# AD3002 - HEALTH CARE ANALYTICS Assignment-2

**Diabetes Analysis and KNN**

Import / Data Cleaning Importing packages

INPUT :

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.neighbors import KNeighborsClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import confusion\_matrix

Import of dataset + Data Cleaning

INPUT :

df = pd.read\_csv('/content/drive/MyDrive/diabetes.csv')

df.rename(columns={'DiabetesPedigreeFunction': 'DPF'}, inplace= True)

to\_nan = ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin']

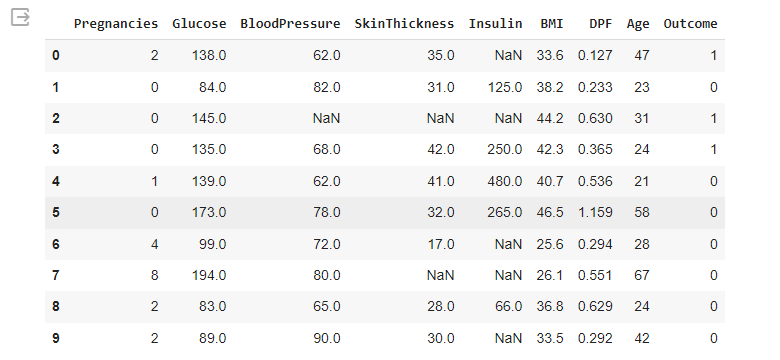
to\_nan.append(['BMI', 'DPF', 'Age'])

for i in range(len(to\_nan)):

    df[to\_nan[i]] = df[to\_nan[i]].replace(0, np.nan)

df.head(10)

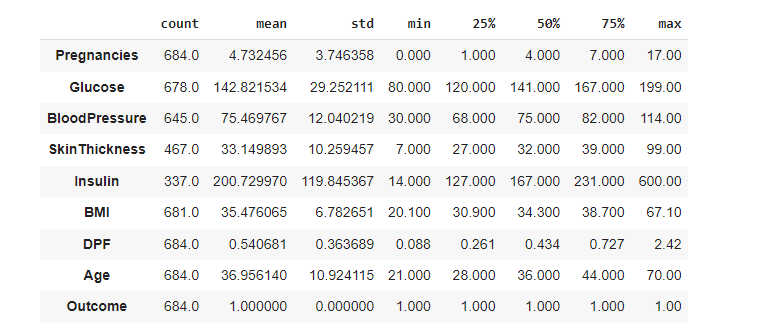
OUTPUT :



INPUT :

df\_true = df[(df.Outcome>0)]

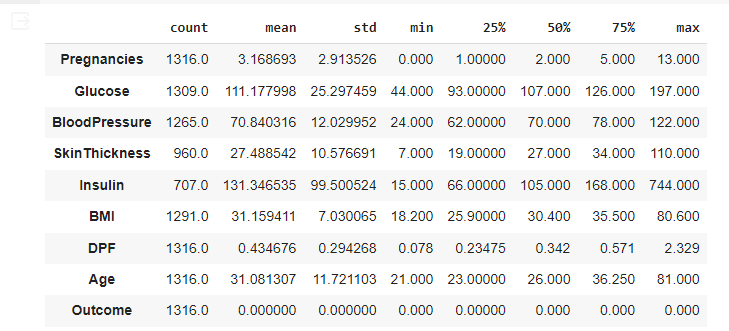
df\_true.describe().T



INPUT :

df\_false = df[(df.Outcome<1)]

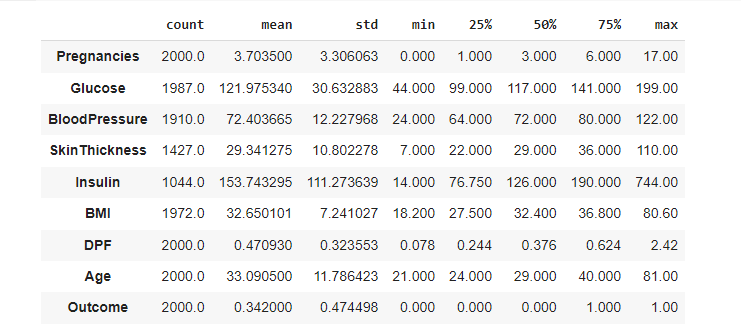
df\_false.describe().T

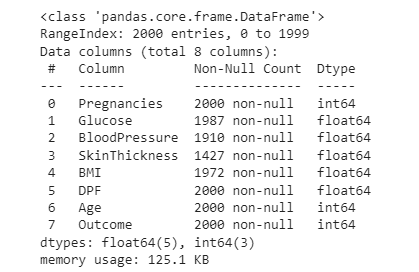


df.describe().T

del df['Insulin']

df.info()





INPUT :

bmi\_to\_out = df[['BMI','Outcome']]

print(bmi\_to\_out.corr())

print("")

fig, axes = plt.subplots(1, 2, figsize=(16, 5), sharey=False, sharex=False)

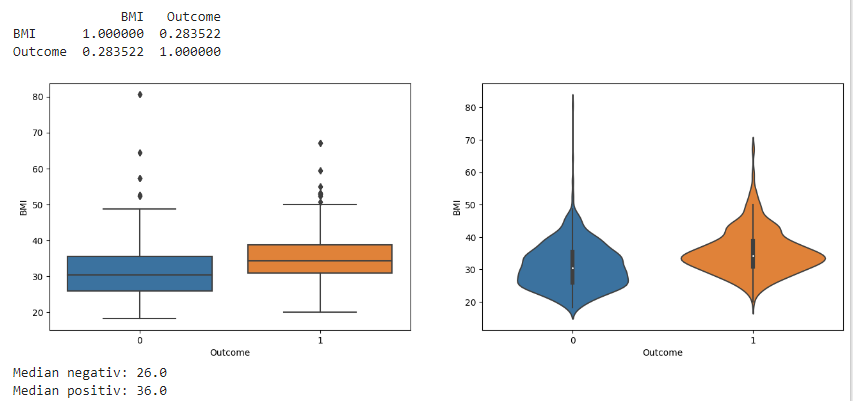
sns.boxplot(x= df['Outcome'], y = df['BMI'], ax= axes[0])

sns.violinplot(x= df['Outcome'], y = df['BMI'], ax= axes[1])

plt.show()

print('Median negativ:', df\_false.Age.median())

print('Median positiv:', df\_true.Age.median())



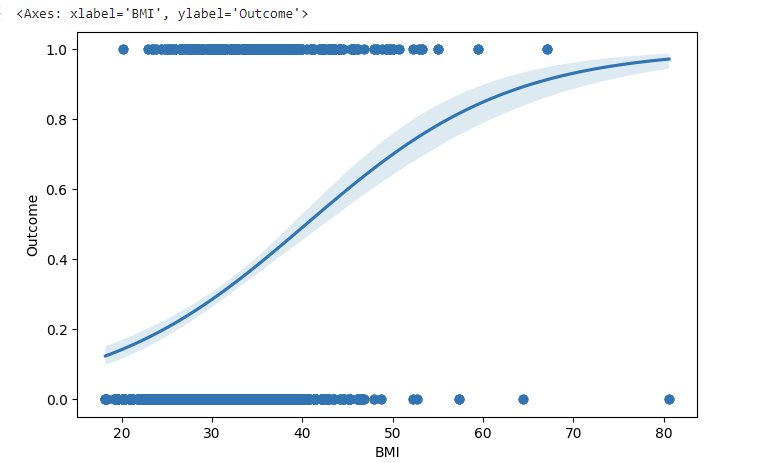
f = plt.figure()

f.set\_figwidth(8)

f.set\_figheight(5)

plt.scatter(bmi\_to\_out['BMI'], bmi\_to\_out['Outcome'])

sns.regplot(data = df, x = df['BMI'], y = df['Outcome'], logistic = True)



f = plt.figure()

f.set\_figwidth(16)

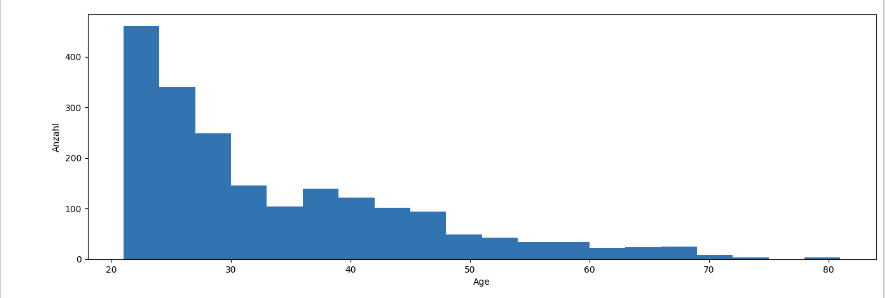
f.set\_figheight(5)

plt.hist(df['Age'], histtype = 'bar', bins = 20)

plt.xlabel('Age')

plt.ylabel('Anzahl')

plt.show()



age\_to\_out = df[['Age','Outcome']]

print(age\_to\_out.corr())

print('')

fig, axes = plt.subplots(1, 2, figsize=(16, 5), sharey=False, sharex=False)

sns.boxplot(x = df['Outcome'], y = df['Age'], ax = axes[0])

sns.violinplot(x = df['Outcome'], y = df['Age'], ax = axes[1])

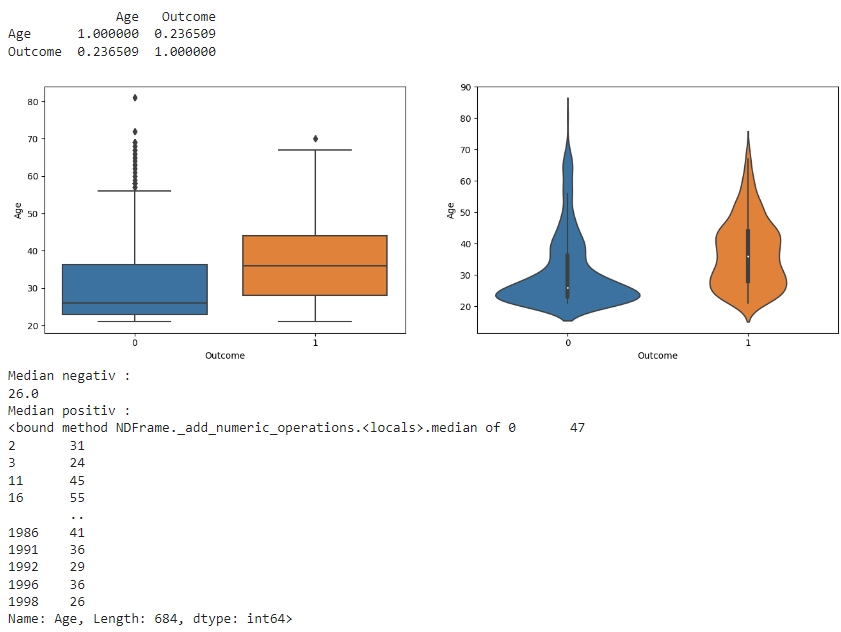
plt.show()

print('Median negativ :')

print(df\_false.Age.median())

print('Median positiv :')

print(df\_true.Age.median)



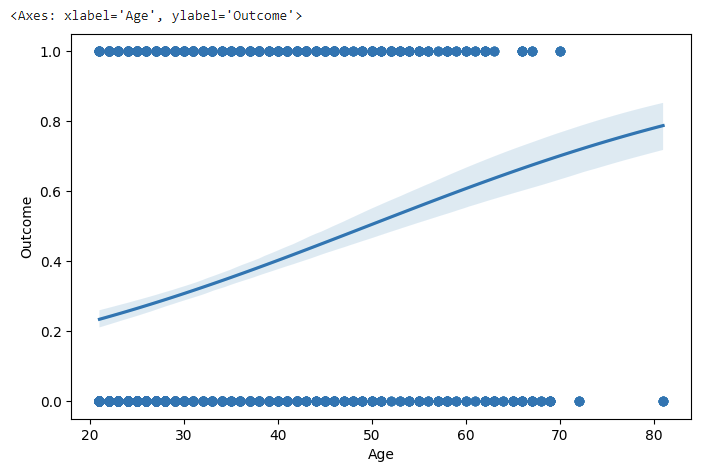
f = plt.figure()

f.set\_figwidth(8)

f.set\_figheight(5)

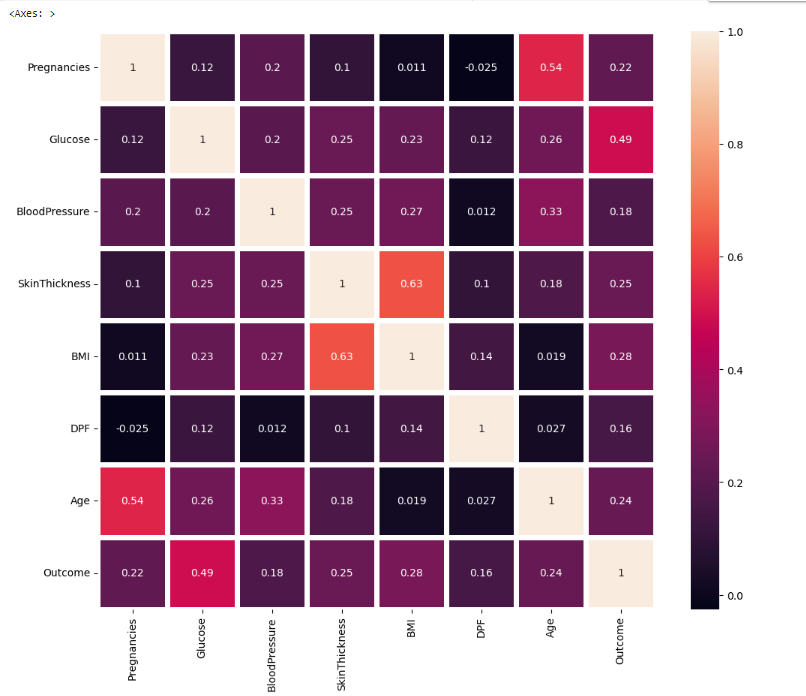
plt.scatter(age\_to\_out['Age'], age\_to\_out['Outcome'])

sns.regplot(data = df, x = df['Age'], y = df['Outcome'], logistic = True)



fig, axes = plt.subplots(1, 1, figsize=(12, 10), sharey=False, sharex=False)

sns.heatmap(df.corr(), annot= True, linewidth = 5)



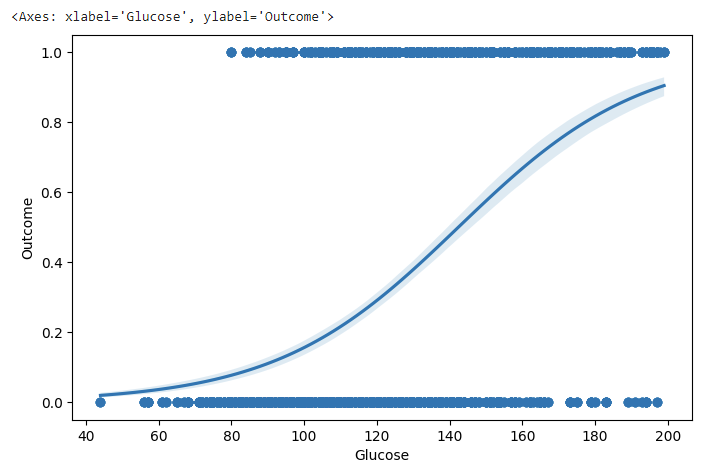
f = plt.figure()

f.set\_figwidth(8)

f.set\_figheight(5)

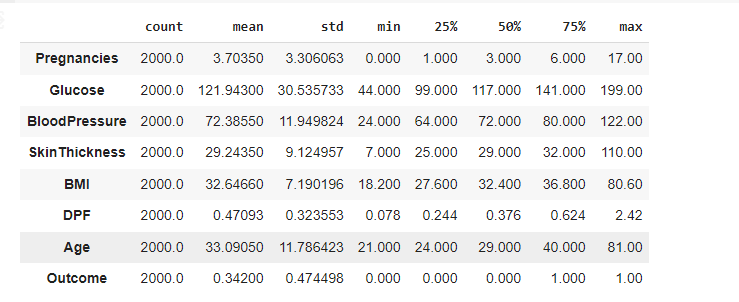
plt.scatter(df['Glucose'], df['Outcome'])

sns.regplot(data = df, x = df['Glucose'], y = df['Outcome'], logistic = True)



df.fillna(df.median(), inplace=True)

df.describe().T



inp = df[["Pregnancies","BloodPressure","SkinThickness","BMI","DPF","Age"]]

out = df["Outcome"]

x\_train, x\_test, y\_train, y\_test = train\_test\_split(inp,out,test\_size=0.2)

results = []

recallresults = []

for k in range (1, 100, 2):

    classifier = KNeighborsClassifier(n\_neighbors=k)

    classifier.fit(x\_train,y\_train)

    prediction = classifier.predict(x\_test)

    tp = sum(prediction[y\_test==1]==1)

    fn = sum(prediction[y\_test==1]==0)

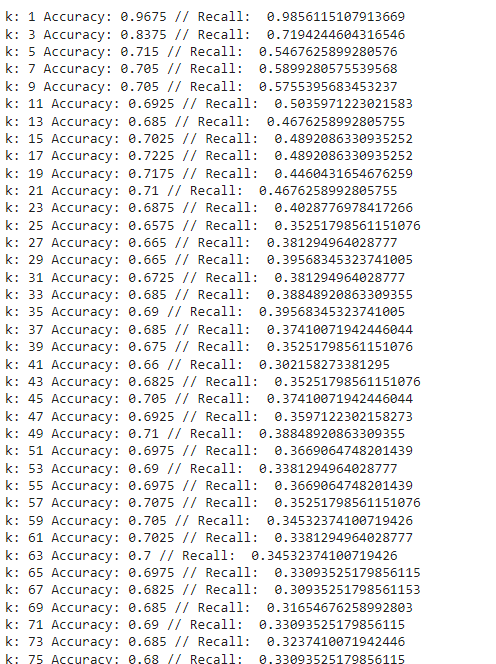
    accuracy = sum(prediction==y\_test)/len(y\_test)

    recall =  tp/(tp+fn)

    print("k:",k,"Accuracy:",accuracy, "// Recall: ", recall)

    results.append([k,accuracy])

    recallresults.append([k,recall])



results = pd.DataFrame(results, columns=["k","accuracy"])

resultsrecall = pd.DataFrame(recallresults, columns=['k','recall'])

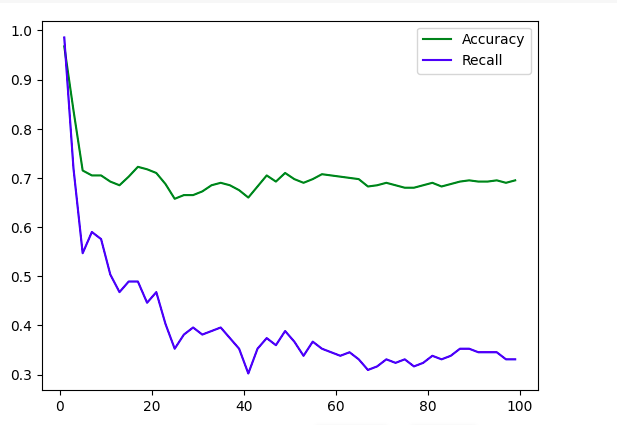
plt.figure()

plt.plot(results.k, results.accuracy, "green", label= 'Accuracy')

plt.plot(resultsrecall.k, resultsrecall.recall, 'blue', label= 'Recall')

plt.legend(loc='upper right')

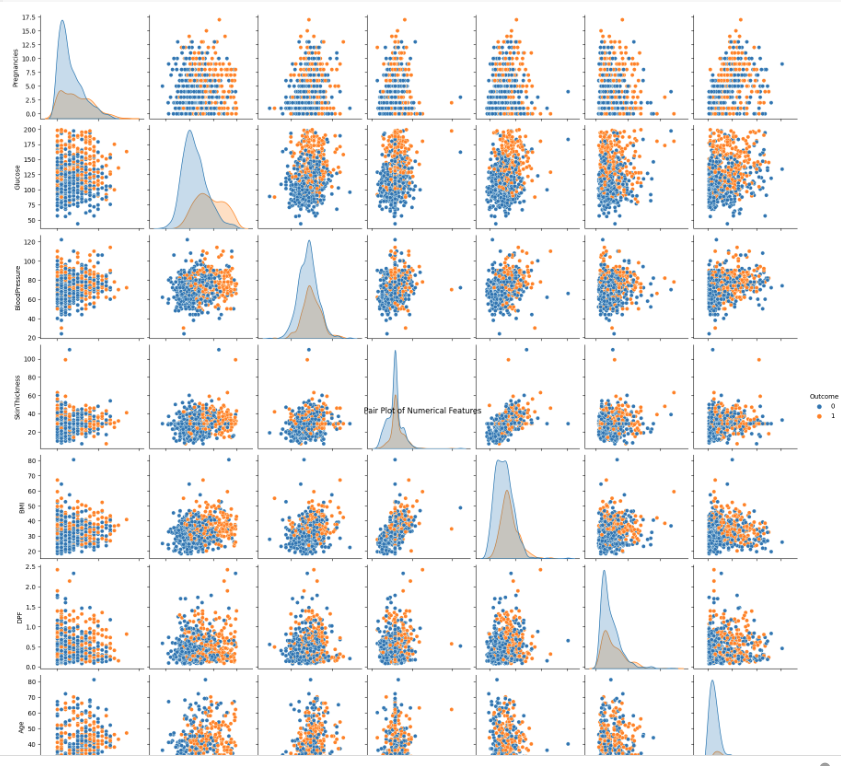
plt.show()



sns.pairplot(df, hue='Outcome', diag\_kind='kde')

plt.suptitle('Pair Plot of Numerical Features', y=1.02)

plt.show()



plt.figure(figsize=(12, 8))

for i, column in enumerate(df.columns[:-1]):

    plt.subplot(2, 4, i + 1)

    sns.boxplot(x='Outcome', y=column, data=df)

    plt.title(f'Box Plot of {column}')

plt.tight\_layout()

plt.show()

