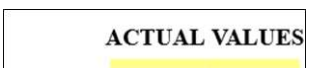


Medical Prob.

Low R

Medical

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} = \frac{560+330}{1000} = \frac{890}{1000} \times 100 = 89\%$$



	ACTUAL VALUES: POSITIVE	ACTUAL VALUES: NEGATIVE
PREDICTED VALUES: POSITIVE	TP 560	FP 60
PREDICTED VALUES: NEGATIVE	FN 50	TN 330

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Precision = \frac{TP}{TP + FP} = \frac{560}{560 + 60} = \frac{560}{620} = 90.3\% = 100\%$$

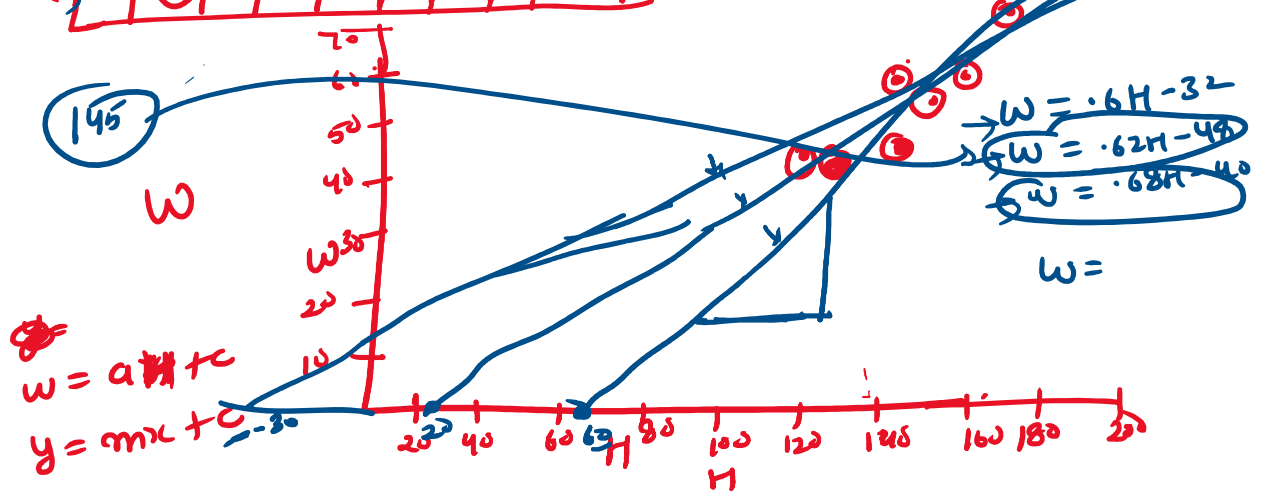
$$Recall = \frac{TP}{TP + FN} = \frac{560}{560 + 50} = \frac{560}{610} = 91.8\% = 100\%$$

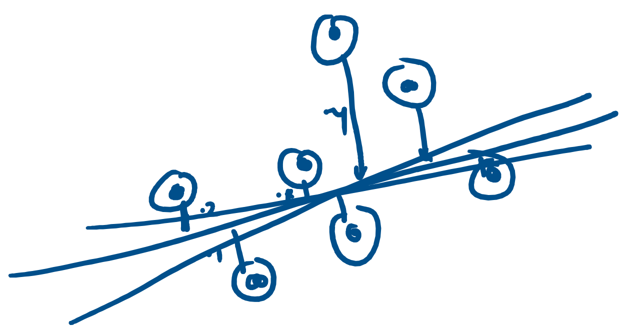
F1-Score =

Movie rating → Su, → Fa; High P, Low R

H → 2 Months; 1000 → 3000

H	140	150	135	160	169	175	180	190
W	45	50	40	55	57	62	42	90





$y = -1.75 \times x + 103.1$

$= -1.75 \times 5 + 103.1$

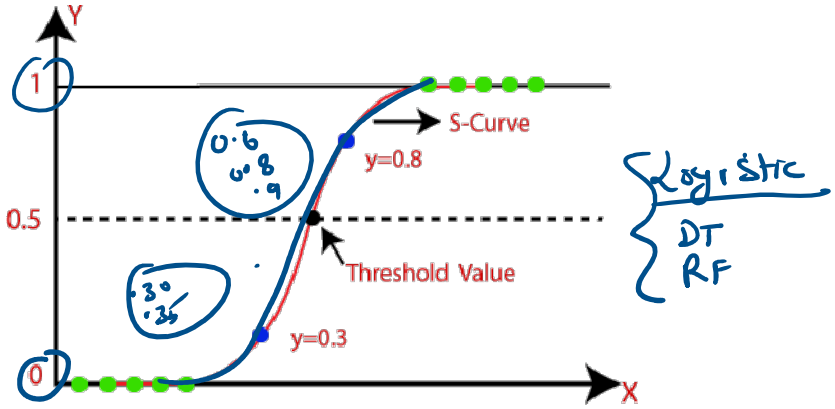
$= -8.75 + 103.1$

$= 94.35$

$= -1.75 \times 9 + 103.1$

$= -1.75128 + 103.105$

9



Example: There is a dataset given which contains the information of various users obtained from the social networking sites. There is a car making company that has recently launched a new SUV car. So the company wanted to check how many users from the dataset, wants to purchase the car.

For this problem, we will build a Machine Learning model using the Logistic regression algorithm. The dataset is shown in the below image. In this problem, we will predict the **purchased variable (Dependent Variable)** by using **age and salary (Independent variables)**.

Steps in Logistic Regression: To implement the Logistic Regression using Python, we will use the same steps as we have done in previous topics of Regression. Below are the steps:

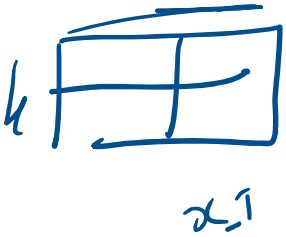
- Data Pre-processing step

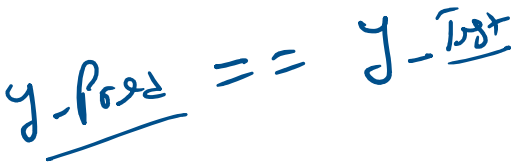
- Fitting Logistic Regression to the Training set
- Predicting the test result
- Test accuracy of the result(Creation of Confusion matrix)
- Visualizing the test set result.

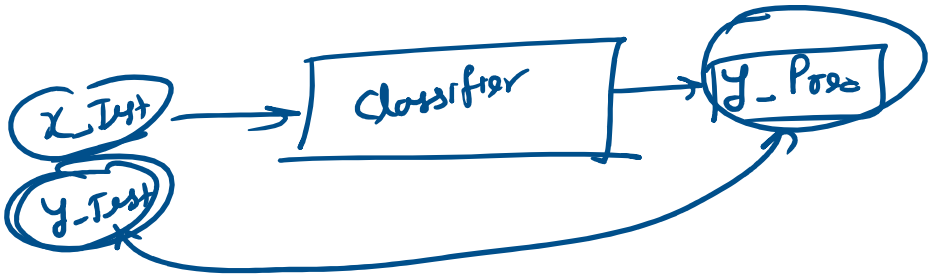
```
y([[   44,  39000],
   [   32, 120000],
   [   38,  50000],
   [   32, 135000],
   [   52,  21000],
   [   53, 104000],
   [   39,  42000],
   [   38,  61000],
```

```
[ 0.58164944, -0.88670699],
 [-0.60673761,  1.46173768],
 [-0.01254409, -0.5677824 ],
 [-0.60673761,  1.89663484],
 [ 1.37390747, -1.40858358],
```

From <http://localhost:8892/lab/tree/Fundamental/ML%20Algorithms/Logistic-Regression.ipynb>








```
array([[65,  3],
       [ 8, 24]], dtype=int64)
```

From <http://localhost:8892/lab/tree/Fundamental/ML%20Algorithms/Logistic-Regression.ipynb>

```
array([[ 0.58164944, -0.88670699],
       [-0.60673761,  1.46173768],
       [-0.01254409, -0.5677824 ],
```

[-0.60673761, 1.89663484],
[1.37390747, -1.40858358],
[1.47293972, 0.99784738],
[0.08648817, -0.79972756],

From <http://localhost:8892/lab/tree/Fundamental/ML%20Algorithms/Logistic-Regression.ipynb>

[[0.58164944, -0.88670699],
[-0.60673761, 1.46173768],
[-0.01254409, -0.5677824],
[-0.60673761, 1.89663484],
[1.37390747, -1.40858358],
[1.47293972, 0.99784738],
[0.08648817, -0.79972756],
[-0.01254409, -0.24885782],

From <http://localhost:8892/lab/tree/Fundamental/ML%20Algorithms/Logistic-Regression.ipynb>

Logistic Regression

```
[17]: cm

[17]: array([[65,  3],
            [ 8, 24]], dtype=int64)
```

User-Data

```
[18]: from sklearn.metrics import classification_report

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

	precision	recall	f1-score	support
0	0.89	0.96	0.92	68
1	0.89	0.75	0.81	32
accuracy			0.89	100
macro avg	0.89	0.85	0.87	100
weighted avg	0.89	0.89	0.89	100

99%

Train Test

Decision Tree

```
[20]: cm

[20]: array([[62,  6],
            [ 3, 29]], dtype=int64)
```

```
[21]: from sklearn.metrics import classification_report

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

	precision	recall	f1-score	support
0	0.95	0.91	0.93	68
1	0.83	0.91	0.87	32
accuracy			0.91	100
macro avg	0.89	0.91	0.90	100
weighted avg	0.91	0.91	0.91	100

91%

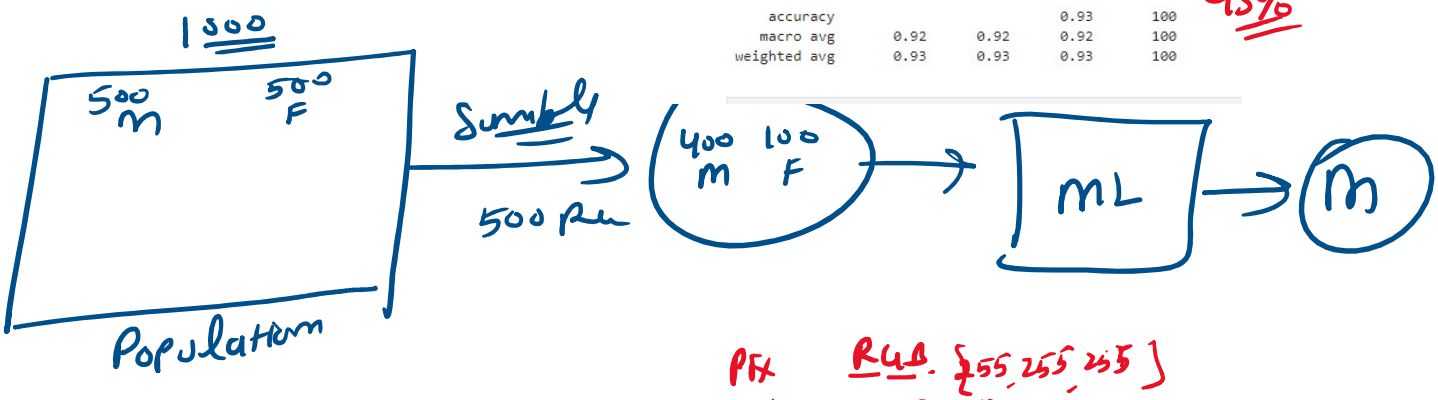
Random Forest

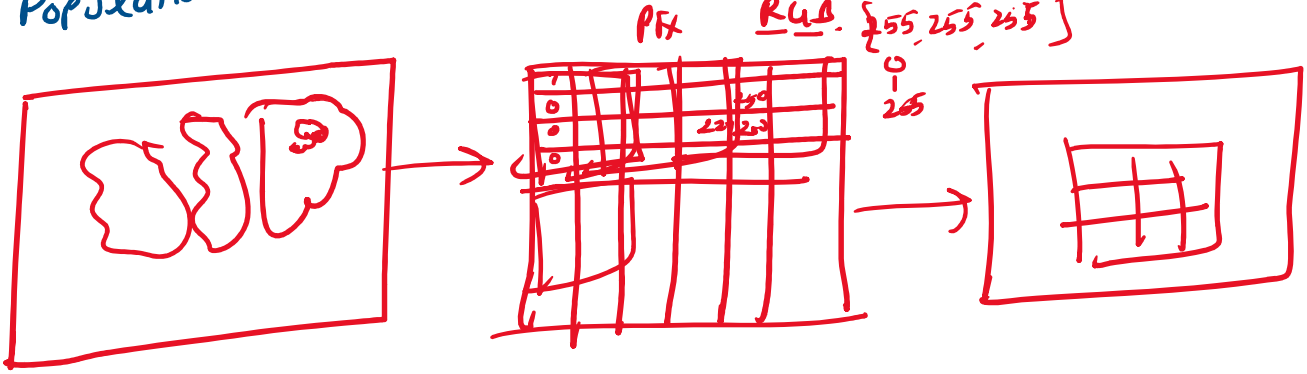
```
[11]: array([[64,  4],
            [ 3, 29]], dtype=int64)
```

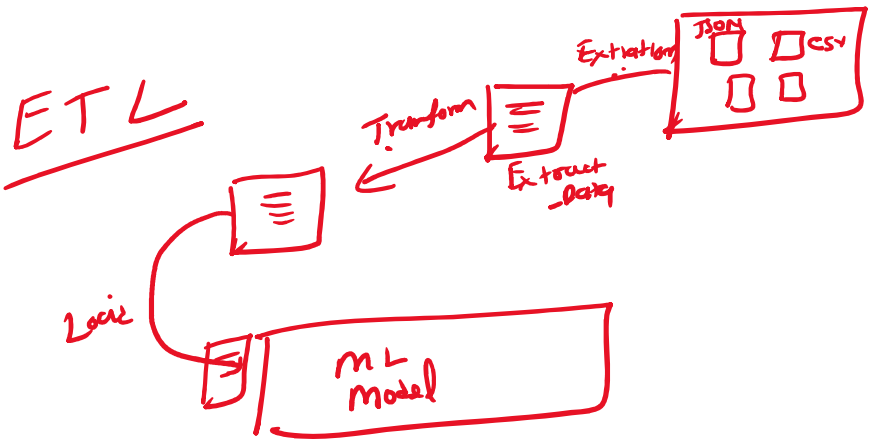
```
[12]: from sklearn.metrics import classification_report

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

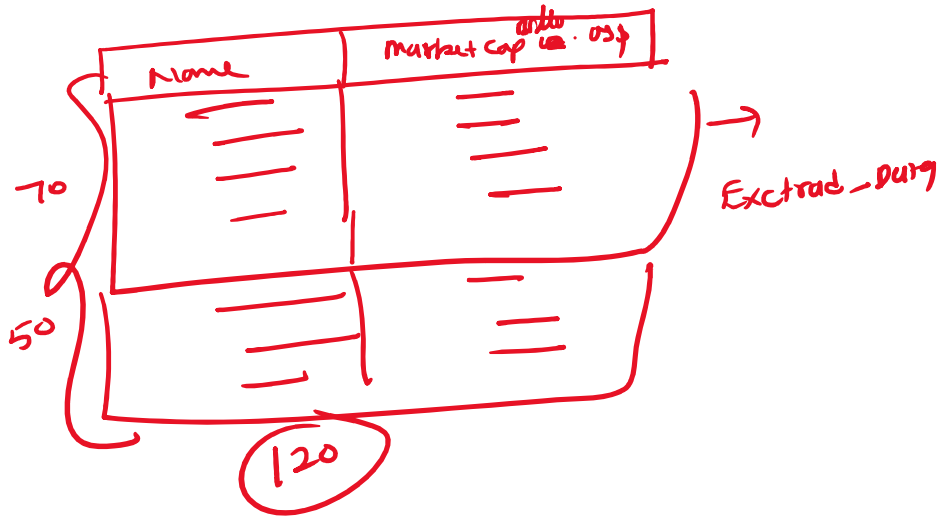
	precision	recall	f1-score	support
0	0.96	0.94	0.95	68
1	0.88	0.91	0.89	32
accuracy			0.93	100
macro avg	0.92	0.92	0.92	100
weighted avg	0.93	0.93	0.93	100

93%






ETL
Extraction
JSON
CSV
Transform
Extract_data
Load
ML Model


Name
Market cap
70
50
120
Exctrad_Data