INNOVATION OF AI BASED DIABETES PREDICTION SYSTEM

Description:

This document outlines the innovation of an AI powered diabetes prediction system.The system primary objective is to analyse the given medical data and predict the likelihood of an individual developing diabetes using a machine learning algorithm.

Procedure:

The dataset comprises crucial health-related features such as 'Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', and 'Age'. The main objective was to predict the 'Outcome' label, which signifies the likelihood of diabetes.

Importing Data Set:

Want to import the NumPy, Matplotlib, and Pandas libraries in Python, you can do so by adding the following lines to your code:

```
import numpy as np  # Import NumPy and use 'np' as an alias for it
import matplotlib.pyplot as plt  # Import Matplotlib's pyplot module and use 'plt' as an alias
import pandas as pd  # Import Pandas and use 'pd' as an alias
```

Here's the code with these imports:

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

# Replace 'your_dataset.csv' with the actual path to your dataset file
data = pd.read_csv('your_dataset.csv')

# Display the first few rows of the dataset to verify the import
print(data.head())
```

In this code:

- `import numpy as np` imports the NumPy library and assigns the alias "np" for convenience. NumPy is often used for numerical operations and working with arrays.

- `import matplotlib.pyplot as plt` imports the `pyplot` module from Matplotlib and assigns the alias "plt." Matplotlib is a powerful library for creating data visualisations, including graphs and charts.

- `import pandas as pd` imports the Pandas library and assigns the alias "pd." Pandas is used for data manipulation and analysis, including importing, cleaning, and exploring datasets.

Ensure that you have these libraries installed in your Python environment by running `pip install numpy matplotlib pandas`

Advanced Machine Learning Algorithms:

Continuous innovation in machine learning algorithms, such as deep learning and reinforcement learning, can lead to more accurate and efficient diabetes prediction models. These algorithms can handle complex, multi-dimensional data and extract meaningful patterns and insights.

Integration of Multiple Data Sources:

Innovations in data integration techniques can enable the incorporation of diverse data sources, including genetic, clinical, wearable device, and environmental data, to provide a more comprehensive view of an individual's diabetes risk.

Real-time Predictive Analytics:

The development of real-time predictive analytics systems can enable instant risk assessment and proactive interventions. This innovation is crucial for patients with diabetes, as it allows for immediate feedback and timely management of glucose levels.

Personalised Medicine: Innovations in personalised medicine within the context of diabetes prediction involve tailoring recommendations and interventions to an individual's unique genetic and lifestyle factors. AI can play a pivotal role in providing personalised guidance for diabetes prevention and management.

User-Friendly Interfaces:

Innovations in user interfaces and user experience design can make diabetes prediction and management tools more accessible to both healthcare providers and patients. Mobile apps, chatbots, and other user-friendly platforms can facilitate engagement and adherence to recommendations.

Telemedicine and Remote Monitoring:

The integration of AI with telemedicine and remote monitoring technologies can empower healthcare providers to remotely monitor and manage diabetes patients. This innovation can extend the reach of diabetes care to underserved areas and improve patient outcomes.

Ethical and Privacy Considerations:

Innovations in ethical and privacy considerations are essential, especially when dealing with sensitive health data. Developing robust data anonymization, encryption, and consent management systems is crucial to protect patient information.

Interoperability and Data Sharing:

Innovations in interoperability standards and data sharing mechanisms can facilitate the exchange of health data between different healthcare providers and systems. This promotes more comprehensive diabetes prediction and care.

AI-Enabled Drug Discovery:

AI-driven drug discovery and development can lead to the creation of more effective medications for diabetes prevention and treatment. Innovations in this area can result in novel therapeutic options.

Collaborative Research Initiatives:

Encouraging collaboration among researchers, healthcare providers, technology companies, and patient advocacy groups can foster innovation in diabetes prediction. Multidisciplinary efforts can lead to breakthroughs and the integration of new technologies and insights.

Innovation in AI-based diabetes prediction is essential for staying at the forefront of healthcare advancements. These innovations not only enhance the accuracy of predictions but also improve patient outcomes, reduce the economic burden of diabetes care, and promote proactive health management. Additionally, ethical and privacy considerations remain central to the development and deployment of these innovations to ensure the responsible use of health data.

AI-based diabetes prediction is a sophisticated application of artificial intelligence (AI) and machine learning that aims to predict the likelihood of an individual developing diabetes or their risk of complications associated with diabetes. Here is a detailed description:

Artificial Intelligence (AI) for Diabetes Prediction:

In the realm of healthcare, AI has emerged as a powerful tool for early diagnosis and risk assessment. When applied to diabetes, AI-based prediction systems leverage advanced algorithms and data analysis techniques to provide invaluable insights into a patient's susceptibility to diabetes or their potential complications. The following components define AI-based diabetes prediction:

Data Collection and Integration: AI diabetes prediction systems begin by aggregating extensive datasets. These datasets can include patient medical records, lifestyle information, genetic data, and biomarkers like blood glucose levels, HbA1c measurements, and insulin sensitivity.

Feature Selection:

The AI system identifies the most relevant features or variables from the collected data, such as family history, age, BMI, dietary habits, physical activity, and more. This step helps in reducing noise and improving the accuracy of predictions.

Machine Learning Algorithms:

Various machine learning algorithms are then employed to analyse and model the data. Common algorithms include logistic regression, decision trees, random forests, support vector machines, and deep learning neural networks.

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)**

Import Required Libraries

```
# Ignore warning messages to prevent them from being displayed during code
execution
import warnings
warnings.filterwarnings('ignore')
```

```
import numpy as np   # Importing the NumPy library for linear algebra operations

import pandas as pd  # Importing the Pandas library for data processing and CSV file
handling

import os

for dirname, _, filenames in os.walk('/kaggle/input'):

    for filename in filenames:

        print(os.path.join(dirname, filename))
```

```python
import seaborn as sns                    # Importing the Seaborn library for statistical data visualization

import matplotlib.pyplot as plt          # Importing the Matplotlib library for creating plots and visualizations

import plotly.express as px              # Importing the Plotly Express library for interactive visualizations
```

Exploratory Data Analysis:

Load and Prepare Data:

In [5]:

```python
df=pd.read_csv('/kaggle/input/diabetes-data-set/diabetes.csv')
```

UnderStanding the Variables

In [6]:

```python
df.head(10)
```

Out[6]:

	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
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1

In [7]:

```python
df.tail(10)
```

Out[7]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
758	1	106	76	0	0	37.5	0.197	26	0
759	6	190	92	0	0	35.5	0.278	66	1
760	2	88	58	26	16	28.4	0.766	22	0
761	9	170	74	31	0	44.0	0.403	43	1
762	9	89	62	0	0	22.5	0.142	33	0
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

In [8]:

```
df.sample(5)
```

Out[8]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
760	2	88	58	26	16	28.4	0.766	22	0
687	1	107	50	19	0	28.3	0.181	29	0
355	9	165	88	0	0	30.4	0.302	49	1
187	1	128	98	41	58	32.0	1.321	33	1
235	4	171	72	0	0	43.6	0.479	26	1

In [9]:

```
df.describe()
```

Out[9]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.0000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.24088
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.76023
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.00000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.00000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.00000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.00000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.00000

```
df.dtypes
```

\

```
Pregnancies                   int64
Glucose                       int64
BloodPressure                 int64
SkinThickness                 int64
Insulin                       int64
BMI                         float64
DiabetesPedigreeFunction    float64
Age                           int64
Outcome                       int64

dtype: object
```

\

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
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
```

```
df.size
```

```
6912
```

```
df.shape
```

```
(768, 9)
```

Data Cleaning:

```
df.shape
```

```
(768, 9)
```

```
df=df.drop_duplicates()
df.shape
```

```
(768, 9)
```

Check null Values

```
df.isnull().sum()
```

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0

dtype: int64
```

There is no Missing Values present in the Data

```
df.columns
```

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

      dtype='object')
```

Check the number of Zero Values in Dataset

```
print("No. of Zero Values in Glucose ", df[df['Glucose']==0].shape[0])
No. of Zero Values in Glucose  5
print("No. of Zero Values in Blood Pressure ",
df[df['BloodPressure']==0].shape[0])
No. of Zero Values in Blood Pressure  35
print("No. of Zero Values in SkinThickness ",
df[df['SkinThickness']==0].shape[0])
No. of Zero Values in SkinThickness  227
print("No. of Zero Values in Insulin ", df[df['Insulin']==0].shape[0])
No. of Zero Values in Insulin  374
print("No. of Zero Values in BMI ", df[df['BMI']==0].shape[0]
No. of Zero Values in BMI  11
```

Replace zeroes with mean of that Columns

```
df['Glucose']=df['Glucose'].replace(0, df['Glucose'].mean())
print('No of zero Values in Glucose ', df[df['Glucose']==0].shape[0])
```

```
No of zero Values in Glucose  0
df['BloodPressure']=df['BloodPressure'].replace(0,
df['BloodPressure'].mean())
df['SkinThickness']=df['SkinThickness'].replace(0,
df['SkinThickness'].mean())
df['Insulin']=df['Insulin'].replace(0, df['Insulin'].mean())
df['BMI']=df['BMI'].replace(0, df['BMI'].mean())
```

Validate the Zero Values

```
df.describe()
```

Conclusion:

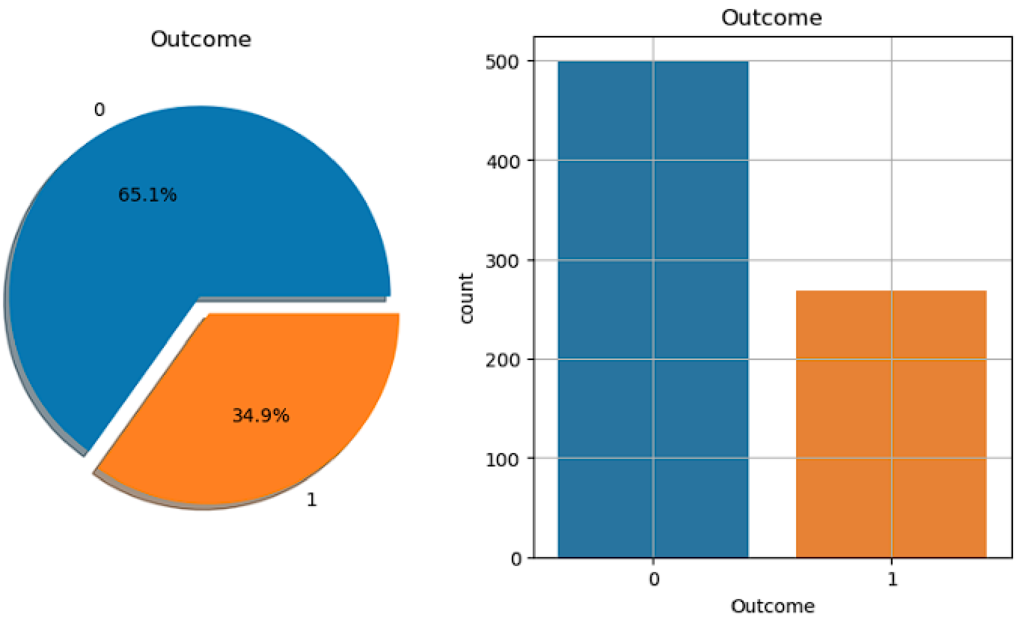
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	121.681605	72.254807	26.606479	118.660163	32.450805	0.471876	33.240885	0.348958
std	3.369578	30.436016	12.115932	9.631241	93.080358	6.875374	0.331329	11.760232	0.476951
min	0.000000	44.000000	24.000000	7.000000	14.000000	18.200000	0.078000	21.000000	0.000000

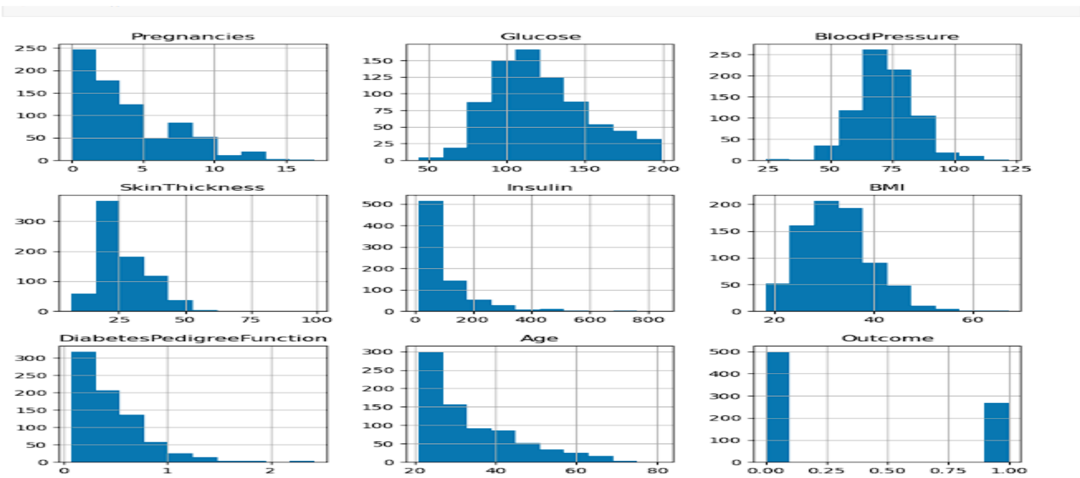
Data Visualization:

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

# Assuming 'df' is your DataFrame containing the dataset
# If you haven't imported your dataset yet, import it here

# Create subplots
f, ax = plt.subplots(1, 2, figsize=(10, 5))

# Pie chart for Outcome distribution
df['Outcome'].value_counts().plot.pie(explode=[0, 0.1],
autopct='%1.1f%%', ax=ax[0], shadow=True)
ax[0].set_title('Outcome')
ax[0].set_ylabel(' ')

# Count plot for Outcome distribution
sns.countplot(x='Outcome', data=df, ax=ax[1])  # Use 'x' instead of
'Outcome'
ax[1].set_title('Outcome')
```

```python
# Display class distribution
N, P = df['Outcome'].value_counts()
print('Negative (0):', N)
print('Positive (1):', P)

# Adding grid and showing plots
plt.grid()
plt.show()
```

```
Negative (0): 500
Positive (1): 268
```





AI-based diabetes prediction holds significant promise in improving patient outcomes, reducing the burden of diabetes-related healthcare costs, and enhancing the overall quality of diabetes care. By harnessing the power of data and artificial intelligence, healthcare professionals can make more informed decisions and patients can take proactive steps to manage or prevent diabetes.

Thankyou

Done by:
SURENDAR S
RAVIKUMAR S
GOWTHAM S
KARTHIKEYAN V
HARINI S