Hello guyz,

Welcome to build a diabetes prediction model.

Domain:- Machine Learning

* **Step 1 :**pip install \_\_\_\_\_\_\_\_  
  /\* open cmd prompt and download few libraries using above mentioned code by simply replacing the library name with the blank \*/

/\*

numpy

pandas

seaborn

matplotlib

\*/

* **Step 2 :**open jupyter notebook  
  /\* here I am using anaconda \*/
* **Step 3 :**
* **Cell 1:-**
* import pandas as pd
* import numpy as np
* import seaborn as sns
* import matplotlib.pyplot as plt
* %matplotlib inline

Run the entire cell

* **Cell 2:-**
* dataset = pd.read\_csv("diabetes.csv")
* dataset

Run the entire cell

* **Cell 3:-**
* dataset.info()

Run the entire cell

* **Cell 4:-**
* dataset.isnull().sum()

Run the entire cell

* **Cell 5:-**
* dataset.describe()

Run the entire cell

* **Cell 6:-**
* plt.figure(figsize=(10, 8))
* sns.heatmap(dataset.corr(), annot=True, fmt=".3f", cmap="YlGnBu")
* plt.title("Correlation Heatmap")

Run the entire cell

* **Cell 7:-**
* plt.figure(figsize=(10, 8))
* kde = sns.kdeplot(dataset["Pregnancies"][dataset["Outcome"]==1], color="Red", fill=True)
* kde = sns.kdeplot(dataset["Pregnancies"][dataset["Outcome"]==0], color="Blue", fill=True)
* kde.set\_xlabel("Pregnancies")
* kde.set\_ylabel("Density")
* kde.legend(["Positive", "Negative"])

Run the entire cell

* **Cell 8:-**
* plt.figure(figsize=(10, 8))
* sns.violinplot(data=dataset,x="Outcome",y="Glucose",split=True,linewidth=2,inner="quart")

Run the entire cell

* **Cell 9:-**
* plt.figure(figsize=(10, 8))
* kde = sns.kdeplot(dataset["Glucose"][dataset["Outcome"]==1], color="Red", fill=True)
* kde = sns.kdeplot(dataset["Glucose"][dataset["Outcome"]==0], color="Blue", fill=True)
* kde.set\_xlabel("Glucose")
* kde.set\_ylabel("Density")
* kde.legend(["Positive", "Negative"])

Run the entire cell

* **Cell 10:-**
* dataset["Glucose"] = dataset["Glucose"].replace(0,dataset["Glucose"].median())
* dataset["BloodPressure"] = dataset["BloodPressure"].replace(0,dataset["BloodPressure"].median())
* dataset["BMI"] = dataset["BMI"].replace(0,dataset["BMI"].mean())
* dataset["SkinThickness"] = dataset["SkinThickness"].replace(0,dataset["SkinThickness"].mean())
* dataset["Insulin"] = dataset["Insulin"].replace(0,dataset["Insulin"].mean())

Run the entire cell

* **Cell 11:-**
* dataset

Run the entire cell

* **Cell 12:-**
* X = dataset.drop(["Outcome"],axis=1)
* y = dataset["Outcome"]

Run the entire cell

* **Cell 13:-**
* X

Run the entire cell

* **Cell 14:-**
* y

Run the entire cell

* **Cell 15:-**
* from sklearn.model\_selection import train\_test\_split

Run the entire cell

* **Cell 16:-**
* X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.33,random\_state=42)

Run the entire cell

* **Cell 17:-**
* X\_train

Run the entire cell

* **Cell 18:-**
* from sklearn.neighbors import KNeighborsClassifier

Run the entire cell

* **Cell 19:-**
* training\_accuracy = []
* test\_accuracy = []
* for n\_neighbors in range(1, 11):
* knn = KNeighborsClassifier(n\_neighbors=n\_neighbors)
* knn.fit(X\_train, y\_train)
* training\_accuracy.append(knn.score(X\_train, y\_train))
* test\_accuracy.append(knn.score(X\_test, y\_test))

Run the entire cell

* **Cell 20:-**
* plt.plot(range(1,11),training\_accuracy,label="training\_accuracy")
* plt.plot(range(1,11),test\_accuracy,label="test\_accuracy")
* plt.ylabel("Accuracy")
* plt.xlabel("n\_neighbors")
* plt.legend()

Run the entire cell

* **Cell 21:-**
* knn = KNeighborsClassifier(n\_neighbors=9)
* knn.fit(X\_train,y\_train)
* print(knn.score(X\_train,y\_train),"; Training accuracy")
* print(knn.score(X\_train,y\_train),"; Test accuracy")

Run the entire cell

/\* here in this phase we are getting an accuracy using K-Nearest Neighbour Approach \*/

* **Step 4 :**

Upto here the model is ready now further more steps can be reffered from the notebook shared by me for other approaches also to see there accuracy.

