Department of Computer Science and Engineering (Data Science)
Sy.B.Tech. Sem: IV
Subject: Statistics For Data Science (DJS23DLPC403)
Experiment 2

Name: Smayan Kulkarni **SAP ID: 60009230142**

Date:	**Experiment Title: Correlation**
Aim	Given a data set of 10 rows. Calculate Karl Pearson's coefficient of correlation, Spearman's rank correlation coefficient (using repeated ranks) manually. Then write a python program to calculate both coefficients and match it with the manually calculated values. Solve the real world problem statements.
Software	Google Colab, Visual Studio Code, Jupyter Notebook
Theory To Be written	What is a strong monotonic relationship? State the types of monotonic relationship with examples.
Implementation	Data: X = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 Y = 5, 6, 7, 8, 7, 9, 10, 10, 11, 12 Plot a scatter plot of the above data. **Step 2: Karl Pearson's Coefficient of Correlation (r)** The formula for Karl Pearson's correlation coefficient is: $r = \frac{n\sum xy - \sum x \sum y}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}}$ Where: • n is the number of data points (in this case 10), • x and y are the individual data points of the variables X and Y.

Step 3: Spearman's Rank Correlation Coefficient (ρ)

Spearman's rank correlation is based on the ranks of the data. The formula is:

$$\rho = 1 - \frac{6 \sum d^2}{n(n^2 - 1)}$$

Where:

- d is the difference between the ranks of the corresponding values of X and Y,
- n is the number of data points.

Python Code :

1, Write a function to calculate pearson_correlation(X, Y).

2. Write a function to calculate spearman_rank_correlation(X, Y).

3. Use scipy to verify Spearman's rank.

4. Print all the three results.

Real world problems.

$\boldsymbol{Q1}$. *The following table gives the data on weekly family consumption expenditure(Y) and weekly family income(X)*

$Y:$	70	65	90	95	110	115	120	140	155	150
$X:$	80	100	120	140	160	180	200	220	240	260

(i) Compute the coefficient of correlation between X and Y.

(ii) Test the significance of the coefficient of correlation between X and Y at 5 percent level of significance.

$\boldsymbol{Q2}$. *The following table gives the per capita household expenditure on food (Y) and per capita total household expenditure (X)*

$Y:$	60	90	110	125	150	170	180	200	220	230	240	250	255	260	260
$X:$	100	150	200	250	300	350	400	450	500	550	600	650	700	750	800

(i) Compute the coefficient of correlation between X and Y.

(ii) Test the significance of the coefficient of correlation between X and Y at 5 percent level of significance.

	Q3. A company wants to analyze the **factors affecting employee productivity**. The HR department wants to know: 1. Which independent variable (**X1, X2, X3**) has the **strongest correlation** with employee productivity (**Y**)? 2. Is the correlation **statistically significant** at a **5% level**? 3. Can we visualize the relationships using scatter plots and a heatmap?

Employee	Hours Worked (X1)	Experience (X2)	Training Programs (X3)	Productivity Score (Y)
1	35	2	1	50
2	40	3	2	55
3	45	5	3	65
4	50	7	2	70
5	52	9	3	78
6	55	10	4	85
7	60	12	4	88
8	62	14	5	90
9	65	15	6	92
10	68	18	6	94
11	70	20	7	96
12	75	22	8	98

Conclusion	Hence, we have calculated Karl Pearson's coefficient of correlation, Spearman's rank correlation coefficient (using repeated ranks) manually and have implemented a python program to calculate both. Both the coefficients are matching. Conclusion of real world problem 1 Conclusion of real world problem 2 Conclusion of real world problem 3
Colab Link	https://colab.research.google.com/github/SmayanKulkarni/AI-and-ML-Course/blob/master/SDS/exp_2.ipynb

Signature of Faculty

zhnkdstp5

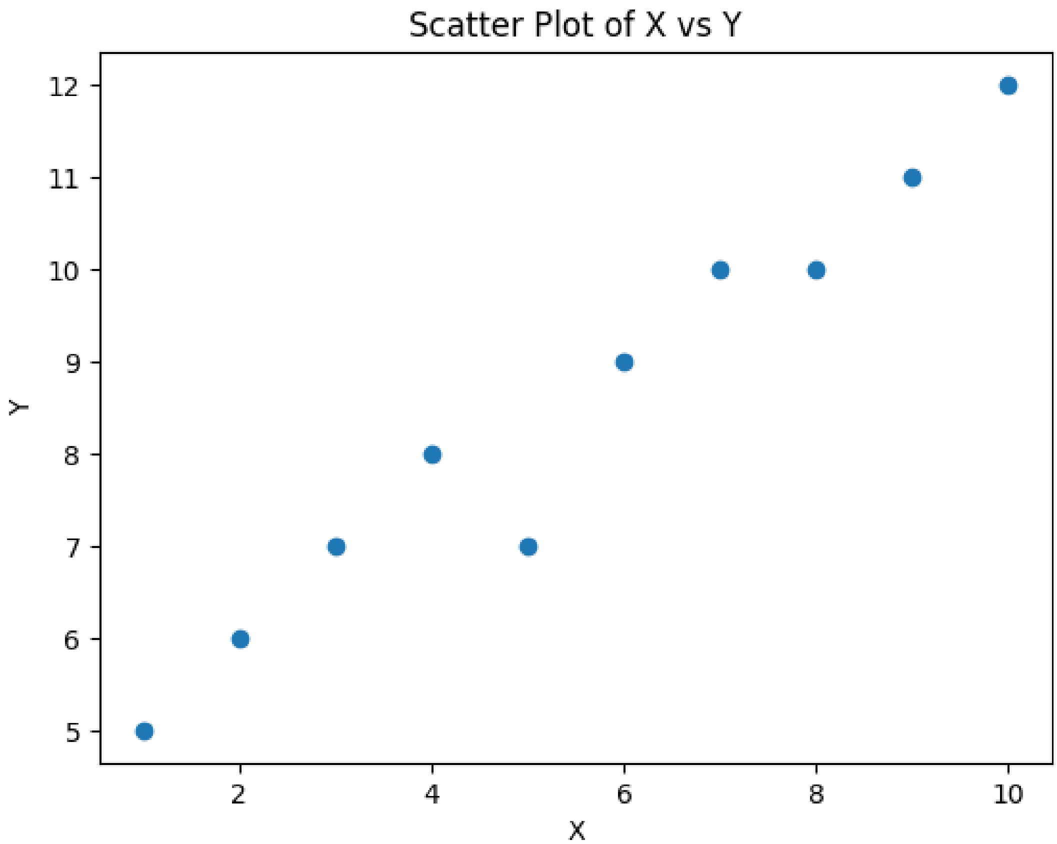
February 23, 2025

[1]:
```python
import numpy as np
import scipy.stats as stats
import matplotlib.pyplot as plt
import seaborn as sns
```

Plotting a scatter plot

[2]:
```python
# Data for the first part of the experiment
X = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Y = [5, 6, 7, 8, 7, 9, 10, 10, 11, 12]
```

[3]:
```python
# Plotting the scatter plot
plt.scatter(X, Y)
plt.title("Scatter Plot of X vs Y")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
```



Function to calculate Pearson's correlation coefficient

[4]:
```
# Function to calculate Pearson's correlation coefficient
def pearson_correlation(x, y):
    n = len(x)
    mean_x, mean_y = np.mean(x), np.mean(y)
    numerator = sum((xi - mean_x) * (yi - mean_y) for xi, yi in zip(x, y))
    denominator = np.sqrt(sum((xi - mean_x) ** 2 for xi in x) * sum((yi - 
 ↪mean_y) ** 2 for yi in y))
    return numerator / denominator
```

Function to calculate Spearman's rank correlation coefficient

[5]:
```
# Function to calculate Spearman's rank correlation coefficient
def spearman_rank_correlation(x, y):
    n = len(x)
    rank_x = np.argsort(np.argsort(x))
    rank_y = np.argsort(np.argsort(y))
    return pearson_correlation(rank_x, rank_y)
```

Calculate correlation using in-built libraries

[6]:
```python
import scipy.stats as stats
```

[7]:
```python
scipy_spearman = stats.spearmanr(X, Y).correlation
```

Printing all the results

[8]:
```python
# Calculate and print correlations
pearson_result = pearson_correlation(X, Y)
spearman_result = spearman_rank_correlation(X, Y)
```

[9]:
```python
print(f"Pearson Correlation: {pearson_result}")
print(f"Spearman Rank Correlation (manual): {spearman_result}")
print(f"Spearman Rank Correlation (scipy): {scipy_spearman}")
```

```
Pearson Correlation: 0.976791617387907
Spearman Rank Correlation (manual): 0.9878787878787879
Spearman Rank Correlation (scipy): 0.9756278933105668
```

1 Real-World Problem-1

[10]:
```python
from scipy.stats import pearsonr, t

# Data from the table
X = [80, 100, 120, 140, 160, 180, 200, 220, 240, 260]
Y = [70, 65, 90, 95, 110, 115, 120, 140, 155, 150]

# Step 1: Calculate Pearson correlation
correlation, p_value = pearsonr(X, Y)
print(f"Pearson Correlation Coefficient: {correlation:.4f}")

# Step 2: Test the significance of the correlation
n = len(X)
t_statistic = correlation * np.sqrt((n - 2) / (1 - correlation ** 2))
alpha = 0.05

# Critical value for t-distribution at (n - 2) degrees of freedom
critical_value = t.ppf(1 - alpha / 2, df=n - 2)

# Output results
print(f"T-Statistic: {t_statistic:.4f}")
print(f"Critical Value (5% significance level): {critical_value:.4f}")

if abs(t_statistic) > critical_value:
    print("The correlation is statistically significant at the 5% level.")
else:
    print("The correlation is not statistically significant at the 5% level.")
```

```
Pearson Correlation Coefficient: 0.9808
T-Statistic: 14.2432
Critical Value (5% significance level): 2.3060
The correlation is statistically significant at the 5% level.
```

2 Real-World Problem-2

[11]:
```python
from scipy.stats import pearsonr, t

# Data from the table
X = [100, 150, 200, 250, 300, 350, 400, 450, 500, 550, 600, 650, 700, 750, 800]
Y = [60, 90, 110, 125, 150, 170, 180, 200, 220, 230, 240, 250, 255, 260, 260]

# Step 1: Calculate Pearson correlation
correlation, p_value = pearsonr(X, Y)
print(f"Pearson Correlation Coefficient: {correlation:.4f}")

# Step 2: Test the significance of the correlation
n = len(X)
t_statistic = correlation * np.sqrt((n - 2) / (1 - correlation ** 2))
alpha = 0.05

# Critical value for t-distribution at (n - 2) degrees of freedom
critical_value = t.ppf(1 - alpha / 2, df=n - 2)

# Output results
print(f"T-Statistic: {t_statistic:.4f}")
print(f"Critical Value (5% significance level): {critical_value:.4f}")

if abs(t_statistic) > critical_value:
    print("The correlation is statistically significant at the 5% level.")
else:
    print("The correlation is not statistically significant at the 5% level.")
```

```
Pearson Correlation Coefficient: 0.9766
T-Statistic: 16.3686
Critical Value (5% significance level): 2.1604
The correlation is statistically significant at the 5% level.
```

3 Real-World Problem-3

[12]:
```python
import pandas as pd
```

[13]:
```python
data = {
    "X1_Hours_Worked": [35, 40, 45, 50, 52, 55, 60, 62, 65, 68, 70, 75],
    "X2_Experience": [2, 3, 5, 7, 9, 10, 12, 14, 15, 18, 20, 22],
```

```
    "X3_Training_Programs": [1, 2, 3, 2, 3, 4, 4, 5, 5, 6, 7, 8],
    "Y_Productivity_Score": [50, 55, 65, 70, 78, 85, 88, 90, 92, 94, 96, 98]
}

df = pd.DataFrame(data)
```

Q1

[14]:
```
correlations = df.corr()["Y_Productivity_Score"].drop("Y_Productivity_Score")
print(correlations)
strongest_correlation = correlations.idxmax()
print(f"Strongest correlation is with: {strongest_correlation}")
```

```
X1_Hours_Worked         0.975667
X2_Experience           0.946043
X3_Training_Programs    0.904125
Name: Y_Productivity_Score, dtype: float64
Strongest correlation is with: X1_Hours_Worked
```

Q2

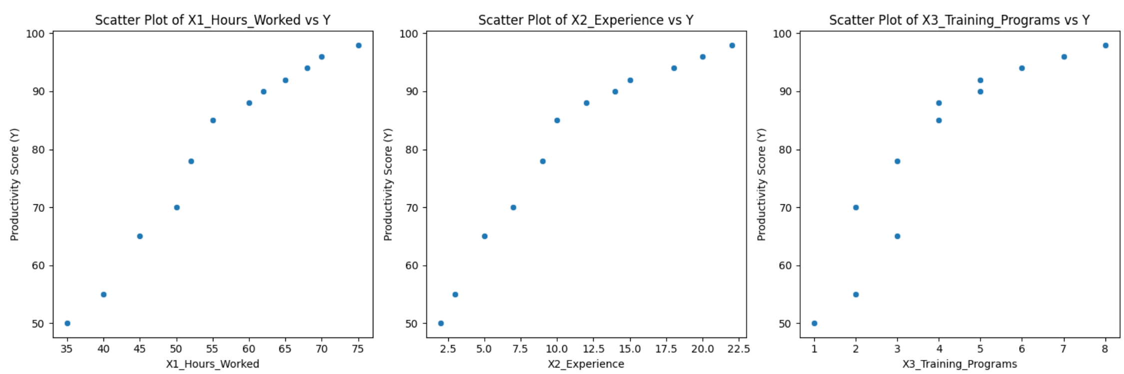
[15]:
```
alpha = 0.05
n = len(df)

for col in ["X1_Hours_Worked", "X2_Experience", "X3_Training_Programs"]:
    r, _ = stats.pearsonr(df[col], df["Y_Productivity_Score"])
    t_statistic = r * np.sqrt((n - 2) / (1 - r**2))
    p_value = 2 * (1 - stats.t.cdf(abs(t_statistic), df=n-2))

    print(f"Variable: {col}")
    print(f"Pearson Correlation Coefficient: {r:.4f}")
    print(f"T-statistic: {t_statistic:.4f}")
    print(f"P-value: {p_value:.4f}")
    print("Significant at 5% level" if p_value < alpha else "Not significant at␣
 ↪5% level")
    print("-" * 50)
```

```
Variable: X1_Hours_Worked
Pearson Correlation Coefficient: 0.9757
T-statistic: 14.0717
P-value: 0.0000
Significant at 5% level
--------------------------------------------------
Variable: X2_Experience
Pearson Correlation Coefficient: 0.9460
T-statistic: 9.2323
P-value: 0.0000
Significant at 5% level
```

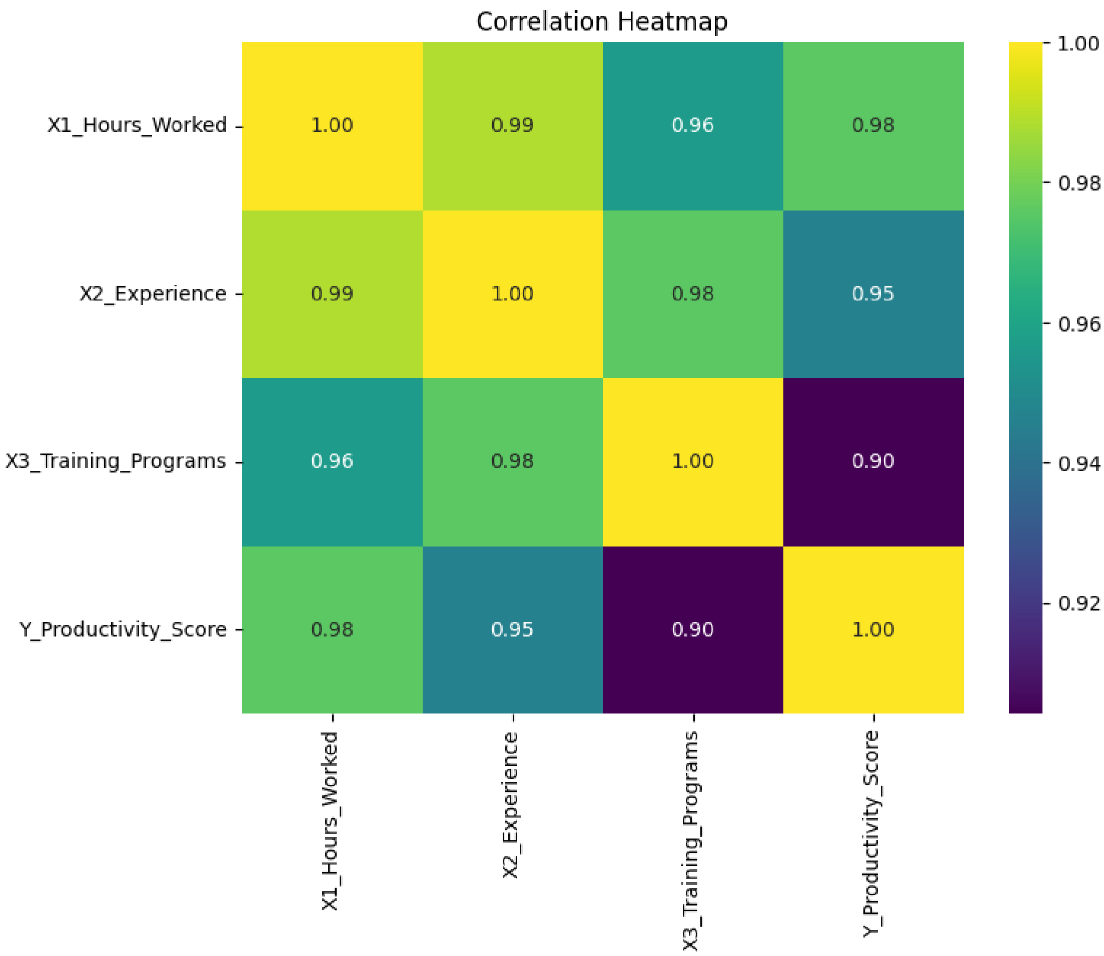
```
--------------------------------------------------
Variable: X3_Training_Programs
Pearson Correlation Coefficient: 0.9041
T-statistic: 6.6916
P-value: 0.0001
Significant at 5% level
--------------------------------------------------
```

Q3

[16]:
```python
plt.figure(figsize=(15,5))
for i, col in enumerate(["X1_Hours_Worked", "X2_Experience",␣
↪"X3_Training_Programs"], 1):
    plt.subplot(1,3,i)
    sns.scatterplot(x=df[col], y=df["Y_Productivity_Score"])
    plt.xlabel(col)
    plt.ylabel("Productivity Score (Y)")
    plt.title(f"Scatter Plot of {col} vs Y")
plt.tight_layout()
plt.show()
```



[17]:
```python
plt.figure(figsize=(8,6))
sns.heatmap(df.corr(), annot=True, cmap="viridis", fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```


Correlation Heatmap
X1_Hours_Worked
X2_Experience
X3_Training_Programs
Y_Productivity_Score
1.00
0.99
0.96
0.98
0.99
1.00
0.98
0.95
0.96
0.98
1.00
0.90
0.98
0.95
0.90
1.00
X1_Hours_Worked
X2_Experience
X3_Training_Programs
Y_Productivity_Score
1.00
0.98
0.96
0.94
0.92