```
# -*- coding: utf-8 -*-
"""diabetes_prediction

Automatically generated by Colab.

Original file is located at
    https://colab.research.google.com/drive/1JPMVrxUJGvLfra3StzVEv5W7-ldLNnJx

# **Name : Oza Tirth Nareshbhai**
# **Div : 5IT - A**
# **Enrollment : 2303031080006**
"""

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

df = pd.read_csv('./diabetes.csv')
df.head()

df.columns

df.isnull().sum()

df.info()

df.corr().T

sns.heatmap(df.corr() , annot=True)

(df == 0).sum()

cols = ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI']

for col in cols:
  df[col] = pd.to_numeric(df[col], errors='coerce')
  df[col] = df[col].replace(0 , np.nan)
  df[col].fillna(df[col].median() , inplace=True)


df.head()

(df==0).sum()

sns.heatmap(df.corr() , annot=True,cmap='coolwarm',fmt='.2f')

plt.figure(figsize=(12,6))
sns.boxplot(data=df.drop('Outcome' , axis=1))

sns.boxplot(y=df['Insulin'])
plt.show()

from sklearn.model_selection import train_test_split

x = df.drop('Outcome' , axis=1)
y = df['Outcome']

x_train , x_test , y_train , y_test = train_test_split(x , y , test_size=0.2 , random_state=42)

from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier

model = {
    'LogisticRegression' : LogisticRegression() ,
    "DecisionTreeClassifier" : DecisionTreeClassifier(),
    "RandomForestClassifier" : RandomForestClassifier(),
    "SVC" : SVC(),
    "KNeighborsClassifier" : KNeighborsClassifier()
}


for name , model in model.items():
   model.fit(x_train , y_train)
   print(f'{name} accuracy : {model.score(x_test , y_test)}')

from sklearn.metrics import classification_report , confusion_matrix , roc_auc_score

y_pred = model.predict(x_test)
conf = confusion_matrix(y_test , y_pred)
print(conf)

print(classification_report(y_test , y_pred))

cm = conf
sns.heatmap(cm , annot=True , fmt='d' , cmap='Blues' , xticklabels=['Healthy(0)','Diabetes(1)'] , yticklabels=['Healthy(0)','Diabetes(1)'])
plt.xlabel("predicted")
plt.ylabel("Actual")
plt.title("Confusion matrix")
plt.show()
```