Day: 19.06.2025

**Practice problems**

1. import pandas as pd

from statistics import mode

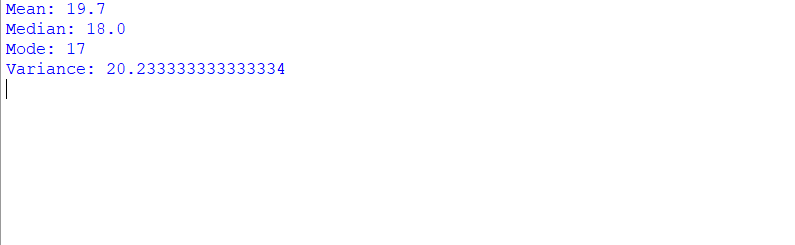
data = [15, 22, 17, 19, 22, 17, 29, 24, 17, 15]

print("Mean:", sum(data)/len(data))

print("Median:", pd.Series(data).median())

print("Mode:", mode(data))

print("Variance:", pd.Series(data).var())

****

2. from scipy.stats import norm

import numpy as np

n1, n2 = 100, 120

x1, x2 = 6.2, 5.8

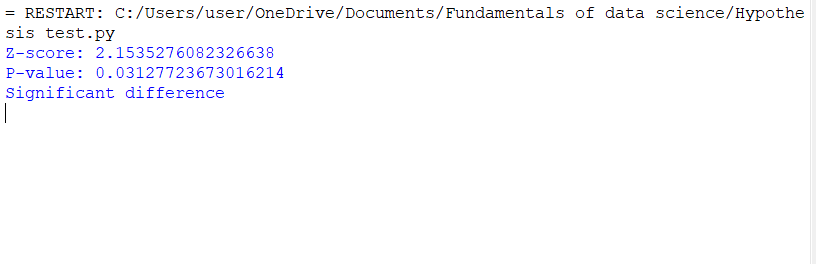
s1, s2 = 1.5, 1.2

z = (x1 - x2) / np.sqrt(s1\*\*2/n1 + s2\*\*2/n2)

p = 2 \* (1 - norm.cdf(abs(z)))

print("Z-score:", z, "\nP-value:", p)

print("Significant difference" if p < 0.05 else "Not significant")

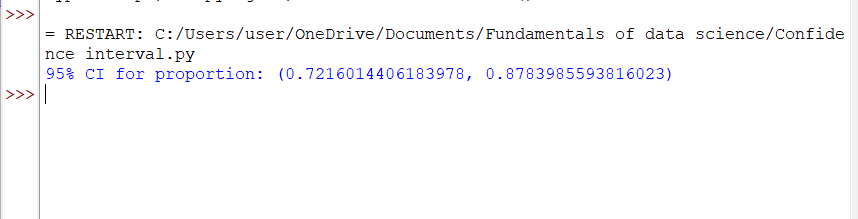


3. from statsmodels.stats.proportion import proportion\_confint

count, nobs = 80, 100

ci\_low, ci\_upp = proportion\_confint(count, nobs, alpha=0.05, method='normal')

print("95% CI for proportion:", (ci\_low, ci\_upp))



4. import numpy as np

import matplotlib.pyplot as plt

x = np.array([8, 3, 2, 10, 11, 3, 6, 5, 6, 8])

y = np.array([4, 12, 1, 12, 9, 4, 9, 6, 1, 14])

m, b = np.polyfit(x, y, 1)

print(f"Line: y = {m:.2f}x + {b:.2f}")

plt.scatter(x, y)

plt.plot(x, m\*x + b, color='red')

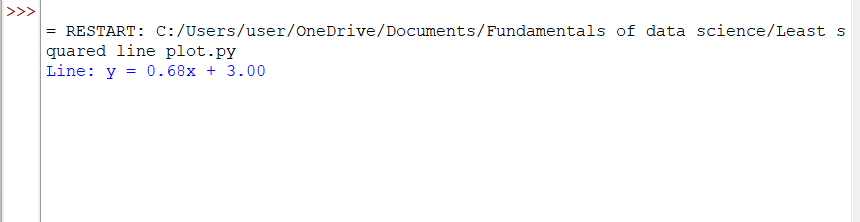
plt.title("Line of Best Fit")

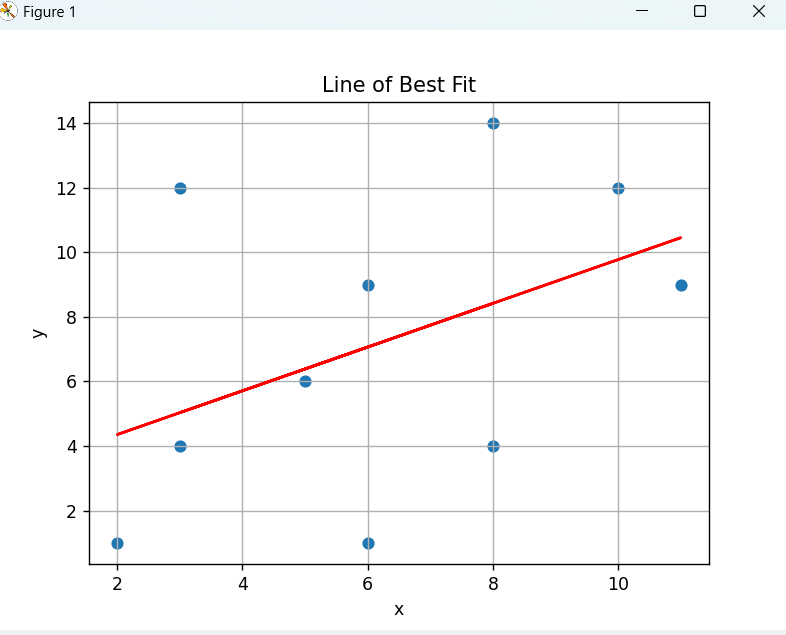
plt.xlabel("x")

plt.ylabel("y")

plt.grid(True)

plt.show()





5. import pandas as pd

x = pd.Series([8, 3, 2, 10, 11, 3, 6, 5, 6, 8])

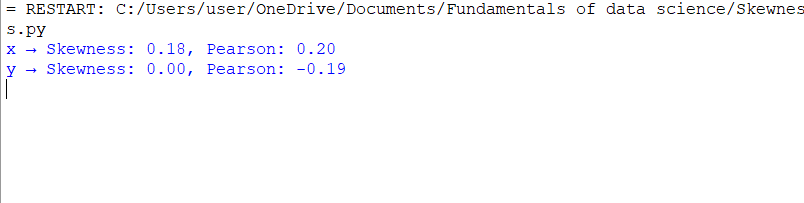
y = pd.Series([4, 12, 1, 12, 9, 4, 9, 6, 1, 14])

for name, data in zip(['x', 'y'], [x, y]):

skewness = data.skew()

pearson = 3 \* (data.mean() - data.median()) / data.std()

print(f"{name} → Skewness: {skewness:.2f}, Pearson: {pearson:.2f}")



6. Data manipulation

import pandas as pd

import matplotlib.pyplot as plt

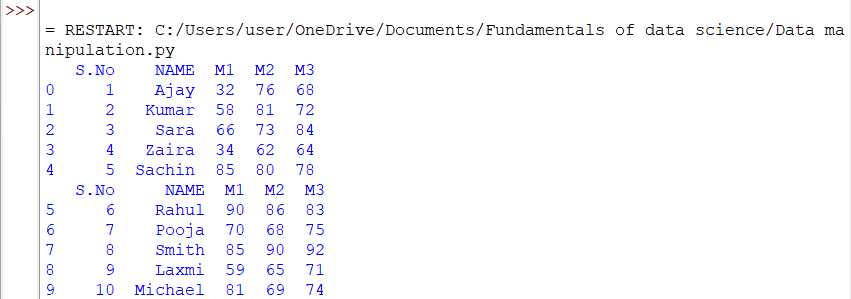
# i) Read the data

marks = pd.read\_csv("marks.csv")

# ii) First and last 5 rows

print(marks.head())

print(marks.tail())



# Purpose of describe():

# Gives summary statistics (mean, std, min, max, etc.)

# iv) Select 3rd to 6th row (index 2 to 5)

print(marks.iloc[2:6])



# v) Select rows where M3 > 84

print(marks[marks['M3'] > 84])



# vi) Filter rows with missing values

print(marks[marks.isnull().any(axis=1)])

# vii) Remove rows with missing values

print(marks.dropna())

# viii) Sort M1 column in descending order

print(marks.sort\_values(by='M1', ascending=False))

# ix) Plot the table (bar chart for marks)

marks.set\_index('NAME')[['M1','M2','M3']].plot(kind='bar')

plt.ylabel("Marks")

plt.title("Student Marks")

plt.tight\_layout()

plt.show()

# x) marks.ix[3:6, ['m2','m3']] → outdated method, use loc or iloc

# Updated version:

print(marks.loc[3:6, ['M2', 'M3']])

