AI BASED DIABETES PREDICTION SYSTEM

TEAM MEMBER: R. Japhia Pauline

AI _Phase 1 Submission

PROJECT: **Creating an AI model predicting diabetes risk using medical data**

OBJECTIVE:

The objective of this project is to develop a machine learning model that accurately predicts the diabetes risk using the given medical data.

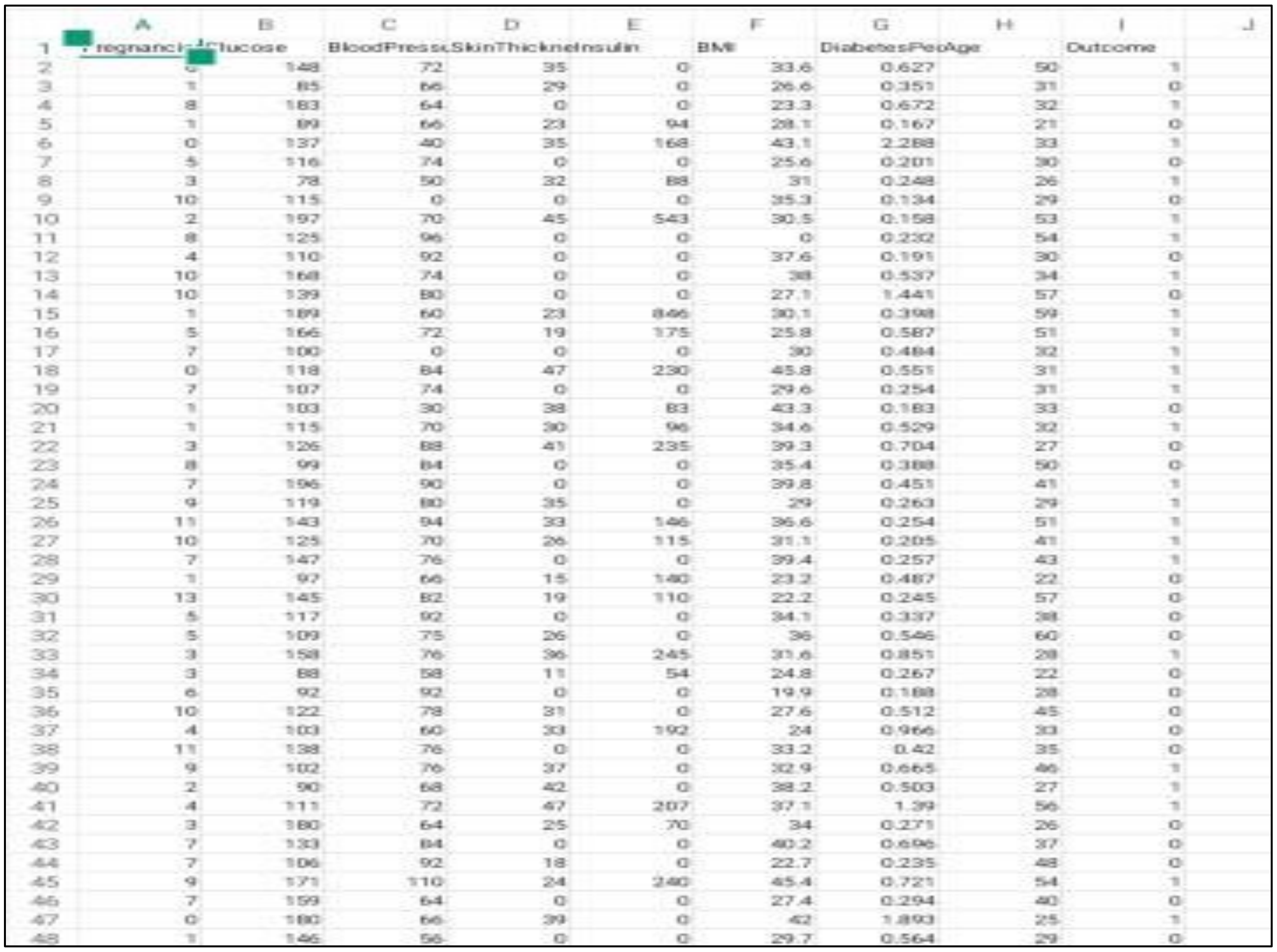
PHASE 1: Problem Defining and Design Thinking

1.DATA COLLECTION

There are a variety of sources of data that can be used for an AI-based diabetes prediction system, including: Medical records: This data can provide information about a patient’s medical history, including demographics, lab results, medications, and diagnoses.

- Wearable devices: Wearable devices can track a variety of health metrics, such as blood sugar levels, heart rate, and activity levels.
- Patient surveys: Patient surveys can collect information about a patient’s lifestyle, symptoms, and risk factors for diabetes.

It is important to collect data from a diverse population in order to train a model that is generalizable. The data should also be representative of the population that the model will be used to predict diabetes.



	A	B	C	D	E	F	G	H	I	J
1	Pregnancie	Glucose	BloodPressu	SkinThickne	Insulin	BMI	DiabetesPed	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	
26	11	143	94	33	146	36.6	0.254	51	1	
27	10	125	70	26	115	31.1	0.205	41	1	
28	7	147	76	0	0	39.4	0.257	43	1	
29	1	97	66	15	140	23.2	0.487	22	0	
30	13	145	82	19	110	22.2	0.245	57	0	
31	5	117	92	0	0	34.1	0.337	38	0	
32	5	109	75	26	0	36	0.546	60	0	
33	3	158	76	36	245	31.6	0.851	28	1	
34	3	88	58	11	54	24.8	0.267	22	0	
35	6	92	92	0	0	19.9	0.188	28	0	
36	10	122	78	31	0	27.6	0.512	45	0	
37	4	103	60	33	192	24	0.966	33	0	
38	11	138	76	0	0	33.2	0.42	35	0	
39	9	102	76	37	0	32.9	0.665	46	1	
40	2	90	68	42	0	38.2	0.503	27	1	
41	4	111	72	47	207	37.1	1.39	56	1	
42	3	180	64	25	70	34	0.271	26	0	
43	7	133	84	0	0	40.2	0.696	37	0	
44	7	106	92	18	0	22.7	0.235	48	0	
45	9	171	110	24	240	45.4	0.721	54	1	
46	7	159	64	0	0	27.4	0.294	40	0	
47	0	180	66	39	0	42	1.893	25	1	
48	1	146	56	0	0	29.7	0.564	29	0	

DATASET**LINK:**
https://www.kaggle.com/code/japhiapauliner/diabetespredictor-using-logistic-regression/edit

2.DATA PREPROCESSING

Once the data has been collected, it needs to be preprocessed to make it suitable for machine learning. This may involve the following steps:

- Cleaning the data: This involves removing errors and inconsistencies from the data. For example, you may need to remove missing values, correct typos, and standardize the format of the data.
- Scaling the data: This involves normalizing the values of different features so that they are on the same scale. This can be done using a variety of scaling techniques, such as min-max scaling and standard scaling.
- Encoding categorical features: Categorical features, such as gender and race, need to be encoded into numerical values before they can be used by machine learning algorithms. This can be done using a variety of encoding techniques, such as one-hot encoding and label encoding.

Once the data has been preprocessed, it is ready to be used to train a machine learning model to predict diabetes.

- Here are some additional tips for data collection and data preprocessing for an AI-based diabetes prediction system:

Use a large and diverse dataset. The more data you have, the better your model will be able to learn to predict diabetes. Additionally, it is important to use a dataset that is representative of the population that you are trying to predict diabetes for.

- Remove outliers. Outliers can skew the results of your model. It is important to remove outliers from the data before training the model.
- Use a variety of data preprocessing techniques. There is no one-size-fits-all approach to data preprocessing. It is important to use a variety of techniques to clean, scale, and encode the data in a way that is appropriate for your machine learning algorithm.

PYTHON PROGRAM:

INPUT AND OUTPUT:

```
 import num p y as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e g. pd. read_ csv)
import mat plot  lib.py plot as p l t
import seaborn as s  ns
In [2]:
dataset=pd.read_csv("/kaggle/input/diabetes-data-set/diabetes.csv")
In [3]:
dataset. Head ()
```

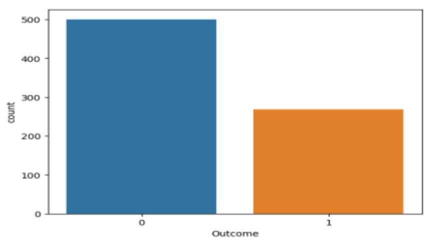
Out[3] : Pregnancies	Glucose	Blood Pressure	Skin Thickness	Insulin	BMI	Diabetes Pedigree Function	Age	Outcome	
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
4]:
(768, 9)
In [5]:
#check if null value is present
Dataset .is null(). Values .any()
Out[5]:
False
In [6]:
dataset.info()
<class 'pandas. core. frame. Data Frame'>
Range Index: 768 entries, 0 to 767
Data columns (total 9 columns):
# Column Non-Null Count D type
--- ------ -------------- -----
0 Pregnancies 768 non-null int64
1 Glucose 768 non-null int64
2 Blood Pressure 768 non-null int64
3 Skin Thickness 768 non-null int64
4 Insulin 768 non-null int64
5 BMI 768 non-null float64
6 Diabetes Pedigree Function 768 non-null float64
7 Age 768 non-null int64
8 Outcome 768 non-null int64
D types: float64(2), int64(7)
memory usage: 54.1 KB
In [7]:
```

```
dataset. describe()
```

	Pregnancies	Glucose	Blood Pressure	Skin Thickness	Insulin	BMI	Diabetes Pedigree Function	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
	Pregnancies	Glucose	Blood Pressure	Skin Thickness	Insulin	BMI	Diabetes Pedigree Function	Age	Outcome
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

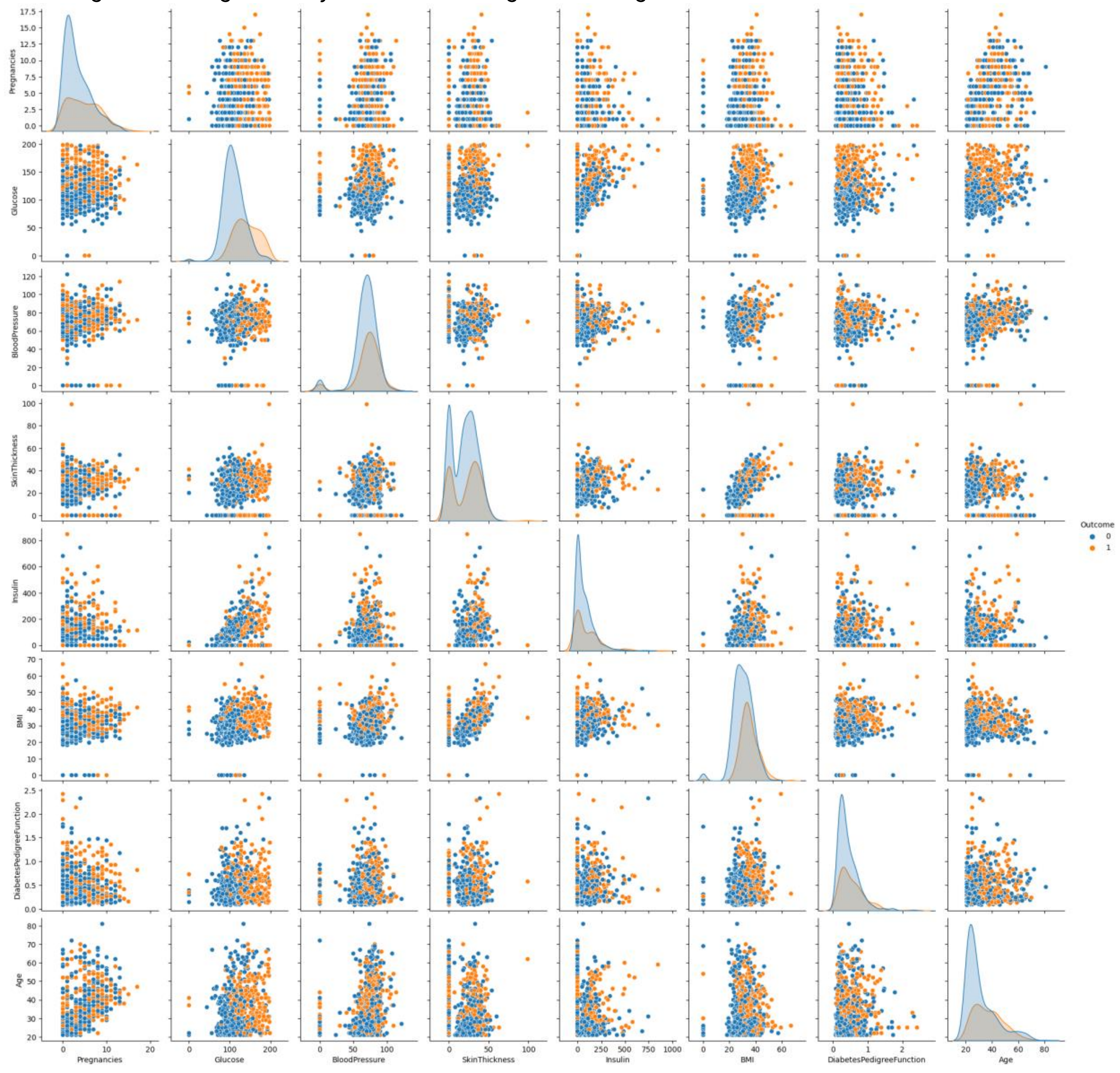
```
Dataset is  null().sum()
Out[8]:
Pregnancies 0
Glucose 0
Blood Pressure 0
Skin Thickness 0
Insulin 0
BMI 0
Diabetes Pedigree Function 0
Age 0
Outcome 0
D type: int64
In [9]:
#data visualization
S ns.  Count plot(x = 'Outcome', data = dataset)
Out[9]:
```



```
<Axes: x label='Outcome', y label='count>
```

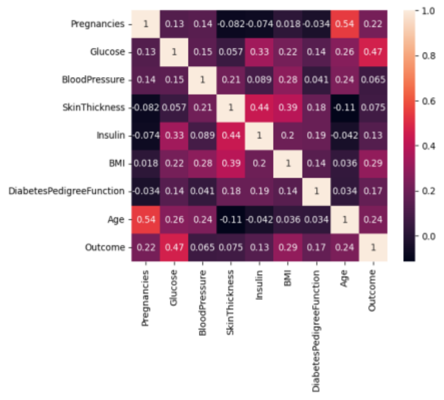
'

```
In [10]:
# Pair plot
S ns. pair plot(data = dataset, hue = 'Outcome')
P lt. show()
/opt/co n da/lib/python3.10/site-packages/seaborn/axisgrid.py:118: User
Warning: The figure layout has changed to tight
```



```
self._ figure. tight_ layout(*a r g s, **k w a r g s)
```

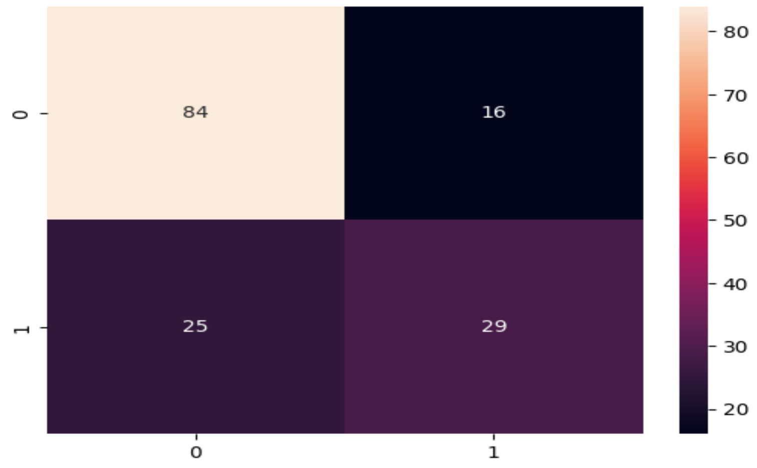
```
# Heatmap
S ns. heat map(data set .c o r r(), a n  not = True)
P lt. show()
```



```
In [12]:
# Replacing zero values with Na N
Data set _ new = dataset
dataset_ new[["Glucose", "Blood Pressure", "Skin Thickness", "Insulin",
"BMI"]] = dataset_new[["Glucose", "Blood Pressure", "Skin Thickness",
"Insulin", "BMI"]].replace(0, np.NaN)
# Count of Na N
dataset_ new. Is null(). sum()
Out[13]:
Pregnancies 0
Glucose 5
Blood Pressure 35
Skin Thickness 227
Insulin 374
BMI 11
Diabetes Pedigree Function 0
Age 0
Outcome 0
D type: int64
In [14]:
# Replacing Na N with mean values
dataset_ new["Glucose"].fill n a(dataset_ new["Glucose"].mean(), in place =
True)
dataset_new["BloodPressure"].fillna(dataset_new["BloodPressure"].mean(), in
place = True)
dataset_new["SkinThickness"].fillna(dataset_new["SkinThickness"].mean(), in
place = True)
dataset_ new["Insulin"].fill n a(dataset_ new["Insulin"].mean(), in place =
True)
dataset_ new["BMI"] .fill n a (dataset_ new["BMI"].mean(), in place = True)
In [15]:
dataset_ new is null (). Sum ()
Out[15]:
Pregnancies 0
Glucose 0
```

```
Blood Pressure 0
Skin Thickness 0
Insulin 0
BMI 0
Diabetes Pedigree Function 0
Age 0
Outcome 0
D type: int64
In [16]:
#Logistic regression
y = dataset_ new['Outcome']
X = dataset_ new. drop('Outcome', axis=1)
In [17]:
# Splitting X and Y
from s k learn. model_  selection import train_ test_ split
X_ train, X_ test, Y_ train, Y_ test = train_ test split(X, y, test_ size =
0.20, random_ state = 42, stratify = dataset_ new['Outcome'] )
In [18]:
from s k learn. linear_ model import Logistic Regression
model = Logistic Regression()
model. fit(X_ train, Y_ train)
y_ predict = model.  predict(X_ test)
/opt/conda/lib/python3.10/site-
packages/sklearn/linear_model/_logistic.py:458: Convergence Warning: l b f
g s failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
Increase the number of iterations (max_ I ter) or scale the data as shown
in:
https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
n_ I t er_ i = _check_  opt I m I z_ result(
In [19]:
y_ predict
Out[19]:
array([1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1,
0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0,
1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1,
0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1,
0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0])
In [20]:
# Confusion matrix
from s k learn . metrics import confusion_ matrix
cm = confusion_ matrix (Y_ test, y_ predict)
cm
```

```
Out[20]:
array([[84, 16],
[25, 29]])
In [21]:
# Heatmap of Confusion matrix
S  ns .heat map(pd .Data Frame(cm), anno t=True)
Out[21]:
```



```
<Axes: >

from s k l ear   .  metrics import accuracy_ score
In [23]:
accuracy =accuracy_ score(Y_ test, y_ predict)
accuracy
Out[23]:
0.7337662337662337
In [24]:
#Example: Let's check whether the person have diabetes or not using some random
values
y_ predict = model .predict([[1,148,72,35,79.799,33.6,0.627,50]])
print (y_ predict)
if y_ predict==1:
print("Diabetic")
else:
print (" Non Diabetic ")
[1]
Diabetic
/opt /co n da /lib/python3.10/site-packages/s k learn/base.py:439: User
Warning: X does not have valid feature names, but Logistic Regression was
fitted with feature names
Warnings   warn (
In [  ]  :
```

3.FEATURE SELECTION

We will select relevant features that can impact diabetes risk prediction.

Feature selection is the process of identifying and selecting the most relevant and informative features from a dataset for use in a machine learning model. This is an important step in developing an AI-based diabetes prediction system, as it can help to improve the accuracy and efficiency of the model.
There are a number of different feature selection algorithms that can be used, and the best algorithm to use will depend on the specific dataset and machine learning model being used. Some common feature selection algorithms include:

- Filter methods: These methods rank features based on a statistical measure of their importance, such as information gain or correlation.
- Wrapper methods: These methods evaluate features by training and testing a machine learning model on different subsets of features.
- Embedded methods: These methods incorporate feature selection into the machine learning process itself.

4.MODEL SELECTION

Model selection is the process of choosing the best machine learning model for a given dataset and task. In the context of AI-based diabetes prediction, model selection involves evaluating the performance of different machine learning models on a held-out test set.

Once the most relevant features have been selected, they can be used to train a machine learning model to predict diabetes. There are a number of different machine learning models that can be used for diabetes prediction, such as:

- Logistic regression: This is a simple but effective model for binary classification tasks, such as predicting whether or not someone has diabetes.
- Support vector machines (SVMs): SVMs are a powerful machine learning model that can be used for both classification and regression tasks.
- Decision trees: Decision trees are a type of machine learning model that can be used to create rules for predicting diabetes.
- Random forests: Random forests are an ensemble machine learning model that combines the predictions of multiple decision trees to improve accuracy.

5.EVALUATION:

Some common metrics used to evaluate the performance of machine learning models for diabetes prediction include:

- Accuracy: This metric measures the proportion of predictions that are correct.
- Sensitivity: This metric measures the proportion of people with diabetes who are correctly predicted to have diabetes.
- Specificity: This metric measures the proportion of people without diabetes who are correctly predicted not to have diabetes.
- AUC-ROC: This metric measures the area under the receiver operating characteristic (ROC) curve, which is a plot of sensitivity against 1 - specificity.

The best machine learning model for AI-based diabetes prediction will depend on the specific dataset and the desired trade-off between sensitivity and specificity.

6.ITERATIVE IMPROVEMENT:

We will fine tune the model parameters and explore techniques like feature engineering to enhance prediction accuracy.

ADDITIONAL CONSIDERATION:

In addition to the tasks listed above, there are a few other things to keep in mind when collecting and preprocessing data for an AI-based diabetes prediction system:

a)Data privacy and security: It is important to ensure that the data is collected and stored in a secure manner, and that the privacy of the patients is protected.

b)Data bias: It is important to be aware of the potential for bias in the data. For example, if the data is collected from a single hospital or clinic, it may be biased towards patients with certain characteristics.

Data imbalance: Diabetes is a relatively rare disease, so it is common for datasets of patients with and without diabetes to be imbalanced. This means that there are many more patients without diabetes than with diabetes. Imbalanced

datasets can make it difficult for machine learning models to learn to predict diabetes accurately.
There are a variety of ways to handle imbalanced datasets, such as oversampling the minority class (patients with diabetes) or under sampling the majority class (Patients without diabetes).
By following these guidelines, you can collect and preprocess data for an AI-based diabetes prediction system that is high quality and suitable for machine learning.

CONCLUSION:

It is also important to note that AI-based diabetes prediction systems should be carefully evaluated and validated before being used in clinical practice. This is because these systems can have a significant impact on people's lives, and it is important to ensure that they are accurate and reliable . In Phase 1 ,we have established a clear understanding of our goal :to predict diabetes of given data using machine language .We have outlined a structured approach that includes data collection ,data preprocessing, feature selection, model selection, evaluation and iterative improvement. This sets the stage for our project’s successful execution in subsequent phases.