diabetic_knn.py

```
# -*- coding: utf-8 -*-
"""
Created on Wed Sep 21 16:42:12 2022

@author: Admin
"""

# First let's start with calling all the dependencies for this project
import numpy as np
import pandas as pd
import math
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('G:/dypiemr2/dypiemr22-23/sem_I/BE_I/databse/diabetes.csv')
df.head()

df.drop(['Pregnancies', 'BloodPressure', 'SkinThickness'], axis=1,
inplace=True)
df.info()

df.describe().T
#aiming to impute nan values for the columns in accordance
#with their distribution
df[['Glucose','Insulin','BMI']].replace(0,np.NaN)

columns = ['Glucose','Insulin','BMI']
for col in columns:
val = df[col].mean()
df[col].replace(0, val)

#plot graph
graph = ['Glucose','Insulin','BMI','Age','Outcome']
sns.set()
print(sns.pairplot(df[graph],hue='Outcome', diag_kind='kde'))

#separate outcome or target col
X = df.drop(['Outcome'], axis=1)
y = df['Outcome']


from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score


X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.2,random_state=0)



from sklearn.preprocessing import StandardScaler
```

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score

# feature scaling

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

classifier = KNeighborsClassifier(n_neighbors=11,p=2,metric='euclidean')

classifier.fit(X_train,y_train)

y_pred = classifier.predict(X_test)

# evaluating model

conf_matrix = confusion_matrix(y_test,y_pred)
print(conf_matrix)
print(f1_score(y_test,y_pred))
# accuracy

print(accuracy_score(y_test,y_pred))

# roc curve
from sklearn.metrics import roc_curve
plt.figure(dpi=100)
fpr, tpr, thresholds = roc_curve(y_test, y_pred)
from sklearn.metrics import roc_auc_score

temp=roc_auc_score(y_test,y_pred)

plt.plot(fpr,tpr,label = "%.2f" %temp)
plt.legend(loc = 'lower right')
plt.grid(True)
```