def RemoveOutlier(df, var):

Q1 = df[var].quantile(0.25)

Q3 = df[var].quantile(0.75)

IQR = Q3 - Q1

high, low = Q3 + 1.5 \* IQR, Q1 - 1.5 + IQR

df = df[((df[var] >= low) & (df[var] <= high))]

return df

def DisplayOutlier(df, msg):

fig,axes = plt.subplots(2, 2)

fig.suptitle(msg)

sns.boxplot(data=df, x="sepal.length", ax=axes[0,0])

sns.boxplot(data=df, x="sepal.width", ax=axes[0,1])

sns.boxplot(data=df, x="petal.length", ax=axes[1,0])

sns.boxplot(data=df, x="petal.width", ax=axes[1,1])

fig.tight\_layout()

plt.show()

# import libraries

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Read Dataset

df = pd.read\_csv(r"iris - 2023-05-20T153708.481.csv")

print("Iris dataset is successfully loaded")

# Display information of dataset

print("Information of Dataset:\n", df.info)

print("Shape of Dataset row x column):", df.shape)

print("Columns Name:", df.columns)

print("Total elements in dataset:", df.size)

print("Datatype of attributes (columns):", df.dtypes)

print("First 5 rows: \n", df.head().T)

print("last 5 rows:\n", df.tail().T)

print("Any 5 rows: \n", df.sample(5).T)

# Find missing values

print("Missing values")

print(df.isnull().sum())

#Finding and removing outliers

print("Finding and removing outliers: ")

DisplayOutlier(df, "Before removing Outliers")

df = RemoveOutlier(df, "sepal.length")

df = RemoveOutlier(df, "sepal.width")

df = RemoveOutlier(df, "petal.length")

df = RemoveOutlier(df, "petal.width")

DisplayOutlier(df,"After removing Outliers")

#Encoding of output variable

df["variety"]=df["variety"].astype('category')

df["variety"]=df["variety"].cat.codes

#Find correlation matrix-

print("Finding correlation matrix using heatmap:")

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

plt.show()

#Split the data into inputs and outputs

x= df.iloc[:, [0,1,2,3]].values

y= df.iloc[:, 4].values

#Training and testing data

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

#Assign test data size 20%

x\_train, x\_test, y\_train, y\_test =train\_test\_split(x,y,test\_size=0.20, random\_state=0)

#Normalization of input data

from sklearn.preprocessing import StandardScaler

sc\_X = StandardScaler()

x\_train= sc\_X.fit\_transform(x\_train)

x\_test= sc\_X.fit\_transform(x\_test)

#Apply Gaussian Naive Bayes classifier

from sklearn.naive\_bayes import GaussianNB

model=GaussianNB()

model.fit(x\_train, y\_train)

y\_pred=model.predict(x\_test)

#Display classification report

from sklearn.metrics import classification\_report

print(classification\_report(y\_test, y\_pred))

#Display confusion matrix

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(y\_test, y\_pred)

print("confusion matrix\n",cm)

fig, ax=plt.subplots(figsize=(5, 5))

sns.heatmap(cm,annot=True,linewidths=.3,cmap="Blues")

plt.show()