**Z test 2 sample:**

import pandas as pd

from statsmodels.stats import weightstats as stests

df = pd.read\_csv("blood\_pressure.csv")

df[['bp\_before','bp\_after']].describe()

print(df)

ztest ,pval = stests.ztest(df['bp\_before'],x2=df['bp\_after'], value=0,alternative='two- sided')

print(float(pval))

if pval<0.05:

  print("reject null hypothesis")

else:

  print("accept null hypothesis")

**Z Test 1 Sample:**

from statsmodels.stats import weightstats as stests

import pandas as pd

from scipy import stats

df = pd.read\_csv("blood\_pressure.csv")

df[['bp\_before','bp\_after']].describe()

print(df)

ztest ,pval = stests.ztest(df['bp\_before'], x2=None, value=156)

print(float(pval))

if pval<0.05:

  print("reject null hypothesis")

else:

  print("accept null hypothesis")

**Import Data:**

import os

import pandas as pd

Base='C:/VKHCG'

sFileDir=Base + '/01-Vermeulen/01-Retrieve/01-EDS/02-Python' #if not os.path.exists(sFileDir):

#os.makedirs(sFileDir)

CurrencyRawData = pd.read\_excel('C:/VKHCG/01-Vermeulen/00-RawData/Country\_Currency.xlsx')

sColumns = ['Country or territory', 'Currency', 'ISO-4217']

CurrencyData = CurrencyRawData[sColumns]

CurrencyData.rename(columns={'Country or territory': 'Country', 'ISO-4217': 'CurrencyCode'}, inplace=True)

CurrencyData.dropna(subset=['Currency'],inplace=True)

CurrencyData['Country'] = CurrencyData['Country'].map(lambda x: x.strip())

CurrencyData['Currency'] = CurrencyData['Currency'].map(lambda x: x.strip())

CurrencyData['CurrencyCode'] = CurrencyData['CurrencyCode'].map(lambda x: x.strip())

print(CurrencyData)

print('~~~~~~ Data from Excel Sheet Retrived Successfully ~~~~~~~ ')

sFileName=sFileDir + '/Retrieve-Country-Currency.csv'

CurrencyData.to\_csv(sFileName, index = False)

**Hypothesis using T-test:**

from scipy import stats

import matplotlib.pyplot as plt

import pandas as pd

df = pd.read\_csv("blood\_pressure.csv")

print(df[['bp\_before','bp\_after']].describe())

df[['bp\_before', 'bp\_after']].plot(kind='box')

plt.savefig('boxplot\_outliers.png')

df['bp\_difference'] = df['bp\_before'] - df['bp\_after']

df['bp\_difference'].plot(kind='hist', title= 'Blood Pressure Difference Histogram')

plt.savefig('blood pressure difference histogram.png')

stats.probplot(df['bp\_difference'], plot= plt)

plt.title('Blood pressure Difference Q-Q Plot')

plt.savefig('blood pressure difference qq plot.png')

stats.shapiro(df['bp\_difference'])

stats.ttest\_rel(df['bp\_before'], df['bp\_after'])

**Correlation:**

**positive**

import matplotlib

import numpy as np

import matplotlib.pyplot as plt

np.random.seed(1)

# 1000 random integers between 0 and 50

x = np.random.randint(0, 50, 1000)

# Positive Correlation with some noise

y = x + np.random.normal(0, 10, 1000)

np.corrcoef(x, y)

matplotlib.style.use('ggplot')

plt.scatter(x, y)

plt.show()

**negative**

import numpy as np

import matplotlib.pyplot as plt

np.random.seed(1)

# 1000 random integers between 0 and 50

x = np.random.randint(0, 50, 1000)

# Negative Correlation with some noise

y = 100 - x + np.random.normal(0, 5, 1000)

np.corrcoef(x, y)

plt.scatter(x, y)

plt.show()

**no/weak**

import numpy as np

import matplotlib.pyplot as plt

np.random.seed(1)

x = np.random.randint(0, 50, 1000)

y = np.random.randint(0, 50, 1000)

np.corrcoef(x, y)

plt.scatter(x, y)

plt.show()

**1 Sample T test:**

fromscipy.stats

import ttest\_1samp

import numpy as np

ages = np.genfromtxt('ages.csv')

print(ages)

ages\_mean = np.mean(ages)

print(ages\_mean)

tset, pval = ttest\_1samp(ages, 30)

print('p-values - ',pval)

if pval< 0.05: # alpha value is 0.05

  print(" we are rejecting null hypothesis")

else:

  print("we are accepting null hypothesis")