AIM to predict weather diabetes have or not

```
from google.colab import drive
```

```
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
import pandas as pd #read dataset
import numpy as np #numeric python
import matplotlib.pyplot as plt #plot the graph
import seaborn as sns #plot the graph in graphical
```

```
#read dataset
df = pd.read_csv("/content/drive/MyDrive/data/diabetes.csv")
```

```
df.head() #to display the dataset first 5
```

	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

Next steps: Generate code with df | View recommended plots

```
df.shape #how many rows and column
```

```
(768, 9)
```

```
df.info() #data set with data type and null value
```

```
<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.isnull() #check null value
```

	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
...	...	...	...	...	...	...	...	...	...
763	False	False	False	False	False	False	False	False	False
764	False	False	False	False	False	False	False	False	False
765	False	False	False	False	False	False	False	False	False
766	False	False	False	False	False	False	False	False	False
767	False	False	False	False	False	False	False	False	False

768 rows × 9 columns

```
df.isnull().sum() #check null value
```

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

```
#Train and Test split
from sklearn.model_selection import train_test_split
```

```
x=df.iloc[:,df.columns!='Outcome'] #pragnacy to age not outcome (row,columns)
y=df.iloc[:,df.columns=='Outcome'] #only outcome
```

```
print(x)
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6
1              1       85             66             29        0  26.6
2              8      183             64              0        0  23.3
3              1       89             66             23       94  28.1
4              0      137             40             35      168  43.1
..           ...      ...            ...            ...      ...   ...
763           10      101             76             48      180  32.9
764            2      122             70             27        0  36.8
765            5      121             72             23      112  26.2
766            1      126             60              0        0  30.1
767            1       93             70             31        0  30.4

     DiabetesPedigreeFunction  Age
0                       0.627   50
1                       0.351   31
2                       0.672   32
3                       0.167   21
4                       2.288   33
..                        ...  ...
763                     0.171   63
```

```
764                     0.340   27
765                     0.245   30
766                     0.349   47
767                     0.315   23

[768 rows x 8 columns]
```

```
print(y)
```

```
     Outcome
0          1
1          0
2          1
3          0
4          1
..       ...
763        0
764        0
765        0
766        1
767        0

[768 rows x 1 columns]
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

```
x_train.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
603	7	150	78	29	126	35.2	0.692	54
118	4	97	60	23	0	28.2	0.443	22
247	0	165	90	33	680	52.3	0.427	23
157	1	109	56	21	135	25.2	0.833	23
468	8	120	0	0	0	30.0	0.183	38

Next steps: Generate code with x_train View recommended plots

```
x_test.shape
```

```
(154, 8)
```

Suggested code may be subject to a license | pt.linkedin.com/pulse/sele%C3%A7%C3%A3o-de-atributos-com-python···

```
#Algorithms
from sklearn.ensemble import RandomForestClassifier
```

Suggested code may be subject to a license | AP-State-Skill-Development-Corporation/Machine-Learning-Using-Pyth···

```
model=RandomForestClassifier()
```

Suggested code may be subject to a license | bhaveshlohana/HacktoberFest2020-Contributions

```
model.fit(x_train,y_train.values.ravel()) #to train algorithm
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
predict_output = model.predict(x_test) #to test algorithm
print(predict_output)
```

```
[1 0 0 1 0 0 1 1 0 0 1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 1 1
 0 0 1 0 0 0 1 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 1 1 0 1 0 1 0 0 0 0 1
 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0
 0 1 1 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 1 0 0 1 0 0 0
 0 0 0 0 0 0]
```

+ Code + Text

```
y_test.head()
```

	Outcome
661	1
122	0
113	0
14	1
529	0

Next steps: Generate code with y_test View recommended plots

```
#compare the actual output and predict output
from sklearn.metrics import accuracy_score


acc = accuracy_score(y_test,predict_output)
print(acc)
```

```
0.8051948051948052
```

Start coding or generate with AI.