AI BASED DIABETES PREDICTION SYSTEM

Phase 2: Innovation

Introduction:

Diabetes is a health condition that affects how your body turns food into energy. Most of the food you eat is broken down into sugar (also called glucose) and released into your bloodstream. When your blood sugar goes up, it signals your pancreas to release insulin.

Without ongoing, careful management, diabetes can lead to a buildup of sugars in the blood, which can increase the risk of dangerous complications, including stroke and heart disease. So that i decide to predict using Machine Learning in Python.

Objectives:

1. Predict if person is diabetes patient or not
2. Find most indicative features of diabetes
3. Try different classification methods to find highest accuracy

Installing Libraries:

In this first step I have imported most common libraries used in python for machine learning such as Pandas, Seaborn, Matplitlib etc.

I am using Python because if very flexible and effective programming language i ever used. I used Python in software development field too.

```
# Import libraries
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns # for data visualization
import matplotlib.pyplot as plt # to plot charts
from collections import Counter
import os

# Modeling Libraries
from sklearn.preprocessing import QuantileTransformer
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier,
```

```
GradientBoostingClassifier, VotingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV, cross_val_score, StratifiedKFold,
learning_curve, train_test_split
```

The sklearn library is very versatile and handy and serves real-world purposes. It provides wide range of ML algorithms and Models.

Importing Data:

```
# Import dataset
df = pd.read_csv("../input/pima-indians-diabetes-database/diabetes.csv")
# Get familier with dataset structure
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
```

Excepting BMI and DiabetesPedigreeFunction all the columns are integer. Outcome is the label containing 1 and 0 values. 1 means person has diabetes and 0 mean person is not diabetic.

 # Show top 5 rows
df.head()

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148.0	72.0	35	30.5	33.6	0.627	50	1
1	1	85.0	66.0	29	30.5	26.6	0.351	31	0
2	8	183.0	64.0	23	30.5	23.3	0.672	32	1
3	1	89.0	66.0	23	94.0	28.1	0.167	21	0
4	0	137.0	40.0	35	168.0	43.1	2.288	33	1

Missing Value Analysis:

Next, I will cleanup the dataset which is the important part of data science. Missing data can lead to wrong statistics during modeling and predictions.

 # Explore missing values
df.isnull().sum()

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

```
# Review dataset statistics
df.describe()
```

	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	27.334635	94.652344	32.450911	0.471876	33.240885	0.348958
std	3.369578	30.436016	12.115932	9.229014	105.547598	6.875366	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
25%	1.000000	99.750000	64.000000	23.000000	30.500000	27.500000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	31.250000	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

RandomForestClassifier:

```
# Define models and parameters for LogisticRegression
model = RandomForestClassifier(random_state=42)# Define grid
search
tuned_parameters = {
    'n_estimators': [200, 500],
    'max_features': ['auto', 'sqrt', 'log2'],
    'max_depth' : [4,5,6,7,8],
    'criterion' :['gini', 'entropy']
}
cv = StratifiedKFold(n_splits = 2, random_state = 1, shuffle =
True)
grid_search = GridSearchCV(estimator = model, param_grid =
tuned_parameters, cv = cv, scoring = 'accuracy', error_score =
0)
grid_result = grid_search.fit(x_train, y_train)# SVC
Hyperparameter Result
analyze_grid_result(grid_result)
```

Output:

```
Tuned hyperparameters: (best parameters) {'criterion':
'entropy', 'max_depth': 5, 'max_features': 'log2',
'n_estimators': 200}
Accuracy : 0.7663648051875454
Detailed classification report:

precision recall f1-score support

0 0.78 0.83 0.80 147
1 0.66 0.58 0.62 83

accuracy 0.74 230
macro avg 0.72 0.70 0.71 230
weighted avg 0.73 0.74 0.74 230
```

Randomforest model gave max 0.76% accuracy which is not best comparing to other model. So i decided to use LogisticRegression Model for prediction.

Prediction:

Till now, I worked on EDA, Feature Engineering, Cross Validation of Models, and Hyperparameter Tuning and find the best working Model for my dataset. Next, I did prediction from my test dataset and storing the result in CSV.

```
# Test predictions
y_pred = logi_result.predict(x_test)
print(classification_report(y_test, y_pred))
# output
              precision    recall  f1-score   support

           0       0.78      0.84      0.81       147
           1       0.68      0.58      0.62        83

    accuracy                           0.75       230
   macro avg       0.73      0.71      0.72       230
weighted avg       0.74      0.75      0.74       230
```

Finally append new feature column in test dataset called Prediction and print the dataset.

```
x_test['pred'] = y_pred
print(x_test)
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	pred
236	7	181.0	84.0	21	192.0	35.9	0.586	51	1
715	7	187.0	50.0	33	392.0	33.9	0.826	34	1
766	1	126.0	60.0	23	30.5	30.1	0.349	47	0
499	6	154.0	74.0	32	193.0	29.3	0.839	39	1
61	8	133.0	72.0	23	30.5	32.9	0.270	39	1
...	...	...	...	...	...	...	...	...	...
189	5	139.0	80.0	35	160.0	31.6	0.361	25	0
351	4	137.0	84.0	23	30.5	31.2	0.252	30	0
120	0	162.0	76.0	56	100.0	53.2	0.759	25	1
108	3	83.0	58.0	31	18.0	34.3	0.336	25	0
637	2	94.0	76.0	18	66.0	31.6	0.649	23	0

230 rows x 9 columns

Conclusion:

1. Diabetes is one of the ricks during Pregnancy. It has to be treat to avoid complications.
2. BMI index can help to avoid complications of diabetes a way before
3. Diabetes start showing in age of 35 – 40 and increase with person age.