PHASE 3

TITLE : “ AI BASED DIABETES PREDICTION SYSTEM”

In this phase, we are going to discuss about the dataset. The dataset is loaded from the Kaggle and stored in a csv file. Lets build the project by loading and preprocessing the dataset. Start building the prediction system by preparing the environment and implementing basic pre processing techniques.

Importing the Dependencies :

1.import numpy as np

- Numpy is primarily used for various **mathematical operations** and it is easy to working with **arrays**

2.import pandas as pd

- Pandas is used for **data analysis and machine learning tasks**. It supports working with tabular data like CSV.

Data Collection and Analysis :

#loading the diabetes dataset to a pandas DataFrame

diabetes_dataset = pd.read_csv('/content/diabetes.csv')

As already mentioned above the dataset is downloaded from the **Kaggle PIMA dataset** and loaded into a csv file. The above function is used to read the stored csv file.

printing the first 4 rows of the dataset

diabetes_dataset.head()

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Data Preprocessing :

number of rows and Columns in this dataset

diabetes_dataset.shape

(768, 9) : There are **768 rows and 9 columns.**

getting the statistical measures of the data

diabetes_dataset.describe()

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Out
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.00
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.34
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.47
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.00
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.00
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.00
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.00
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.00

diabetes_dataset['Outcome'].value_counts()

0 500

1 268

Name: Outcome, dtype: int64

0 --> Non-Diabetic

1 --> Diabetic

diabetes_dataset.groupby('Outcome').mean()

Outcome	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	3.298000	109.980000	68.184000	19.664000	68.792000	30.304200	0.429734	31.190000
1	4.865672	141.257463	70.824627	22.164179	100.335821	35.142537	0.550500	37.067164

separating the data and labels

X = diabetes_dataset.drop(columns = 'Outcome', axis=1)

Y = diabetes_dataset['Outcome']

```
print(X)
```

```
     Pregnancies  Glucose  BloodPressure  ...   BMI  DiabetesPedigreeFunction  Age
0              6      148             72  ...  33.6                     0.627   50
1              1       85             66  ...  26.6                     0.351   31
2              8      183             64  ...  23.3                     0.672   32
3              1       89             66  ...  28.1                     0.167   21
4              0      137             40  ...  43.1                     2.288   33
..           ...      ...            ...  ...   ...                       ...  ...
763           10      101             76  ...  32.9                     0.171   63
764            2      122             70  ...  36.8                     0.340   27
765            5      121             72  ...  26.2                     0.245   30
766            1      126             60  ...  30.1                     0.349   47
767            1       93             70  ...  30.4                     0.315   23

[768 rows x 8 columns]
```

```
print(Y)
```

```
0      1
1      0
2      1
3      0
4      1
      ..
763    0
764    0
765    0
766    1
767    0
Name: Outcome, Length: 768, dtype: int64
```

For further model selection and training the data set has been separated into two with x & y labels.

In Summary :

Thus the preprocessing for the PIMA dataset . In the following phases we need to standardize & train the dataset in order to build the prediction system.