Code of the project

SVM algorithm using for prediction of diabetes:

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score

#loading the diabetes dataset to a pandas DataFrame
diabetes_dataset = pd.read_csv('/content/diabetes.csv')

#printing the first 5 rows of the datase
diabetes_dataset.head()

#number of rows and Columns in this dataset
diabetes_dataset.shape

# getting the statistical measures of the data
diabetes_dataset.describe()

diabetes_dataset['Outcome'].value_counts()
diabetes_dataset.groupby('Outcome').mean()

# separating the data and labels
X = diabetes_dataset.drop(columns = 'Outcome', axis=1)
Y = diabetes_dataset['Outcome']

print(X)
print(Y)
scaler = StandardScaler()
scaler.fit(X)
standardized_data = scaler.transform(X)
print(standardized_data)
X = standardized_data
Y = diabetes_dataset['Outcome']
print(X)
print(Y)
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.
2, stratify=Y, random_state=2)
print(X.shape, X_train.shape, X_test.shape)
classifier = svm.SVC(kernel='linear')
#training the support vector Machine Classifier
classifier.fit(X_train, Y_train)
# accuracy score on the training data
X_train_prediction = classifier.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

```
print('Accuracy score of the training data : ', training_data_accuracy)

# accuracy score on the test data
X_test_prediction = classifier.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)

print('Accuracy score of the test data : ', test_data_accuracy)

input_data = (5,166,72,19,175,25.8,0.587,51)

# changing the input_data to numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the array as we are predicting for one instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

# standardize the input data
std_data = scaler.transform(input_data_reshaped)
print(std_data)

prediction = classifier.predict(std_data)
print(prediction)

if (prediction[0] == 0):
print('The person is not diabetic')
else:
print('The person is diabetic')
```

Logistic Regression algorithm using for prediction of diabetes:

```
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
import matplotlib.pyplot as plt
%matplotlib inline

df1=pd.read_csv("diabetes.csv")
In [3]:
df1.head()
df1.describe()
sns.heatmap(df1.isnull(),yticklabels=False,cmap='viridis')
sns.heatmap(df1,yticklabels=False,cmap='viridis')
sns.set_style('whitegrid')
sns.countplot(x='Outcome',hue='Outcome',data=df1,palette='cubehelix')
plt.scatter(x='Outcome',y='Age',data=df1)
plt.ylabel('Age')
plt.xlabel('Outcome')
sns.distplot(df1['Age'],kde=False,color='darkblue',bins=30)
sns.distplot(df1['BloodPressure'],kde=False,color='royalblue',bins=20)
sns.jointplot(x='Age',y='BloodPressure',data=df1)
import seaborn as sns
sns.set(style="whitegrid")
#tips=sns.load_dataset("diabetes.csv")
plt.figure(figsize=(15,8))

ax=sns.barplot(x="Age", y="BloodPressure", data=df1,)
from sklearn.model_selection import train_test_split
In [16]:
df1.head()

x=['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI',
'DiabetesPedigreeFunction','Age']
y=['Output']
df2=pd.DataFrame(data=df1)
df2.head()
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(df1.drop('Outcome',axis=1),d
f1['Outcome'],test_size=0.20,random_state=101)
X_test.head()
from sklearn.linear_model import LogisticRegression
LRModel=LogisticRegression(solver='lbfgs', max_iter=7600)
LRModel.fit(X_train,y_train)
predictions_diabetes=LRModel.predict(X_test)
from sklearn.metrics import classification_report, confusion_matrix
print(classification_report(y_test,predictions_diabetes))
# paitentid_54=pd.DataFrame([1,123,126,60,0,30.1,0.349,47],columns=x)
#Defining a sample data to test the model
x=['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI',
'DiabetesPedigreeFunction','Age']
data=[0,170,126,60,35,30.1,0.649,78]
paitentid_54=pd.DataFrame([data],columns=x)
paitentid_54.head()
df1.head()
predictions_diabetes=LRModel.predict(paitentid_54)
print(predictions_diabetes)
```

KNN algorithm using for prediction of diabetes:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
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
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
data = pd.read_csv('C:\dataset\diabetes.csv')
data.head()
zero_not_accepted =
['Glucose','BloodPressure','SkinThickness','BMI','Insulin']
# for col in zero_not_accepted:
```

```
#     for i in data[col]:
#         if i==0:
#             colSum = sum(data[col])
#             meanCol=colSum/len(data[col])
#             data[col]=meanCol

for col in zero_not_accepted:
    data[col]= data[col].replace(0,np.NaN)
    mean = int(data[col].mean(skipna=True))
    data[col] = data[col].replace(np.NaN,mean)
X = data.iloc[:,0:8]
y = data.iloc[:,8]
sns.heatmap(data.corr())
plt.figure(figsize=(25,7))
sns.countplot(x='Age',hue='Outcome',data=data,palette='Set1')
X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.2,random_state=0)
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)
conf_matrix = confusion_matrix(y_test,y_pred)
print(conf_matrix)
print(f1_score(y_test,y_pred))
print(accuracy_score(y_test,y_pred))
```