IBM NAAN MUDHALVAN

ARTIFICIAL INTELLIGENCE-GROUP 2

PHASE – 5 : Project Documentation & Submission

TITLE: AI BASED DIABETES PREDICTION SYSTEM

PROBLEM STATEMENT:

The problem is to build an AI-powered diabetes prediction system that uses machine learning algorithms to analyze medical data and predict the likelihood of an individual developing diabetes. The system aims to provide early risk assessment and personalized preventive measures, allowing individuals to take proactive actions to manage their health.

EXPLANATION:

AN AI-based diabetes prediction system is a computer program or application that uses artificial intelligence techniques, such as machine learning algorithms, to analyze data related to an individual's health, lifestyle, and medical history in order to predict the likelihood of them developing diabetes in the future. This system can help identify individuals at higher risk of diabetes, allowing for early intervention and preventive measures to manage or mitigate the disease.

INTRODUCTION

An AI-powered diabetes prediction system is a cutting-edge technological solution that leverages the capabilities of artificial intelligence (AI) to assist in the early detection, management, and prevention of diabetes, a chronic metabolic disorder characterized by high blood sugar levels. This system combines advanced machine learning algorithms, data analytics, and health information to create a comprehensive tool for healthcare professionals, patients, and individuals at risk of developing diabetes.

Here is a detailed introduction to various aspects of an AI-powered diabetes prediction system:

- **Understanding DiabetES:**

  Diabetes is a prevalent and serious health condition that affects millions of people worldwide. There are two primary types of diabetes: Type 1, which is an autoimmune disease, and Type 2, which is often linked to lifestyle factors like poor diet and lack of exercise. Both types can lead to severe health complications if not managed effectively.

- **The Role of AI:**

  AI, particularly machine learning, plays a pivotal role in diabetes prediction systems. It can analyze vast amounts of data, detect subtle patterns, and generate predictive models, which can aid in identifying individuals at risk, predicting future diabetes cases, and improving overall patient care.

- **Data Sources:**

  These systems utilize diverse sources of data, including electronic health records, patient demographics, medical histories, genetic information, lifestyle data, and clinical measurements like blood glucose levels and body mass index. The more comprehensive and high-quality the data, the more accurate the predictions can be.

- **Early Detection:**

  One of the primary benefits of an AI-powered diabetes prediction system is its ability to identify individuals at risk before they exhibit obvious symptoms. This early detection enables healthcare providers to intervene with preventive easures and lifestyle modifications, potentially delaying or even preventing the onset of diabetes.

- **Personalized Care:**

  AI can create personalized care plans for individuals with diabetes based on their specific health data. This may include dietary recommendations, exercise routines, medication adjustments, and regular monitoring to help manage blood sugar levels effectively.

- **Risk Assessment:**

These systems assess an individual's risk of developing diabetes by considering various factors such as family history, age, weight, physical activity, and dietary habits. The algorithms can provide a risk score and recommendations for lifestyle changes.

- **Continuous Monitoring:**

  For those already diagnosed with diabetes, AI-powered systems can continuously monitor glucose levels and other relevant parameters, alerting patients and healthcare providers to deviations from target values. This continuous monitoring helps in maintaining better control and reducing the risk of complications.

- **Clinical Decision Support:**

Healthcare providers can benefit from AI-driven decision support tools that assist in treatment planning and medication management. These systems can recommend optimal drug regimens and dosages based on individual patient responses and historical data.

- **Research and Public Health:**

  AI-powered diabetes prediction systems can also contribute to public health initiatives by analyzing population-level data to identify trends and risk factors. This information can inform policy decisions and public health campaigns.

- **Challenges and Ethical Considerations:**

There are challenges, including data privacy concerns, model interpretability, and the potential for algorithmic bias. Safeguarding sensitive health data and ensuring equitable access to the technology are critical considerations.

DESIGN THINKING

Design thinking is a user-centered, iterative problem-solving approach that focuses on understanding the needs and perspectives of users to develop innovative solutions. When applying design thinking to the development of an AI-based diabetes prediction system, you can follow a structured process that involves several key phases. Here are the phases of development for an AI-based diabetes prediction system using design thinking:

- **Empathize:**

In this phase, you need to understand the needs and challenges of the target users, including individuals at risk of diabetes, healthcare professionals, and researchers. Conduct interviews, surveys, and observations to gain insights into their experiences and expectations.

- **Define:**

Based on the information gathered in the empathy phase, clearly define the problem or opportunity. For an AI-based diabetes prediction system, this could be understanding the gaps in early detection, monitoring, or patient engagement in diabetes management.

- **Ideate:**

Brainstorm potential solutions and innovative features for the diabetes prediction system. Encourage creative thinking and generate a wide range of ideas. Consider how AI can address the defined problem, such as predictive odelling, data collection, or patient interaction.

- **Prototype:**

Create a preliminary prototype or mockup of the AI-based system. This could be a simplified version of the user interface or a prototype of the predictive algorithm. The goal is to visualize and test the proposed solution.

- **Test:**

Test the prototype with real users to gather feedback and refine the design. Understand how users interact with the system, what works well, and what needs improvement. This phase may involve multiple iterations to fine-tune the system.

- **Develop:**

Once the prototype is validated, move into the development phase. This involves building the actual AI-based diabetes prediction system, including software development, algorithm implementation, and integration with relevant data sources.

- **Test and Iterate:**

Continuously test and iterate the system as it is being developed. This iterative process allows you to refine the system's accuracy, user-friendliness, and performance. Gather feedback from stakeholders and make necessary adjustments.

- **Launch:**

After thorough testing and refinement, launch the AI-based diabetes prediction system. Ensure it is accessible to the intended users and that they are well-informed about its capabilities and benefits.

- **Monitor and Improve:**

Implement monitoring mechanisms to track the system's performance in real-world scenarios. Continuously collect data, analyze user interactions, and make improvements based on user feedback and emerging trends in diabetes management and prediction.

- **Scale and Expand:**

As the system gains acceptance and proves its effectiveness, consider opportunities to scale its use to a larger audience. This may involve partnerships with healthcare providers, integration with electronic health records, or collaboration with public health organizations.

Throughout the development process, it's crucial to keep the user experience and the needs of individuals at risk of diabetes at the forefront. Additionally, consider ethical and privacy concerns, ensuring that the system complies with data protection regulations and maintains user trust.

Design thinking, with its user-centric and iterative approach, can help create a more effective and user-friendly AI-based diabetes prediction system that truly addresses the needs of its users while improving diabetes management and prevention.

INNOVATION

LIBRARY PACKAGES:

Python:

• Python is a versatile and widely-used programming language for AI and data science.

• It provides a large ecosystem of libraries and tools that are essential for developing AI based systems.

• Python is the most commonly used programming language for AI and machine learning. You can download Python from the official website: https://www.python.org/downloads/

1.NumPy (Numerical Python):

• NumPy is a fundamental library for numerical computations in Python.

• It provides support for multi-dimensional arrays and matrices, along with Mathematical functions to operate on these arrays.

• NumPy is crucial for handling and manipulating data efficiently in AI projects.

• To install NumPy using pip (Python package manager), open your command prompt or terminal and run: pip install numpy

2.Pandas:

• Pandas is a data manipulation and analysis library.

• It offers data structures like DataFrames and Series, making it easy to handle and analyze structured data.

• Pandas is particularly useful for preprocessing and exploring datasets in AI projects.

• To install Pandas using pip, run: pip install pandas.

3.Scikit-learn:

• Scikit-learn is a machine learning library that provides simple and efficient tools for data analysis and modeling.

• It includes various algorithms for classification, regression, clustering, and more.

• Scikit-learn is well-suited for building predictive models in diabetes prediction systems.

• To install Scikit-learn using pip, run: pip install scikit-learn.

4.TensorFlow and PyTorch:

• TensorFlow and PyTorch are deep learning frameworks.

• They enable the creation of neural networks and deep learning models.

• These libraries offer tools for training and deploying machine learning and deep learning models, which are essential for advanced AI-based predictions.

• To install TensorFlow, you can use pip: pip install tensorflow.

• To install PyTorch, visit the official website for installation instructions: https://pytorch.org/get-started/locally/

5.Jupyter Notebook:

• Jupyter Notebook is an interactive development environment for data science and machine learning. • It allows you to write and execute code in a notebook-style format, making it easy to document and share your AI experiments and analyses.

• To install Jupyter Notebook using pip, run: pip install jupyter

6.Matplotlib and Seaborn:

• Matplotlib is a data visualization library that provides tools for creating static, animated, or interactive plots and charts.

• Seaborn is a higher-level data visualization library that simplifies the process of creating attractive and informative statistical graphics.

• To install Matplotlib using pip, run: pip install matplotlib

• To install Seaborn using pip, run: pip install seaborn

7.SciPy (Scientific Python):

• SciPy is a library built on top of NumPy and provides additional scientific and technical computing functionality.

• It includes modules for optimization, integration, interpolation, and statistical operations, which can be valuable in AI-based research

• To install Scipy using pip, run: pip install scipy

HOW TO TRAIN AND TEST:

1.Data Collection

• Gather a dataset that includes relevant features (such as age, BMI, family history, glucose levels, etc.) and the target variable, which indicates whether individuals have diabetes (e.g., binary classification: 0 for no diabetes, 1 for diabetes).

- Ensure the dataset is representative and of sufficient size to build a reliable model.

2.Data Preprocessing:

- Clean the data by handling missing values, outliers, and inconsistencies.
- Normalize or scale features to bring them to a common scale, which can improve model performance.
- Encode categorical variables (if any) into numerical values using techniques like one-hot encoding.

3.Data Splitting:

- Split the dataset into two subsets: a training set and a testing set.
- Typically, you might use 70-80% of the data for training and the remaining 20-30% for testing.
- Alternatively, you can use techniques like cross-validation for more robust model evaluation.

4.Model Selection:

- Choose a suitable machine learning or deep learning model for the task. Common choices include logistic regression, decision trees, random forests, support vector machines, neural networks, etc
- Selecting the right model may involve experimentation and evaluation of multiple algorithms.

5.Model Training:

- Train the selected model using the training data.
- The model learns to identify patterns and relationships between the input features and the target variable during this phase.

6.Model Evaluation:

- Evaluate the trained model's performance using the testing dataset.
- Common evaluation metrics for classification tasks include accuracy, precision, recall, F1-score, and ROC-AUC.
- Interpret the evaluation results to assess how well the model predicts diabetes.

7.Hyperparameter Tuning:

- Fine-tune the model's hyperparameters to optimize its performance.
- Techniques like grid search or random search can help you find the best combination of hyperparameters.

8.Validation and Cross-Validation:

- Perform cross-validation (e.g., k-fold cross-validation) to assess the model's generalization performance and ensure it's not overfitting to the training data.

9.Model Deployment:

- Once satisfied with the model's performance, deploy it in a real-world environment where it can make predictions on new, unseen data.

10.Monitoring and Maintenance:

- Continuously monitor the model's performance in production and retrain it periodically with fresh data to keep it accurate and up-to-date.

REAL WORLD CONCEPTS:

1.Early Detection and Prevention:

- One of the primary goals of AI-based diabetes prediction systems is to detect diabetes or prediabetes at an early stage. Early detection allows for timely intervention and lifestyle modifications to prevent or manage the condition effectively.

2.Risk Assessment:

- These systems assess an individual's risk of developing diabetes based on various factors, such as genetics, medical history, lifestyle, and physiological measurements like blood glucose levels, insulin resistance, and BMI (Body Mass Index).

3.Personalized Medicine:

- AI models can provide personalized recommendations for individuals at risk or already diagnosed with diabetes. These recommendations may include dietary plans, exercise routines, medication management, and glucose monitoring strategies tailored to each patient's unique needs.

4.Remote Monitoring:

- AI-enabled diabetes prediction systems can facilitate remote monitoring of patients, enabling healthcare providers to track glucose levels and other relevant health metrics in real-time. This can lead to better disease management and reduced hospital visits.

5.Patient Empowerment:

- Patients can gain valuable insights into their health and lifestyle choices through AI generated predictions and recommendations. This empowerment can motivate individuals to make healthier choices and actively participate in their own health.

6. Clinical Decision Support: -

- Healthcare professionals can benefit from AI-based systems as decision support tools. These systems can assist in diagnosing diabetes, determining treatment plans, and predicting patient outcomes, ultimately improving clinical decision-making.

7. Research and Drug Development: -

- AI can aid researchers in identifying patterns and correlations in large datasets, potentially leading to new discoveries related to diabetes causes, treatments, and drug development.

8. Chronic Disease Management: -

- Beyond prediction, AI can help in the ongoing management of diabetes by providing continuous monitoring and adjustment of treatment plans based on real-time data.

9. Population Health Management: -

- Healthcare organizations and policymakers can use AI-based predictions to identify high risk populations and allocate resources for diabetes prevention and management programs more effectively.

10. Ethical and Privacy Considerations: -

- Collecting and analyzing sensitive health data for AI predictions raises ethical and privacy concerns. Ensuring the security and privacy of patient data is a critical consideration in these systems.

11.Regulatory Compliance: -

- Healthcare AI systems, including diabetes prediction models, must adhere to regulatory standards such as HIPAA (Health Insurance Portability and Accountability Act) in the United States to protect patient information.

12. Interdisciplinary Collaboration: -

- The development and deployment of AI-based diabetes prediction systems often require collaboration between healthcare professionals, data scientists, software engineers, and legal experts to ensure accuracy, reliability, and compliance

13.Continual Improvement:

- AI models for diabetes prediction should be continually updated and improved as new data becomes available. Regular validation and retraining are essential to maintain their accuracy over time.

DATASET DETAILS:

Context:

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective is to predict based on diagnostic measurements whether a patient has diabetes.

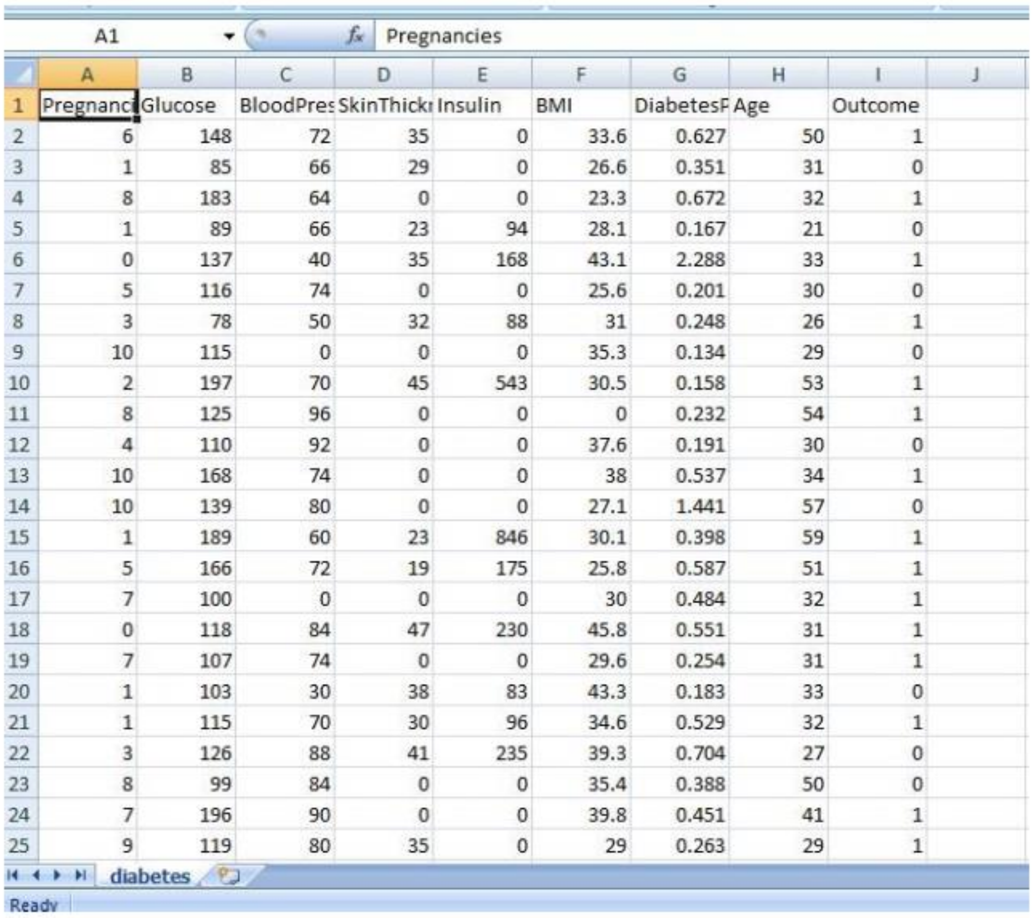
Content:

Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

- ✓ **Pregnancies:** Number of times pregnant
- ✓ **Glucose:** Plasma glucose concentration a 2 hours in an oral glucose tolerance test
- ✓ **BloodPressure:** Diastolic blood pressure (mm Hg)
- ✓ **SkinThickness:** Triceps skin fold thickness (mm)
- ✓ **Insulin:** 2-Hour serum insulin (mu U/ml)
- ✓ **BMI:** Body mass index (weight in kg/(height in m)^2)
- ✓ **DiabetesPedigreeFunction:** Diabetes pedigree function
- ✓ **Age:** Age (years)
- ✓ **Outcome:** Class variable (0 or 1)

DATASET LINK: https://www.kaggle.com/datasets/mathchi/diabetes-data-set

GIVEN DATA SET:



A1 fx Pregnancies

	A	B	C	D	E	F	G	H	I
1	Pregnanc	Glucose	BloodPres	SkinThickr	Insulin	BMI	DiabetesF	Age	Outcome
2	6	148	72	35	0	33.6	0.627	50	1
3	1	85	66	29	0	26.6	0.351	31	0
4	8	183	64	0	0	23.3	0.672	32	1
5	1	89	66	23	94	28.1	0.167	21	0
6	0	137	40	35	168	43.1	2.288	33	1
7	5	116	74	0	0	25.6	0.201	30	0
8	3	78	50	32	88	31	0.248	26	1
9	10	115	0	0	0	35.3	0.134	29	0
10	2	197	70	45	543	30.5	0.158	53	1
11	8	125	96	0	0	0	0.232	54	1
12	4	110	92	0	0	37.6	0.191	30	0
13	10	168	74	0	0	38	0.537	34	1
14	10	139	80	0	0	27.1	1.441	57	0
15	1	189	60	23	846	30.1	0.398	59	1
16	5	166	72	19	175	25.8	0.587	51	1
17	7	100	0	0	0	30	0.484	32	1
18	0	118	84	47	230	45.8	0.551	31	1
19	7	107	74	0	0	29.6	0.254	31	1
20	1	103	30	38	83	43.3	0.183	33	0
21	1	115	70	30	96	34.6	0.529	32	1
22	3	126	88	41	235	39.3	0.704	27	0
23	8	99	84	0	0	35.4	0.388	50	0
24	7	196	90	0	0	39.8	0.451	41	1
25	9	119	80	35	0	29	0.263	29	1

diabetes

Ready

Necessary step to follow:

1.Import Libraries:

Start by importing the necessary libraries:

PROGRAM:

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

sns.set()
```

```
from pandas.plotting import scatter_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import classification_report
import warnings
```

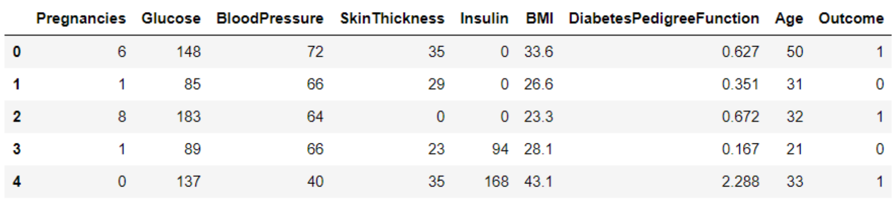
2.LOAD THE DATA

Load your dataset into a Pandas DataFrame. You can typically find diabetes datasets in CSV format we can see the output below.

PROGRAM:

```
# loading the diabetes dataset to a pandas DataFrame
diabetes_dataset = pd.read_csv('diabetes.csv')
diabetes_dataset.head()
```

Output:



```
   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
```

Exploratory Data Analysis (EDA)

Now let' see that what are columns available in our dataset.

PROGRAM:

```
diabetes_dataset.columns
```

Output:

```
Index(['Pregnancies', 'Glucose', 'BloodPressure',
'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age',
'Outcome'],
      dtype='object')
```

Information about the dataset

PROGRAM:

```
diabetes_dataset.info()
```

Output:

```
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

PROGRAM:

```
    diabetes_dataset.shape
```

OUTPUT:

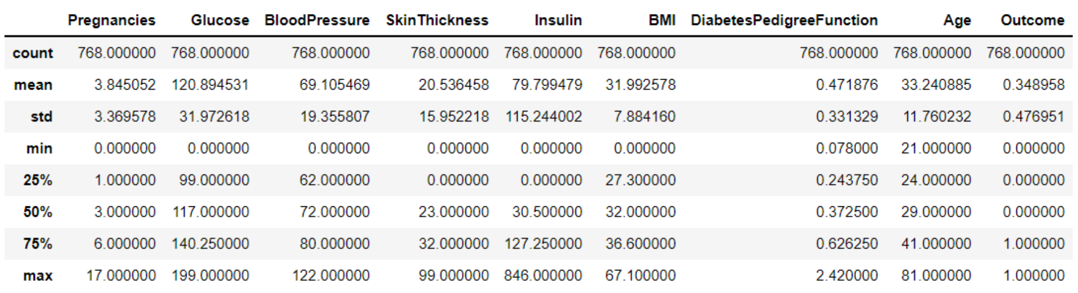
```
(768, 9)
```

To know more about the dataset

PROGRAM:

```
    diabetes_dataset.describe()
```

Output:



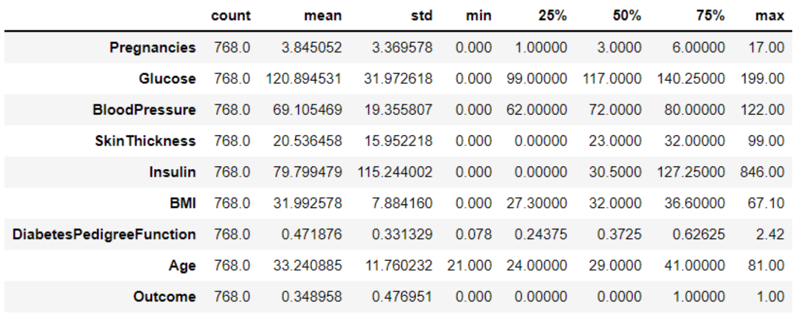
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

To know more about the dataset with transpose – here T is for the transpose

Program:

```
    diabetes_dataset.describe().T
```

Output:



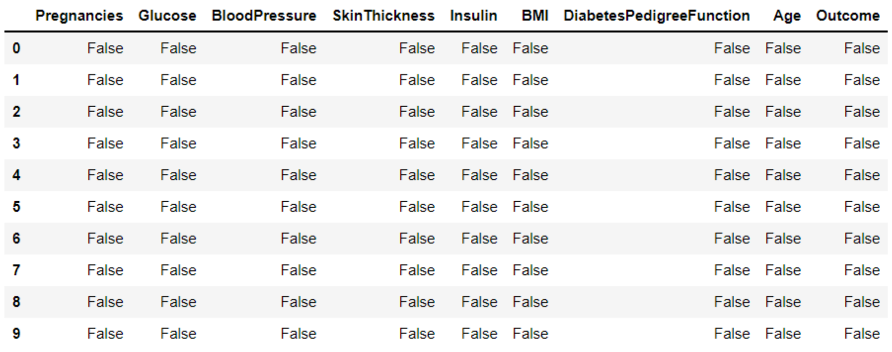
	count	mean	std	min	25%	50%	75%	max
Pregnancies	768.0	3.845052	3.369578	0.000	1.00000	3.0000	6.00000	17.00
Glucose	768.0	120.894531	31.972618	0.000	99.00000	117.0000	140.25000	199.00
BloodPressure	768.0	69.105469	19.355807	0.000	62.00000	72.0000	80.00000	122.00
SkinThickness	768.0	20.536458	15.952218	0.000	0.00000	23.0000	32.00000	99.00
Insulin	768.0	79.799479	115.244002	0.000	0.00000	30.5000	127.25000	846.00
BMI	768.0	31.992578	7.884160	0.000	27.30000	32.0000	36.60000	67.10
DiabetesPedigreeFunction	768.0	0.471876	0.331329	0.078	0.24375	0.3725	0.62625	2.42
Age	768.0	33.240885	11.760232	21.000	24.00000	29.0000	41.00000	81.00
Outcome	768.0	0.348958	0.476951	0.000	0.00000	0.0000	1.00000	1.00

Now let’s check that if our dataset have null values or not

Program:

```
diabetes_dataset.isnull().head(10)
```

Output:



	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False	False	False
6	False	False	False	False	False	False	False	False	False
7	False	False	False	False	False	False	False	False	False
8	False	False	False	False	False	False	False	False	False
9	False	False	False	False	False	False	False	False	False

Now let’s check the number of null values our dataset has.

Program:

```
diabetes_dataset.isnull().sum()
```

Output:

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

program:

```
diabetes_dataset['Outcome'].value_counts()
```

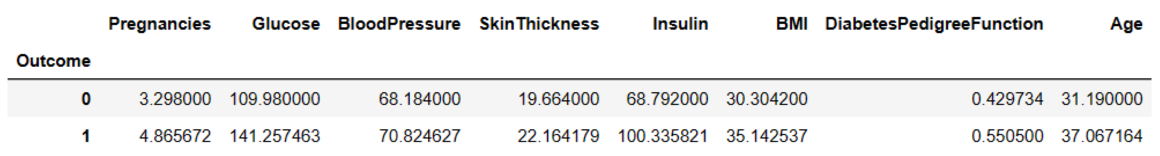
output:

```
Outcome
0     500
1     268
Name: count, dtype: int64
```

Program:

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

output:



	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
Outcome								
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

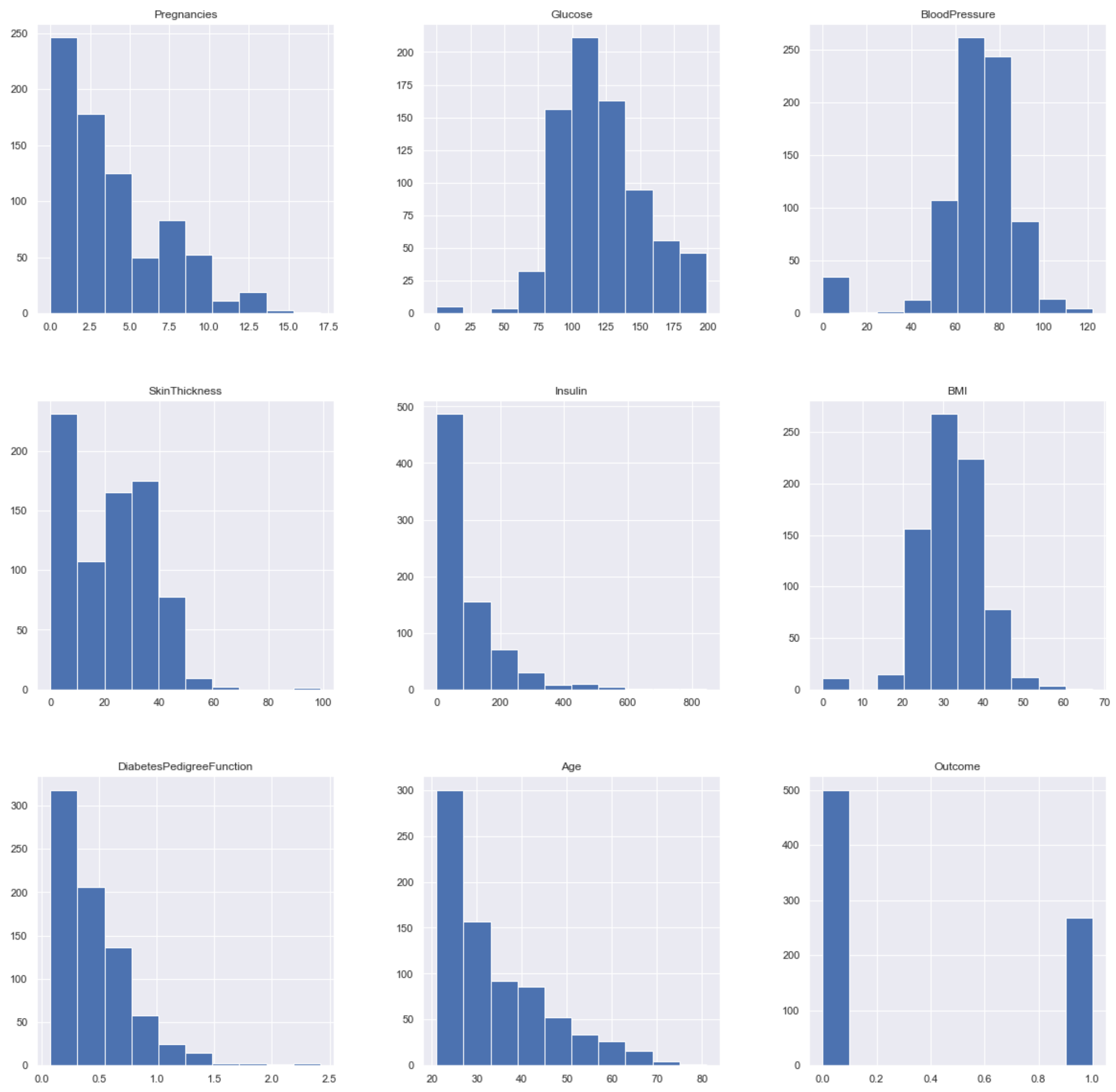
Data Visualization

Plotting the data distribution plots before removing null values

Program:

```
p = diabetes_dataset.hist(figsize = (20,20))
```

Output:



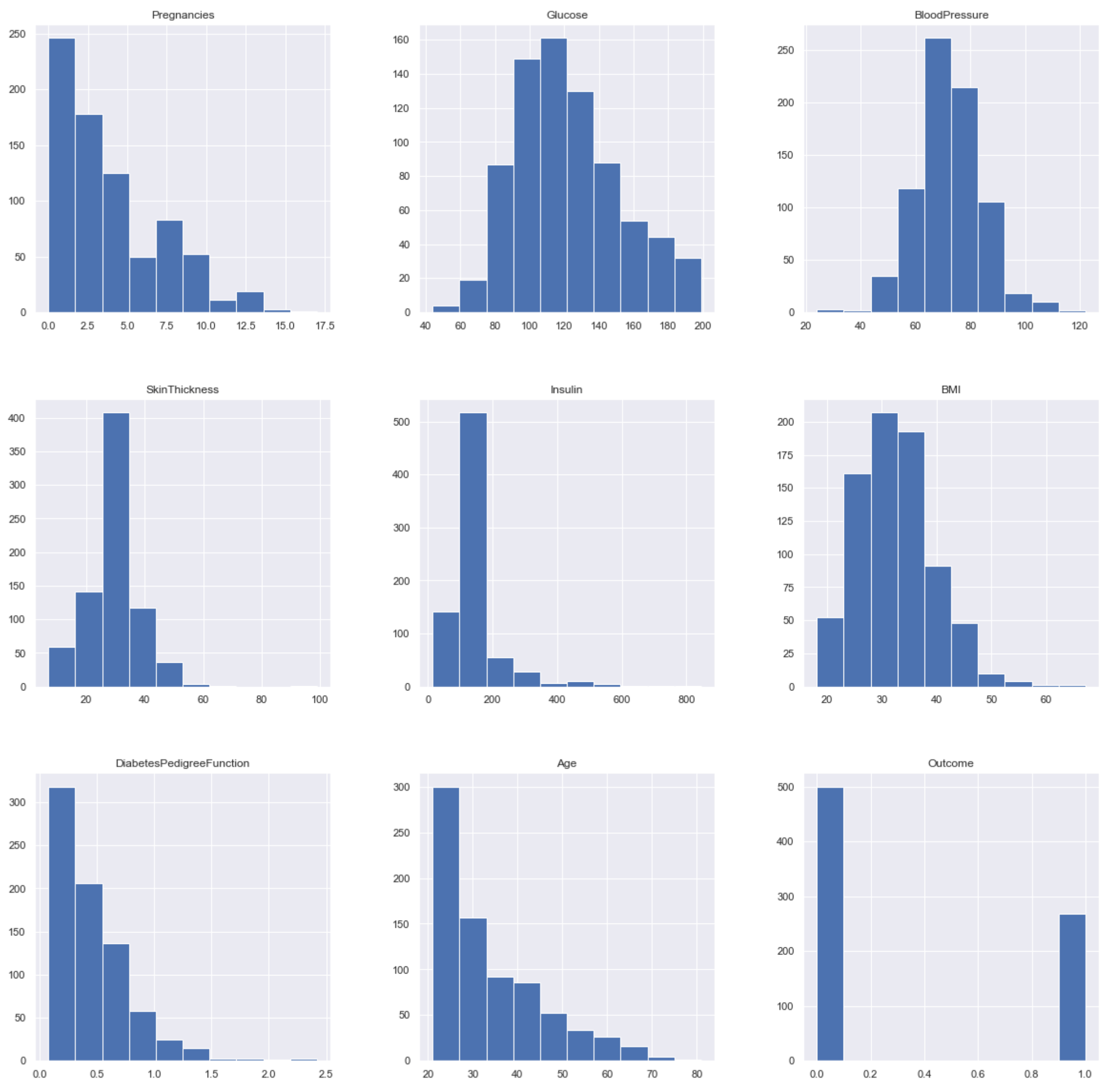
Inference: So here we have seen the distribution of each features whether it is dependent data or independent data and one thing which could always strike that **why do we need to see the distribution of data?** So the answer is simple it is the best way to start the analysis of the dataset as **it shows the occurrence of every kind of value in the graphical structure which in turn lets us know the range of the data.**

Plotting the distributions after removing the NAN values.

Program:

```
p = diabetes_dataset_copy.hist(figsize = (20,20))
```

Output:

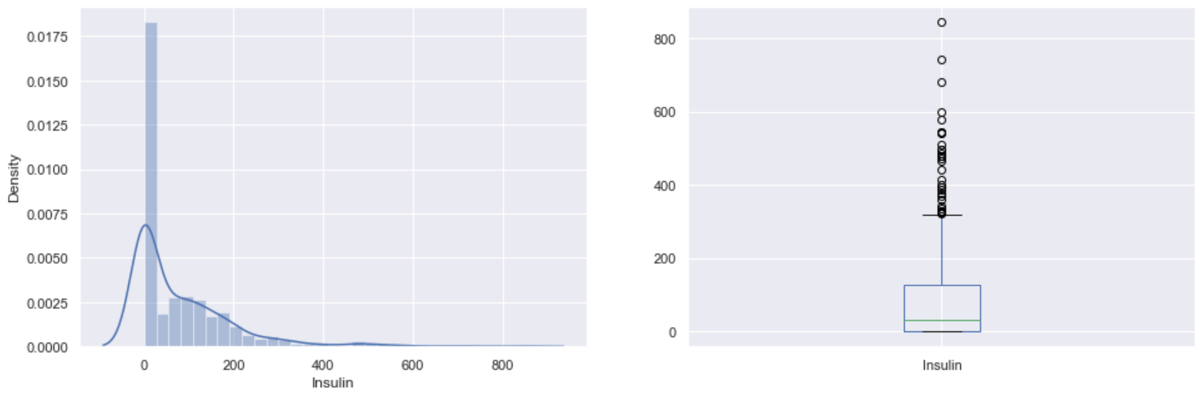


Inference: Here we are again using the hist plot to **see the distribution of the dataset** but this time we are using this visualization to see the changes that we can see after those null values are removed from the dataset and we can clearly see the difference **for example –** In age column after removal of the null values, **we can see that there is a spike at the range of 50 to 100 which is quite logical as well.**

<u>**Program:**</u>

```
plt.subplot(121),
sns.distplot(diabetes_dataset['Insulin'])
plt.subplot(122),
diabetes_dataset['Insulin'].plot.box(figsize=(16,5))
plt.show()
```

<u>**Output:**</u>



Inference: That's how **Distplot** can be helpful where one will able to see the distribution of the data as well as with the help of **boxplot one can see the outliers in that column** and other information too which can be derived by the **box and whiskers plot.**

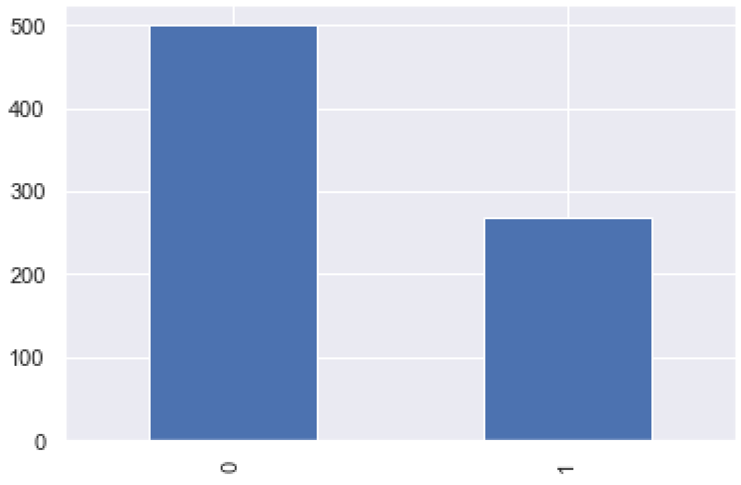
Now, let's check that how well our outcome column is balanced

Program:

```
color_wheel = {1: "#0392cf", 2: "#7bc043"}
colors = diabetes_dataset["Outcome"].map(lambda x:
color_wheel.get(x + 1))
print(diabetes_dataset.Outcome.value_counts())
p=diabetes_dataset.Outcome.value_counts().plot(kind=
"bar")
```

Output:

```
0    500
1    268
Name: Outcome, dtype: int64
```



Inference: Here from the above visualization it is clearly visible that our **dataset is completely imbalanced** in fact the number of patients who are **diabetic is half of the patients who are non-diabetic.**

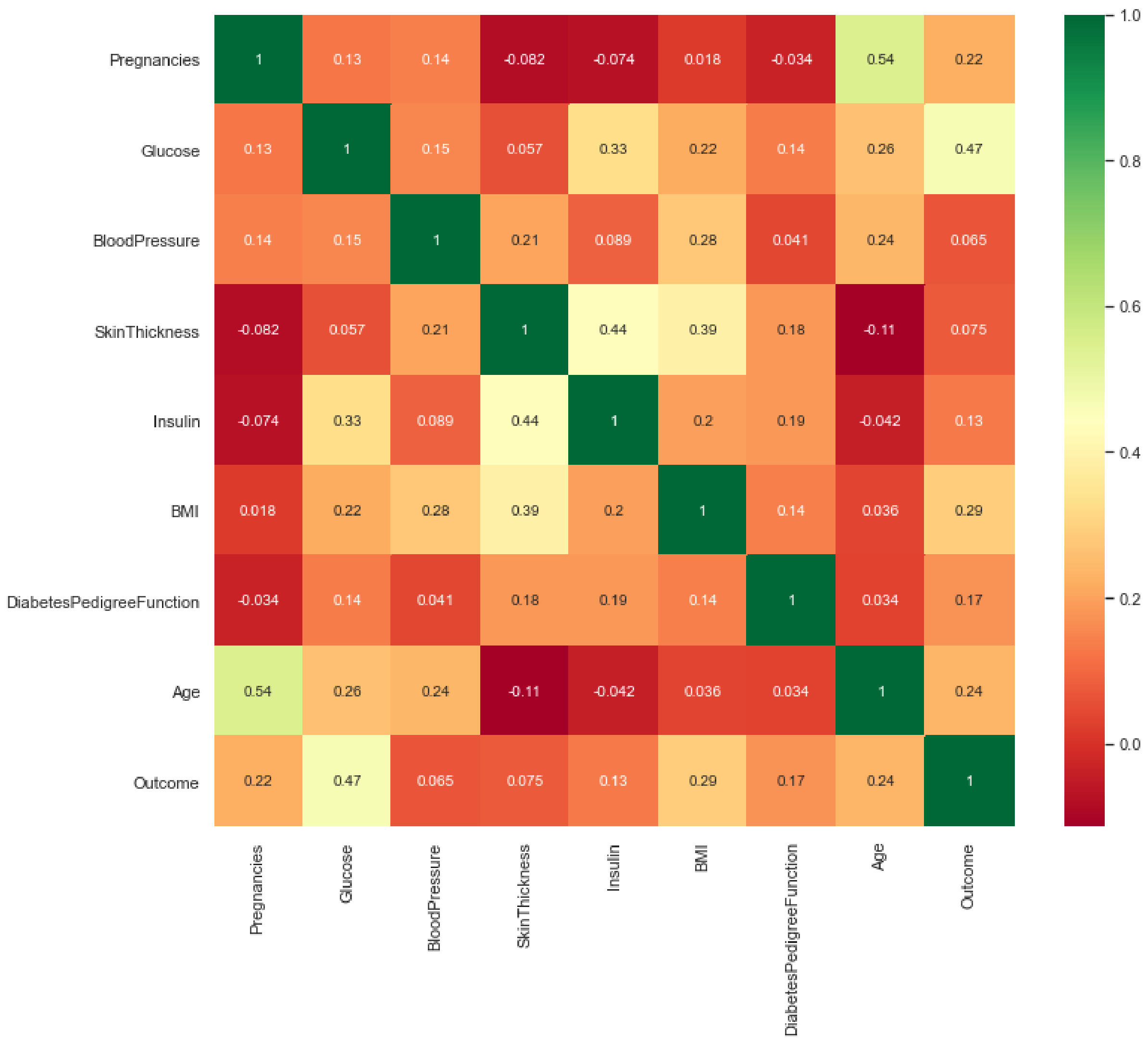
Correlation between all the features

Correlation between all the features before cleaning

Program:

```
plt.figure(figsize=(12,10))
# seaborn has an easy method to showcase heatmap
p = sns.heatmap(diabetes_dataset.corr(),
annot=True,cmap ='RdYlGn')
```

Output:



Program:

```
diabetes_dataset_copy = diabetes_dataset.copy(deep =
True)

diabetes_dataset_copy[['Glucose','BloodPressure','Sk
inThickness','Insulin','BMI']] =
diabetes_dataset_copy[['Glucose','BloodPressure','Sk
inThickness','Insulin','BMI']].replace(0,np.NaN)


# Showing the Count of NANs

print(diabetes_dataset_copy.isnull().sum())
```

Output:

```
Pregnancies                   0
Glucose                       5
BloodPressure                35
SkinThickness               227
Insulin                     374
BMI                          11
DiabetesPedigreeFunction      0
Age                           0
Outcome                       0
dtype: int64
```

As mentioned above that now **we will be replacing the zeros with the NAN values** so that we can impute it later to maintain the authenticity of the dataset as well as trying to have a better Imputation approach i.e **to apply mean values of each column to the null values of the respective columns.**

SELECTING MACHINE LEARNING ALGORITHM:

Creating an AI-based diabetes prediction system involves various steps, including data collection, preprocessing, feature engineering, model selection, training, and evaluation. There are several machine learning algorithms you can use for this task.

CODE:

```
# separating the data and labels
X = diabetes_dataset.drop(columns = 'Outcome', axis=1)
Y = diabetes_dataset['Outcome']
print(X)
```

OUTPUT:

```
   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
```

CODE:

```
print(Y)
```

OUTPUT:

```
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
```

Data Standardization:

Data standardization, also known as feature scaling, is a critical data preprocessing step in machine learning, including for AI-based diabetes prediction systems. It involves transforming numerical features into a common scale to ensure that no single feature dominates the learning process due to differences in their magnitudes.

CODE:

```
scaler = StandardScaler()
scaler.fit(X)
```

OUTPUT:

```
StandardScaler(copy=True, with_mean=True, with_std=True)
```

CODE:

```
standardized_data = scaler.transform(X)
print(standardized_data)
```

OUTPUT:

```
[[ 0.63994726  0.84832379  0.14964075 ...  0.20401277  0.46849198
   1.4259954 ]
 [-0.84488505 -1.12339636 -0.16054575 ... -0.68442195 -0.36506078
  -0.19067191]
 [ 1.23388019  1.94372388 -0.26394125 ... -1.10325546  0.60439732
  -0.10558415]
 ...
 [ 0.3429808   0.00330087  0.14964075 ... -0.73518964 -0.68519336
  -0.27575966]
 [-0.84488505  0.1597866  -0.47073225 ... -0.24020459 -0.37110101
   1.17073215]
 [-0.84488505 -0.8730192   0.04624525 ... -0.20212881 -0.47378505
  -0.87137393]]
```

CODE:

```
X = standardized_data
Y = diabetes_dataset['Outcome']
print(X)
print(Y)
```

OUTPUT:

```
[[ 0.63994726  0.84832379  0.14964075 ...  0.20401277  0.46849198
   1.4259954 ]
 [-0.84488505 -1.12339636 -0.16054575 ... -0.68442195 -0.36506078
  -0.19067191]
 [ 1.23388019  1.94372388 -0.26394125 ... -1.10325546  0.60439732
  -0.10558415]
 ...
 [ 0.3429808   0.00330087  0.14964075 ... -0.73518964 -0.68519336
  -0.27575966]
 [-0.84488505  0.1597866  -0.47073225 ... -0.24020459 -0.37110101
   1.17073215]
 [-0.84488505 -0.8730192   0.04624525 ... -0.20212881 -0.47378505
  -0.87137393]]
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
```

Train Test Split:

In an AI-based diabetes prediction system, the train-test split is a critical step in the data preprocessing phase. This split allows you to assess the model's performance on unseen data, ensuring that it can generalize well to new patient data. The train-test split divides the dataset into two sets: one for training the model and one for testing its performance.

CODE:

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, stratify=Y, random_state=2)

print(X.shape, X_train.shape, X_test.shape)
```

OUTPUT:

```
(768, 8) (614, 8) (154, 8)
```

Training the Model:

Training the model for an AI-based diabetes prediction system involves several steps, and it's essential to follow a systematic approach to ensure the best possible results.

CODE:

```
        classifier = svm.SVC(kernel='linear')


         #training the support vector Machine Classifier
        classifier.fit(X_train, Y_train)
```

OUTPUT:

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
   decision_function_shape='ovr', degree=3, gamma='scale', kernel='linear',
   max_iter=-1, probability=False, random_state=None, shrinking=True,
   tol=0.001, verbose=False)
```

Model Evaluation:

Model evaluation is a crucial step in assessing the performance and reliability of an AI-based diabetes prediction system. You need to ensure that your model is accurate, generalizes well to unseen data, and aligns with your specific project objectives.

Accuracy Score :

Accuracy is a commonly used evaluation metric for classification tasks, including AI-based diabetes prediction systems. It measures the proportion of correctly predicted instances (both true positives and true negatives) to the total number of instances in your dataset.

CODE:

```
        # accuracy score on the training data
        X_train_prediction = classifier.predict(X_train)
```

```
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)

print('Accuracy score of the training data : ', training_data_accuracy)
```

OUTPUT:

```
Accuracy score of the training data :  0.7866449511400652
```

CODE:

```
# accuracy score on the test data

X_test_prediction = classifier.predict(X_test)

test_data_accuracy = accuracy_score(X_test_prediction, Y_test)


print('Accuracy score of the test data : ', test_data_accuracy)
```

OUTPUT:

```
Accuracy score of the test data :  0.7727272727272727
```

Making a Predictive System:

A predictive system, often referred to as a predictive analytics system, is a technology or software solution that uses data analysis and statistical algorithms to make predictions or forecasts about future events, trends, or outcomes. These systems are widely used across various domains, including healthcare, finance, marketing, and manufacturing, to assist in decision-making, improve efficiency, and gain insights.

CODE:

```
input_data = (5,166,72,19,175,25.8,0.587,51)
```

```
# changing the input_data to numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the array as we are predicting for one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

# standardize the input data
std_data = scaler.transform(input_data_reshaped)
print(std_data)

prediction = classifier.predict(std_data)
print(prediction)

if (prediction[0] == 0):
  print('THE PERSSON IS NOT DIABETIC’)
else:
  print('THE PERSON IS DIABETIC’)
```

OUTPUT:

```
[[ 0.3429808   1.41167241  0.14964075 -0.09637905  0.82661621 -0.78595734
   0.34768723  1.51108316]]
[1]                 THE PERSON IS DIABETIC
```

ANOTHER EXAMPLE:

In this example we have predicted the model using with different input data to show the various predicting outcomes such as,

CODE:

```
input_data = (6,146,82,69,125,15.8,0.507,31)

# changing the input_data to numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the array as we are predicting for one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

# standardize the input data
std_data = scaler.transform(input_data_reshaped)
print(std_data)

prediction = classifier.predict(std_data)
print(prediction)

if (prediction[0] == 0):
  print('THE PERSON IS NOT DIABETIC’)
else:
  print('THE PERSON IS DIABETIC')
```

OUTPUT:

```
[[ 0.63994726  0.7857295   0.66661825  3.040024    0.39247142 -2.0551498
   0.10607774 -0.19067191]]
[0]                  THE PERSON IS NOT DIABETIC
```

DIFFICULTIES OF BUILDING AN AI-BASED DIABETES PREDICTION SYSTEM:

Building an AI-based diabetes prediction system presents several challenges and difficulties that need to be addressed for the system to be effective, reliable, and ethically sound. Here are some of the key challenges and difficulties in developing such a system:

- Data Quality and Availability
- Data Privacy and Security
- Data Imbalance
- Feature Engineering
- Model Selection and Tuning
- Interpretability
- Bias and Fairness
- Generalization
- Continuous Monitoring and Updating
- Integration with Healthcare Workflow
- Communication and Education
- Robustness and Security
- Ethical and Legal Considerations
- Scalability

ADVANTAGES OF AI-BASED DIABETES PREDICTION SYSTEM:

AI-based diabetes prediction systems offer numerous advantages that can significantly impact healthcare and patient outcomes. Here are some key advantages of such systems:

- Early Detection
- Personalized Risk Assessment
- Efficiency
- Improved Patient Outcomes
- Data-Driven Insights
- Reduction of Healthcare Costs
- Support for Healthcare Providers
- Continuous Monitoring
- Patient Empowerment

- Research Advancements
- Scalability
- Reduction of Diagnostic Delays
- Population Health Management
- Remote Monitoring

DISADVANTAGES OF AI-BASED DIABETES PREDICTION SYSTEM:

AI-based diabetes prediction systems offer numerous advantages, as mentioned in previous responses. However, they also come with certain disadvantages and challenges that need to be carefully considered. Here are some of the disadvantages of AI-based diabetes prediction systems:

- Bias and Fairness
- Overfitting and Underfitting
- Lack of Context
- Ethical Concerns
- Complexity and Cost
- Maintenance and Updating
- Integration with Healthcare Workflow
- Patient Education
- Legal and Regulatory Compliance
- Dependency on Technology
- Limited Predictive Power

CONCLUSION:

In conclusion, the development of an AI-based diabetes prediction system represents a significant leap forward in the field of healthcare and has the potential to transform how we approach the prevention, management, and early detection of diabetes. This innovative system, designed with a user-centered approach and following a structured process, holds great promise in improving the lives of individuals at risk of diabetes and those already diagnosed with the condition. Here are some key takeaways from the entire process:

- Early Detection and Prevention: An AI-based diabetes prediction system enables early detection of diabetes risk factors and empowers individuals with information to make lifestyle changes, potentially preventing the onset of the disease.

- ❖ Personalized Care: By leveraging AI, the system can provide personalized care plans, offering tailored recommendations for diet, exercise, and medication management based on individual health data.
- ❖ Continuous Monitoring: The system allows for continuous monitoring of glucose levels and other relevant health parameters, which is essential for maintaining stable blood sugar levels and reducing the risk of complications.
- ❖ Data-Driven Decision Support: Healthcare professionals benefit from AI-driven decision support tools that assist in treatment planning and medication management, ultimately improving patient care.
- ❖ Public Health Insights: Population-level data analysis can contribute to public health initiatives, offering insights into diabetes trends, risk factors, and informing public policy and awareness campaigns.
- ❖ Ethical Considerations: Privacy and ethical concerns must be addressed throughout the development process to ensure that sensitive health data is safeguarded, and that the system is fair, transparent, and equitable for all users.
- ❖ User-Centric Approach: Applying design thinking ensures that the system is developed with a strong focus on user needs and preferences, resulting in a more effective and user-friendly solution.
- ❖ Iterative Development: An iterative approach allows for continuous improvement and adaptation, ensuring that the system remains up to date and relevant in the ever-evolving field of healthcare and AI.

Overall, an AI-based diabetes prediction system represents a significant step towards more proactive and effective healthcare. It has the potential to enhance the quality of life for individuals at risk of diabetes, reduce the burden on healthcare systems, and contribute to the broader goal of combatting the global diabetes epidemic. As technology and data continue to advance, the future of diabetes management and prevention holds great promise, driven by the power of artificial intelligence.