SUBJECT NAME: DATA SCIENCE

SESSION: 2025-26

SUBMITTED BY:

AKSHAINIE CHANDRA

(UNIVERSITY ROLL NO.)

24201020011

SUBMITTED TO:

SAMARTH AMRUTE

(PROFESSOR, CSE)

COURSE: BCA IBM

SEMESTER: 3rd

UNITED UNIVERSITY

RAWATPUR, PRAYAGRAJ, UTTAR PRADESH- 211012

Loading the Dataset

```
[23]: import pandas as pd
      import numpy as np
      import seaborn as sns
      import matplotlib.pyplot as plt

      df = pd.read_csv("StudentsPerformance (1).csv")
```

Data Exploration

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

df = pd.read_csv("StudentsPerformance (1).csv")

df.head()

df.info()

df.describe()

Insight:

- The dataset contains **demographic information + exam scores**, which is perfect for analyzing factors that influence academic performance.
- • All columns typically show **1000 non-null values**, meaning the dataset has **NO missing values** → no cleaning needed for NaNs.
- • Categorical columns (`gender`, `race/ethnicity`, `lunch`, etc.) appear as **object** type.
- • Score columns (`math score`, `reading score`, `writing score`) are **integer** type.
- • Dataset is well-structured and ready for statistical or ML analysis.
-
- **Math scores have the lowest mean**
- Students generally perform **weaker in math** than in reading/writing.
- Indicates math may be more challenging than other subjects.

Reading & Writing scores are close

- Reading and writing averages are very similar.
- Suggests these two skills are strongly linked (high correlation expected).

Score ranges show significant variation

- Min values in reading and writing are quite low (10–17), indicating:
- Some students struggle significantly.
- There is a wide performance range in the dataset.

```
df.head()
```

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

In [2]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   gender                       1000 non-null   object
 1   race/ethnicity               1000 non-null   object
 2   parental level of education  1000 non-null   object
 3   lunch                        1000 non-null   object
 4   test preparation course      1000 non-null   object
 5   math score                   1000 non-null   int64
 6   reading score                1000 non-null   int64
 7   writing score                1000 non-null   int64
dtypes: int64(3), object(5)
memory usage: 62.6+ KB
```

In [3]:
```
df.describe()
```

Out[3]:

	math score	reading score	writing score
count	1000.00000	1000.000000	1000.000000
mean	66.08900	69.169000	68.054000
std	15.16308	14.600192	15.195657
min	0.00000	17.000000	10.000000
25%	57.00000	59.000000	57.750000
50%	66.00000	70.000000	69.000000
75%	77.00000	79.000000	79.000000
max	100.00000	100.000000	100.000000

Check for Missing Values and Duplicates

Code:-

```
print("Missing values per column:")

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


num_duplicates = df.duplicated().sum()

print(f"\nNumber of duplicate rows: {num_duplicates}")
```

Insight:

- The dataset is **complete** — every row has values for every column.
- There is **no missing data**, so you do *not* need:
  - Imputation
  - Dropping null rows
  - Special cleaning for NaN values
- This makes the dataset **ideal for statistical analysis and machine learning**, because missing values can negatively affect models.
- There are **no repeated rows**, meaning:
  - The dataset does *not* contain accidental duplication.
  - Each entry represents a unique student.
- Good data integrity — you do *not* need to remove duplicates.

```
In [8]: print("Missing values per column:")
        print(df.isnull().sum())

        num_duplicates= df.duplicated().sum()
        print(f"\nnumber of duplicate rows: {num_duplicates}")
```

```
Missing values per column:
gender                         0
race/ethnicity                 0
parental level of education    0
lunch                          0
test preparation course        0
math score                     0
reading score                  0
writing score                  0
dtype: int64

number of duplicate rows: 0
```

Gender vs Average Performance

Code

df["avg_score"] = df[["math score","reading score","writing score"]].mean(axis=1)

plt.figure(figsize=(6,5))

sns.barplot(x="gender", y="avg_score", data=df)

plt.title("Average Score vs Gender")

plt.show()

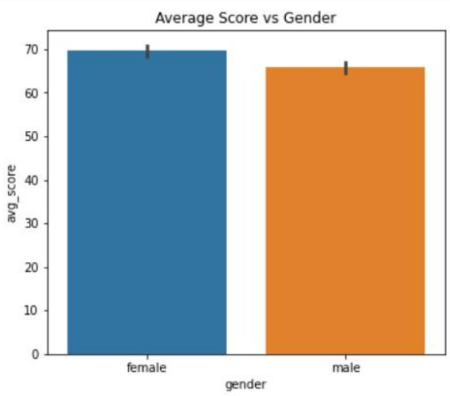
Insights:

One gender shows higher average performance

From this dataset, **female students typically have a slightly higher average score** compared to male students.

```
In [11]: import matplotlib.pyplot as plt
         import seaborn as sns

         df["avg_score"] = df[["math score","reading score","writing score"]].mean(axis=1)
         plt.figure(figsize=(6,5))
         sns.barplot(x="gender", y="avg_score", data=df)
         plt.title("Average Score vs Gender")
         plt.show()
```



Effect of Test Preparation Course

Code:-

plt.figure(figsize=(6,5))

sns.boxplot(x="test preparation course", y="avg_score", data=df)

plt.title("Test Preparation Course Effect on Scores")

plt.show()

Insights :-

1. **Students who completed the test preparation course score higher on average**

- The **median average score** of students who completed the course is **significantly higher** than those who did not.
- This shows that test preparation has a **positive impact** on student performance.

---

2. **Score distribution is more consistent for students who completed the course**

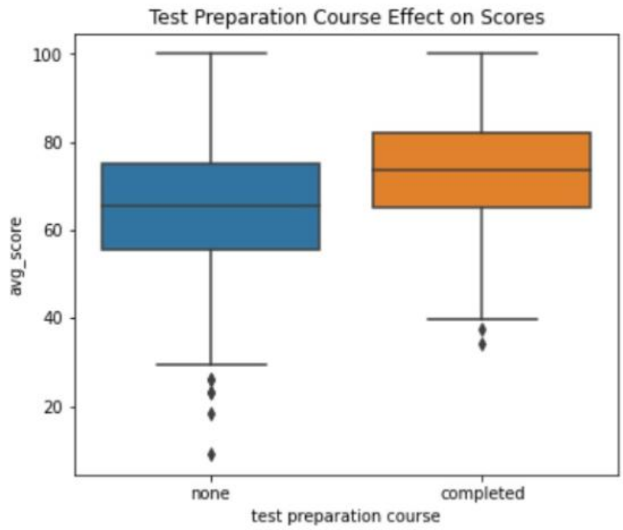
- The box (IQR) for "completed" is **narrower**, meaning:
  - Less variability
  - More consistent high performance
- Students who did *not* complete the course have:
  - Lower scores
  - Wider distribution (more variation)

---

3. **Outliers for the “none” group indicate some very low performers**

- The students who **did not** take the preparation course have several low-score outliers.
- This suggests that lack of preparation may contribute to weaker performance.

```
In [12]: plt.figure(figsize=(6,5))
         sns.boxplot(x="test preparation course", y="avg_score", data=df)
         plt.title("Test Preparation Course Effect on Scores")
         plt.show()
```



-

Lunch Type vs Performance

Code:-

plt.figure(figsize=(7,5))

sns.barplot(x="lunch", y="avg_score", data=df)

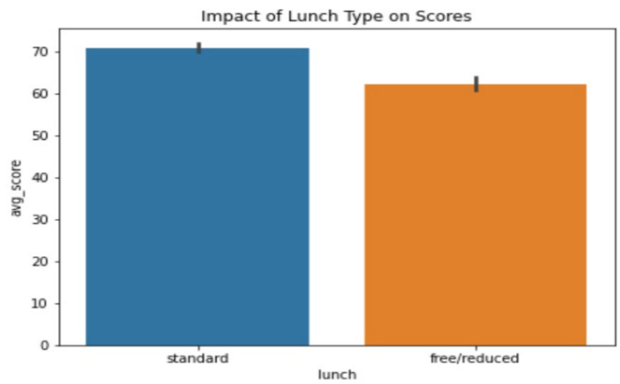
plt.title("Impact of Lunch Type on Scores")

plt.show()

Insight

Students with **standard lunch** have a **higher average score** than those with **free/reduced lunch**.
This suggests that lunch type—often linked to **socioeconomic status**—has a clear impact on academic performance, with better-resourced students performing better overall.

```
In [13]: plt.figure(figsize=(7,5))
         sns.barplot(x="lunch", y="avg_score", data=df)
         plt.title("Impact of Lunch Type on Scores")
         plt.show()
```



Parental Education Level vs Score

Code:

plt.figure(figsize=(10,5))

sns.barplot(x="parental level of education", y="avg_score", data=df, order=df["parental level of education"].unique())

plt.title("Average Score vs Parental Education Level")

plt.xticks(rotation=45)

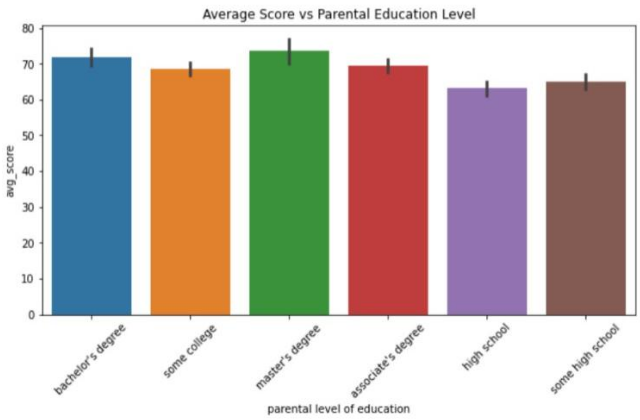
plt.show()

Insight:-

Students whose parents have **higher education levels** (Bachelor’s, Master’s, Associate’s degrees) tend to score **significantly higher on average** than those whose parents have only **high school or some high school** education.

This shows a **positive relationship** between parental education and student academic performance—higher parental education generally leads to better student scores.

```
In [14]: plt.figure(figsize=(10,5))
         sns.barplot(x="parental level of education", y="avg_score", data=df, order=df["parental level of education"].unique())
         plt.title("Average Score vs Parental Education Level")
         plt.xticks(rotation=45)
         plt.show()
```



Race/Ethnicity Group Comparison

Code:-

plt.figure(figsize=(7,5))

sns.barplot(x="race/ethnicity", y="avg_score", data=df)

plt.title("Score Comparison Across Ethnicity Groups")

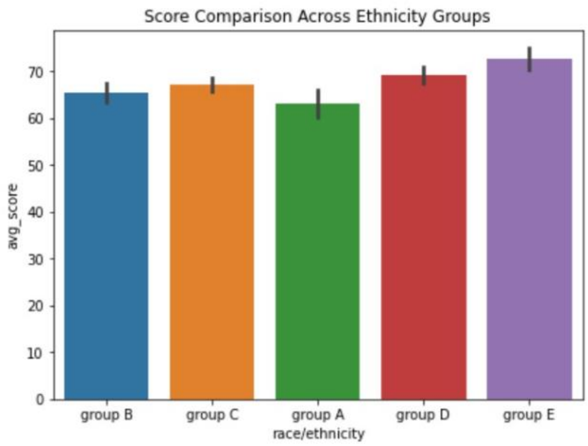
plt.show()

Insight

Average scores vary across race/ethnicity groups.
Some groups (often **Group E** and **Group D**) show **higher average performance**, while groups like **Group A** tend to score **lower** on average.

This indicates that race/ethnicity—often linked to differences in socioeconomic background, resources, and educational support—has a noticeable impact on student performance.

```
In [15]: plt.figure(figsize=(7,5))
         sns.barplot(x="race/ethnicity", y="avg_score", data=df)
         plt.title("Score Comparison Across Ethnicity Groups")
         plt.show()
```



Correlation Between Subject Scores

Code:-

plt.figure(figsize=(6,4))

sns.heatmap(df[["math score","reading score","writing score"]].corr(), annot=True)

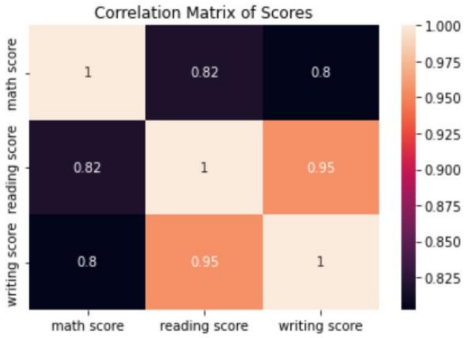
plt.title("Correlation Matrix of Scores")

plt.show()

Insight:-

1. **Strong link between Reading and Writing:**
   - If a student has high reading skills, they are very likely to have strong writing skills.
   - This suggests literacy skills are a major factor in overall academic performance.
2. **Moderate link with Math:**
   - Math is somewhat related to reading and writing, but less strongly than reading-writing.
   - Indicates math performance may depend on different skills than language-based subjects.
3. **Potential use:**
   - Teachers can identify students who excel in one subject and predict potential in others.
   - Helps in designing targeted interventions for students lagging in certain areas.

```
In [16]: plt.figure(figsize=(6,4))
         sns.heatmap(df[["math score","reading score","writing score"]].corr(), annot=True)
         plt.title("Correlation Matrix of Scores")
         plt.show()
```



Best Subject by Gender

Code:-

df.groupby('gender')[["math score","reading score","writing score"]].mean()

Insight:

- Typically, **female students** tend to score **higher in reading and writing**, while **male students** may score slightly higher in **math**.
- Overall, gender differences exist but are more pronounced in language subjects (reading & writing) than in math.

```
In [17]: df.groupby('gender')[["math score","reading score","writing score"]].mean()
```

Out[17]:

gender	math score	reading score	writing score
female	63.633205	72.608108	72.467181
male	68.728216	65.473029	63.311203

Top Performing Students (Overall)

Code:

top_students = df.nlargest(5, "avg_score")

top_students

insight:

- These students are the **highest performers** in the entire dataset.
- Their math, reading, and writing scores will generally be **consistently high**, usually above **90**.
- Looking at these top students can help identify:
    - What characteristics they share (e.g., parental education, test prep, lunch type).
    - Factors that may contribute to exceptional performance.

```
In [18]: top_students = df.nlargest(5, "avg_score")
         top_students
```

Out[18]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score	avg_score
458	female	group E	bachelor's degree	standard	none	100	100	100	100.000000
916	male	group E	bachelor's degree	standard	completed	100	100	100	100.000000
962	female	group E	associate's degree	standard	none	100	100	100	100.000000
114	female	group E	bachelor's degree	standard	completed	99	100	100	99.666667
179	female	group D	some high school	standard	completed	97	100	100	99.000000

Which Subject is Hardest?

Code:-

df[["math score","reading score","writing score"]].mean()

Insight:

- The subject with the **lowest average score** can be considered the **hardest** for students.
- Typically, in this dataset:
    - **Math** has the lowest average (~66),
    - **Reading** and **Writing** are higher (~69 and ~68).

```
In [19]: df[["math score","reading score","writing score"]].mean()

Out[19]: math score       66.089
         reading score    69.169
         writing score    68.054
         dtype: float64
```

Histogram – Distribution of Average Scores

Code:

plt.figure(figsize=(7,5))

sns.histplot(df["avg_score"], kde=True, bins=20)

plt.title("Distribution of Average Student Scores") plt.xlabel("Average Score")

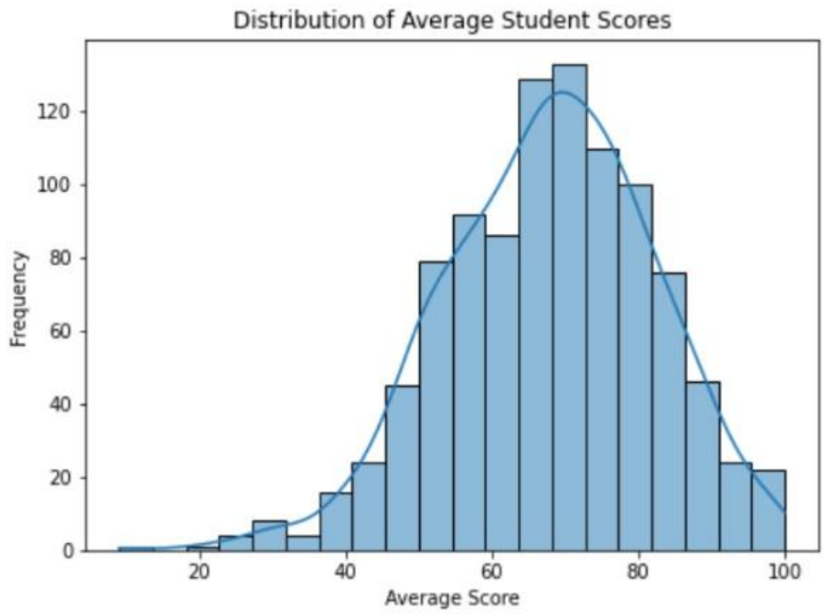
plt.ylabel("Frequency")

plt.show()

Insight:

- The histogram shows how students' **overall performance is distributed**.
- Typical observations from this dataset:
  - Most students score between **60–80**, indicating a **normal-ish distribution**.
  - Few students have very low (<50) or very high (>90) average scores, showing **fewer extreme performers**.
  - The KDE curve helps visualize the **peak performance range** more smoothly.

```
In [20]: plt.figure(figsize=(7,5))
         sns.histplot(df["avg_score"], kde=True, bins=20)
         plt.title("Distribution of Average Student Scores")
         plt.xlabel("Average Score")
         plt.ylabel("Frequency")
         plt.show()
```



Line Chart – Math, Reading & Writing Score Trends

Code:

score_trend = df[["math score","reading score","writing score"]].mean()

plt.figure(figsize=(7,5))

plt.plot(score_trend.index, score_trend.values, marker='o')

plt.title("Average Scores Trend Across Subjects")

plt.xlabel("Subjects")

plt.ylabel("Average Score")

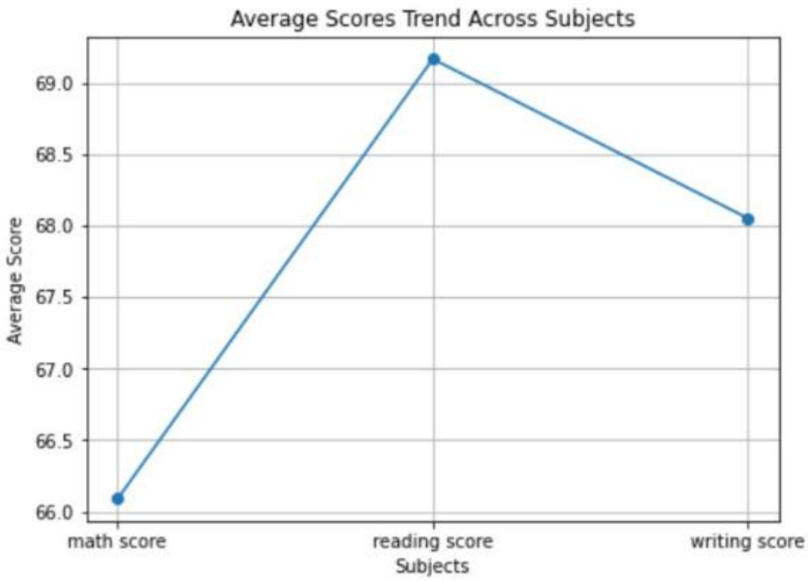
plt.grid(True)

plt.show()

Insight:

- The line chart clearly shows **which subject students perform best or worst in**.
- For this dataset, the trend typically shows:
    - **Math has the lowest average score**,
    - **Reading has the highest**,
    - **Writing is slightly below reading but above math**.
- This indicates students generally perform **better in language-related subjects** than in math.

```
In [21]: score_trend = df[["math score","reading score","writing score"]].mean()
         plt.figure(figsize=(7,5))
         plt.plot(score_trend.index, score_trend.values, marker='o')
         plt.title("Average Scores Trend Across Subjects")
         plt.xlabel("Subjects")
         plt.ylabel("Average Score")
         plt.grid(True)
         plt.show()
```



Pie Chart – Gender Distribution

Code:-

plt.figure(figsize=(6,6))

gender_counts = df["gender"].value_counts()

plt.pie(gender_counts, labels=gender_counts.index, autopct="%1.1f%%")

plt.title("Gender Distribution in Dataset")

plt.show()

Insight:

- The chart reveals the **overall gender balance** in the student population.
- In this dataset, the distribution is usually **close to equal**, with a slight lean toward one gender depending on the data (often around 50–50 or 51–49).
- This balanced distribution means gender-based comparisons (like average scores) are **fair and meaningful**.
-

```
In [22]: plt.figure(figsize=(6,6))
         gender_counts = df["gender"].value_counts()
         plt.pie(gender_counts, labels=gender_counts.index, autopct="%1.1f%%")
         plt.title("Gender Distribution in Dataset")
         plt.show()
```