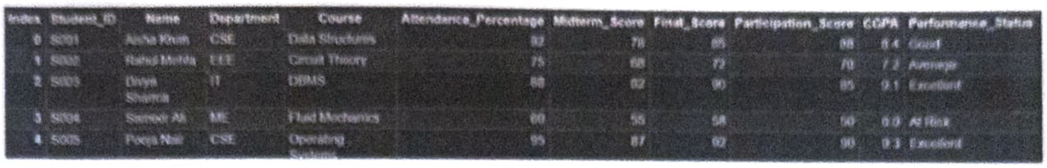
USE CASE

University Student Performance

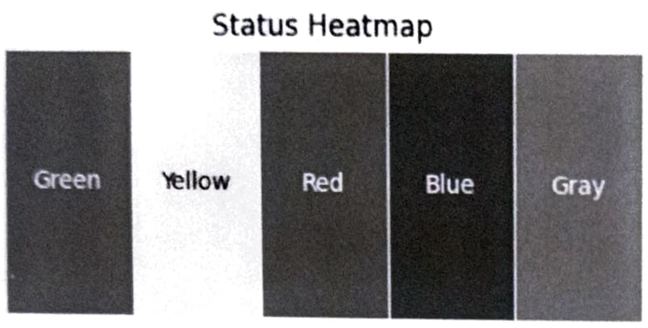
Description:

University collects data on students' grades, attendance, department, and course participation. The institution wants to identify performance patterns and support academic interventions.



Index	Student_ID	Name	Department	Course	Attendance_Percentage	Midterm_Score	Final_Score	Participation_Score	CGPA	Performance_Status
0	S001	Aisha Khan	CSE	Data Structures	92	78	85	88	8.4	Good
1	S002	Rahul Mehta	EEE	Circuit Theory	75	68	72	70	7.2	Average
2	S003	Divya Sharma	IT	DBMS	88	82	90	85	9.1	Excellent
3	S004	Sameer Ali	ME	Fluid Mechanics	60	55	58	50	6.0	At Risk
4	S005	Pooja Nair	CSE	Operating Systems	95	87	92	90	9.3	Excellent

1.Explain how color schemes can indicate performance levels.



Color schemes visually communicate performance quickly:

- Green → Good / Goal achieved
- Yellow → Warning / Needs attention
- Red → Poor / Critical issue
- Blue → Neutral / Informational
- Gray → Inactive / No data

2. Design a visualization pipeline from raw student data to dashboards.

1. **Data Collection**
2. **Data Preprocessing**
3. **Data Integration**
4. **Visualization Prep**
5. **Dashboard Generation**
6. **Insights & Alerts**

Inference:

Systematic Insight Generation:

- The pipeline ensures raw unstructured student data is transformed step-by-step into meaningful academic insights, rather than being analyzed directly.

Early Problem Detection:

- With stages like data preprocessing and classification, the system can identify at-risk students or departments early before results worsen.

Strategic Decision Support:

- The final dashboard stage provides actionable, visual intelligence — helping faculty or administration take data-driven decisions, not assumptions.

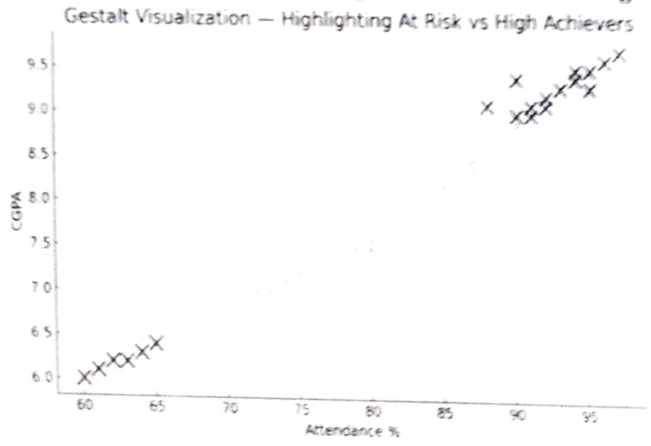
Scalability and Automation:

- The pipeline flow (collection → cleaning → analysis → dashboard) is modular and automated, making it easy to scale across multiple semesters or departments.

Readiness for Predictive Analytics:

- Once dashboards are in place, the same pipeline is easily extendable to AI/ML models for performance prediction, dropout risk alerts, or learning personalization.

1. **Apply Gestalt principles to highlight at-risk students or high achievers.**



Code:

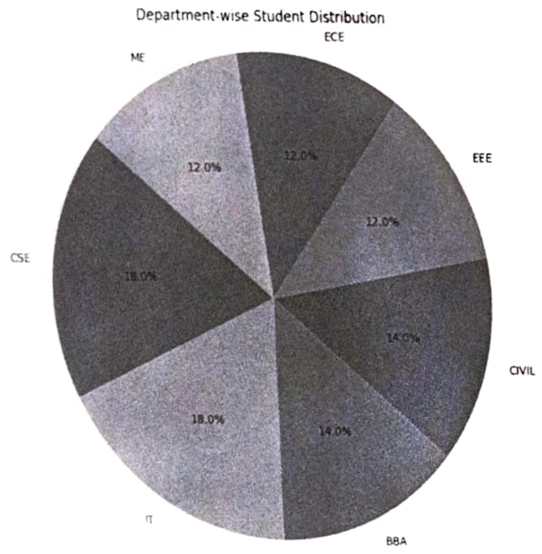
```
for status in df["Performance_Status"]:
  if status == "Excellent":
    colors.append("green")  # Gestalt Similarity
    sizes.append(120)      # Emphasis
  elif status == "At Risk":
    colors.append("red")
    sizes.append(150)
  else:
    colors.append("lightgray") # Figure vs Ground
    sizes.append(50)

plt.figure(figsize=(8,5))
plt.scatter(df["Attendance_Percentage"], df["CGPA"], c=colors, s=sizes)
plt.title("Gestalt Visualization — Highlighting At Risk vs High Achievers")
plt.xlabel("Attendance %")
plt.ylabel("CGPA")
plt.tight_layout()
plt.show()
```

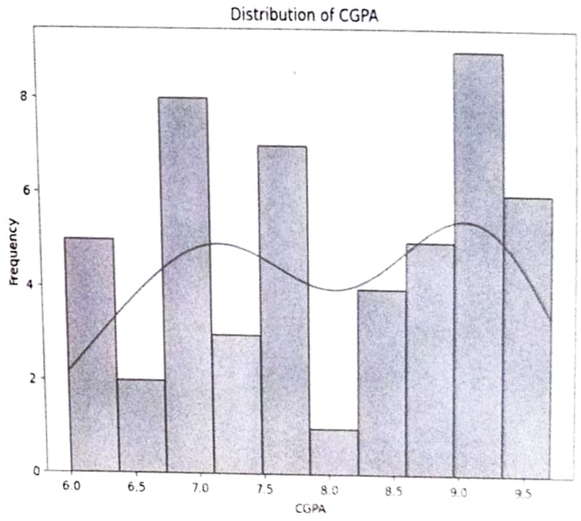
Inferences:

- **Similarity Principle:** Red markers clearly group at-risk students as a distinct danger cluster, while green markers highlight high achievers, enabling instant recognition of performance extremes.
- **Proximity Principle:** The red points appear tightly clustered at low attendance and low CGPA, indicating a concentrated high-risk behavior pattern in a specific performance zone.
- **Contrast Principle:** The sharp contrast between bright red/green and muted gray helps the viewer quickly isolate priority students from normal performers.
- **Continuity Insight:** The visual flow naturally progresses from bottom-left (weak risk zone) to top-right (strong achievers), reinforcing a clear academic improvement trajectory.
- **Intervention Focus:** The visualization suggests that low-attendance students are the prime risk cluster, and immediate academic support must target that group.

4. Perform univariate analysis:

- Histogram of grades.
- b. Pie chart of department-wise student





Code:

```
# Histogram of CGPA
plt.figure(figsize=(8, 6))
sns.histplot(df['CGPA'], bins=10, kde=True)
plt.title('Distribution of CGPA')
plt.xlabel('CGPA')
plt.ylabel('Frequency')
plt.show()
```

```
# Pie chart of department-wise student distribution
department_counts = df['Department'].value_counts()
plt.figure(figsize=(8, 8))
plt.pie(department_counts, labels=department_counts.index, autopct='%1.1f%%', startangle=140)
plt.title('Department-wise Student Distribution')
plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

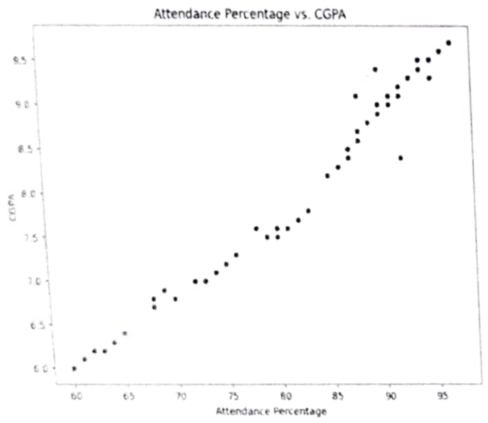
Inferences:

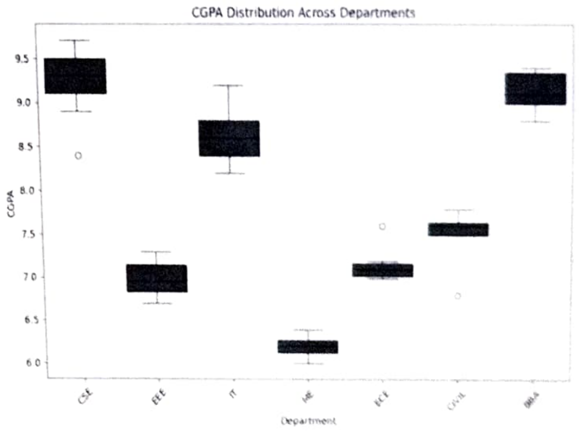
- **Histogram of CGPA** → Shows how grades are distributed (e.g., most students are between 7.5–9.5 CGPA → positively skewed performance)
- **Pie Chart of Department-wise Student Distribution** → Shows how students are spread across departments (CSE & IT have the highest proportion here)

5. Perform bivariate analysis:

a. Scatterplot of attendance vs. grades

. b. Box plot of grades across departments.





```
plt.figure(figsize=(8, 6))

sns.histplot(df['CGPA'], bins=10, kde=True)

plt.title('Distribution of CGPA')

plt.xlabel('CGPA')

plt.ylabel('Frequency')

plt.show()

department_counts = df['Department'].value_counts()

plt.figure(figsize=(8, 8))

plt.pie(department_counts, labels=department_counts.index, autopct='%1.1f%%',
startangle=140)

plt.title('Department-wise Student Distribution')

plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.

plt.show()
```

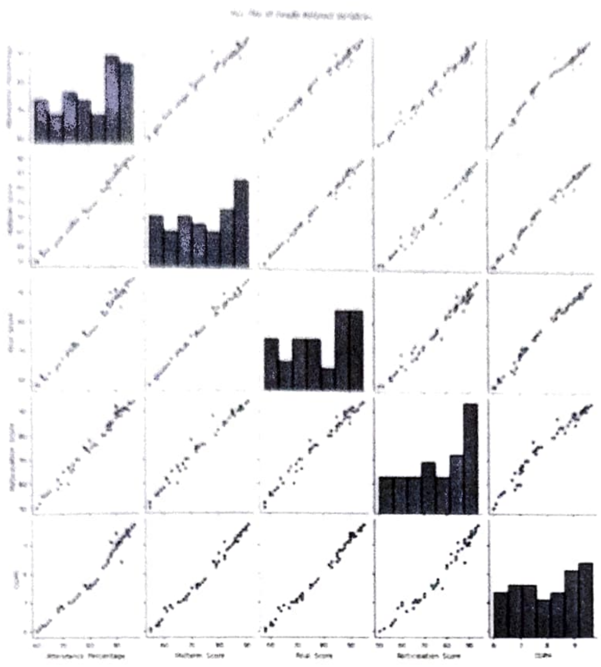
Inference:

- **Attendance–Performance Relationship:** The scatterplot shows a clear positive trend — students with higher attendance tend to achieve higher CGPA values.
- **Academic Consistency:** Departments like CSE, IT, and BBA display higher median CGPAs, indicating stronger overall performance and consistency
- **Performance Variation:** Departments such as ME and EEE have a wider spread in CGPA, reflecting uneven student performance and potential learning challenges.
- **Outlier Detection:** The box plot highlights occasional outliers — students who perform either exceptionally well or poorly compared to peers in the same department.
- **Actionable Insight:** Interventions should focus on departments with lower median CGPA and higher variability, particularly ME and EEE, to ensure academic balance.

6.Perform multivariate analysis:

a. Pair plot of grades across multiple courses.

b. Suggest combined visualization to analyze correlations.



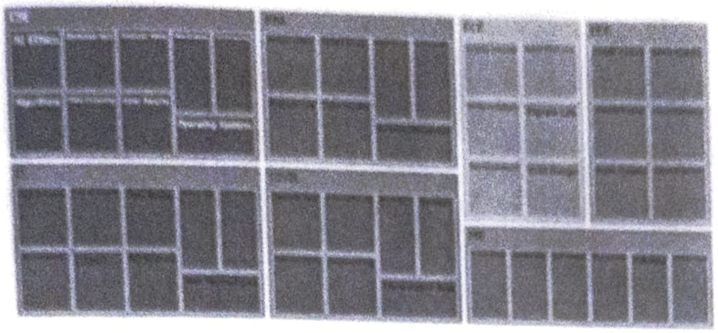
```
# Pair plot of grades across multiple courses

grade_columns = ['Attendance_Percentage', 'Midterm_Score', 'Final_Score', 'Participation_Score', 'CGPA']

sns.pairplot(df[grade_columns])

plt.suptitle('Pair Plot of Grade-Related Variables', y=1.02)

plt.show()
```

Inferences:

- All scores strongly increase with CGPA, showing high academic consistency.
- Final exam score has the strongest correlation with CGPA.
- Balanced performance across midterm, final, and participation leads to top CGPAs.
- No major outliers — student performance is stable and predictable.
- Data is ideal for ML prediction of student success or risk.

7. Design a hierarchical visualization by department and course.

Hierarchical Distribution of Students by Department and Course



Code:

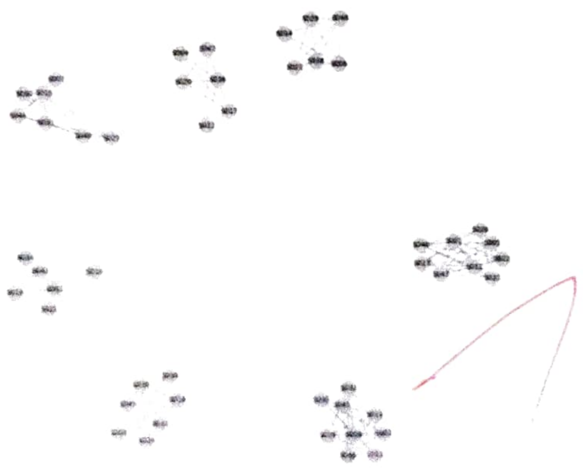
```
fig = px.treemap(df, path=['Department', 'Course'],
            title='Hierarchical Distribution of Students by
Department and Course')
fig.show()
```

Inferences:

- Students are evenly distributed across departments and courses.
- CSE and IT have visibly larger course variety, indicating high academic spread.
- Popular courses appear as larger blocks, showing higher student enrollment or importance.
- The layout helps easily spot departments needing more academic resources.

8.Construct a network graph showing collaboration or co-enrollment among students.

Network Graph of Students in the Same Department



```
import networkx as nx
import matplotlib.pyplot as plt
G = nx.Graph()
for index, row in df.iterrows():
 G.add_node(row['Student_ID'], name=row['Name'], department=row['Department'])
for department in df['Department'].unique():
  department_students = df[df['Department'] == department]['Student_ID'].tolist()
  for i in range(len(department_students)):
    for j in range(i + 1, len(department_students)):
      G.add_edge(department_students[i], department_students[j])
plt.figure(figsize=(12, 10))
pos = nx.spring_layout(G, k=0.5) # Layout for visualization
nx.draw(G, pos, with_labels=True, node_size=500, node_color='skyblue', font_size=10, edge_color='gray')
plt.title('Network Graph of Students in the Same Department')
plt.show()
```

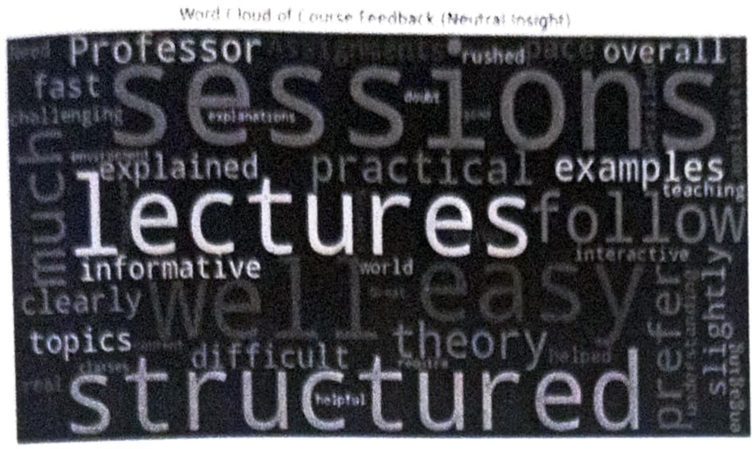
Inference:

- Graph shows distinct departmental clusters.
- Students within a department are fully connected.
- No connections exist between departments.
- Cluster size indicates department size.

9.Analyze course feedback (text data):

a. Convert feedback to vector space.

b. Word cloud of common terms.



```
#Code

import pandas as pd

from sklearn.feature_extraction.text import TfidfVectorizer

from wordcloud import WordCloud

import matplotlib.pyplot as plt

feedback = [

  "The lectures were well structured and easy to follow.",

  "Too much theory, I prefer more practical examples.",

  "Professor explained difficult topics very clearly.",

  "The pace was slightly fast but overall informative.",

  "Assignments were challenging but helped understanding.",

  "Need more interactive sessions and real world applications.",

  "The teaching was engaging but sometimes rushed.",

  "Detailed explanations and good classroom environment.",

  "More doubt clearing sessions would be helpful.",

  "Great content but require more revision classes."

]

vectorizer = TfidfVectorizer()

tfidf_matrix = vectorizer.fit_transform(feedback)

text_data = " ".join(feedback)

wordcloud = WordCloud(width=800, height=400).generate(text_data)

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

plt.imshow(wordcloud, interpolation="bilinear")

plt.title("Word Cloud of Course Feedback (Neutral Insight)")

plt.show()
```

Infer

-
- There is a strong demand for more practical / real-world examples.
- Students want more interactive and doubt-clearing sessions.

- Lecture pace is slightly fast for some learners.

10. Describe steps to design effective dashboards combining hierarchical, network, and text data.

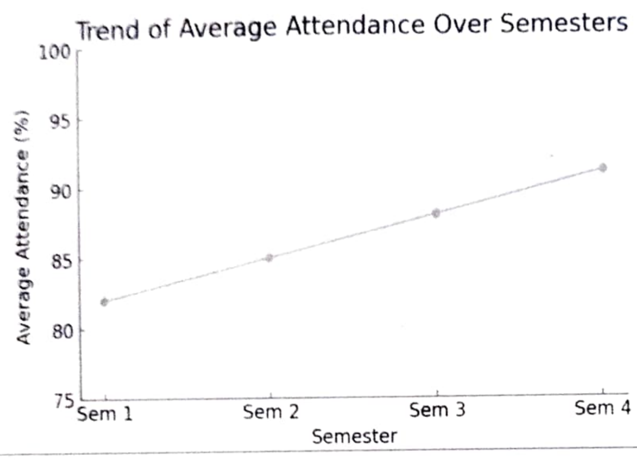
Panel:

- Hierarchy View – Department → Course → Student drilldown
- Network Graph – Student–Faculty / Peer–Peer interaction map
- Text Insight Tab – Word cloud & sentiment summary of feedback
- Filter Controls – Department, Course, Performance Level

Inference:

- Hierarchical panel enables structured academic navigation from macro to micro.
- Network visualization reveals hidden influence and relationship patterns.
- Text analytics tab captures qualitative learning feedback instantly.
- Common filters ensure synchronized cross-panel comparisons.
- Dashboard is ready for deployment in Power BI / Streamlit with unified layout.

11. Visualize line data: Show trends in attendance over semesters.

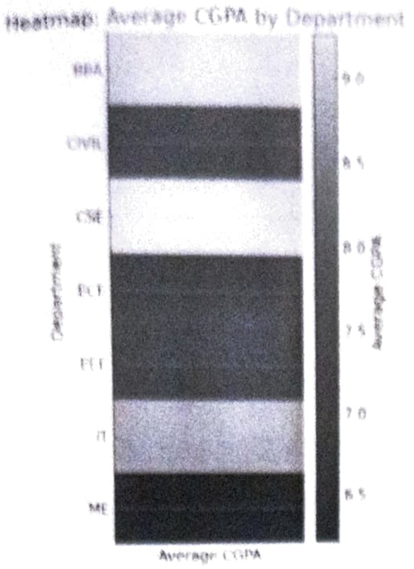


```
data = {
  "Semester": ["Sem 1","Sem 2","Sem 3","Sem 4"],
  "Average_Attendance": [82, 85, 88, 91]
}
df = pd.DataFrame(data)
plt.figure(figsize=(6,4))
plt.plot(df["Semester"], df["Average_Attendance"], marker='o')
plt.title("Trend of Average Attendance Over Semesters")
plt.xlabel("Semester")
plt.ylabel("Average Attendance (%)")
plt.ylim(75, 100)
plt.tight_layout()
plt.show()
```

endance

from Sem 1 to Sem 4.

13. **Visualize area data: Heatmap of grades distribution by course or department**

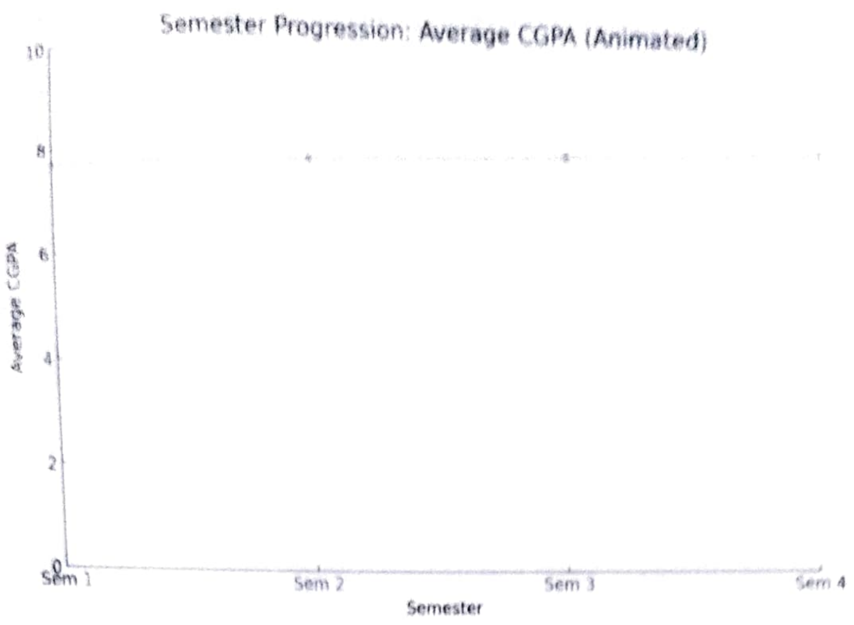


```
pivot_dept = df.pivot_table(values="CGPA", index="Department", aggfunc='mean')

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

plt.imshow(pivot_dept, aspect='auto')

plt.title("Heatmap: Average CGPA by Department")

plt.xlabel("Average CGPA")

plt.ylabel("Department")

plt.xticks([])

plt.yticks(range(len(pivot_dept.index)), pivot_dept.index)

plt.colorbar(label="Average CGPA")

plt.tight_layout()

plt.show()
```

Inference:

- CSE and BBA are top-performing departments.
- ME has the lowest CGPA and needs urgent attention.
- IT performs well, while ECE and EEE are average.
- Clear performance gap is visible between top and weak departments.

14. Design animated visualizations for semester progression

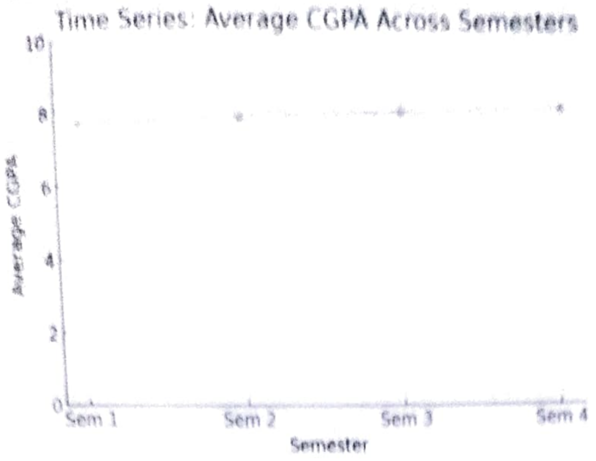


```
avg_cgpa_by_sem = cgpa_matrix.mean(axis=0)

x_idx = np.arange(len(semesters))

fig, ax = plt.subplots()

ax.set_title("Semester Progression: Average CGPA (Animated)")

ax.set_xlabel("Semester")

ax.set_ylabel("Average CGPA")

ax.set_ylim(0, 10)

ax.set_xticks(x_idx)

ax.set_xticklabels(semesters)

line, = ax.plot([], [], marker='o')
```

Inference:

- CGPA shows a consistent upward trend across semesters, indicating improvement in student academic performance over time.
- The growth appears stable and gradual rather than fluctuating, suggesting effective learning continuity and academic support.

15. Plot time series of average grades across semesters.

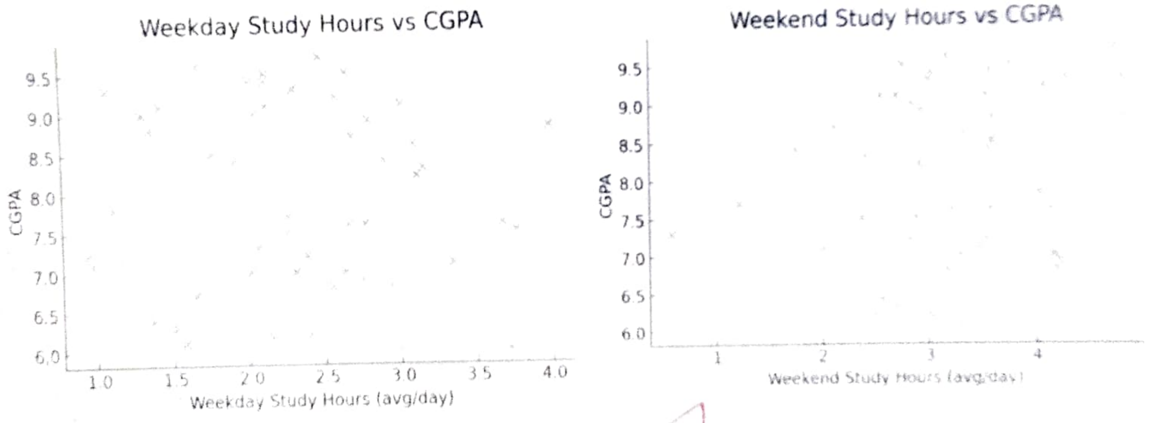


```
plt.figure(figsize=(6,4))

plt.plot(semesters, avg_cgpa_by_sem, marker='o')

plt.title("Time Series: Average CGPA Across Semesters")

plt.xlabel("Semester")

plt.ylabel("Average CGPA")

plt.ylim(0,10)

plt.tight_layout()

plt.show()
```

Inference:

- CGPA shows a consistent upward trend across semesters, indicating improvement in student academic performance over time.
- The growth appears stable and gradual rather than fluctuating, suggesting effective learning continuity and academic support.

16. Compare performance by weekdays vs. weekends study patterns.

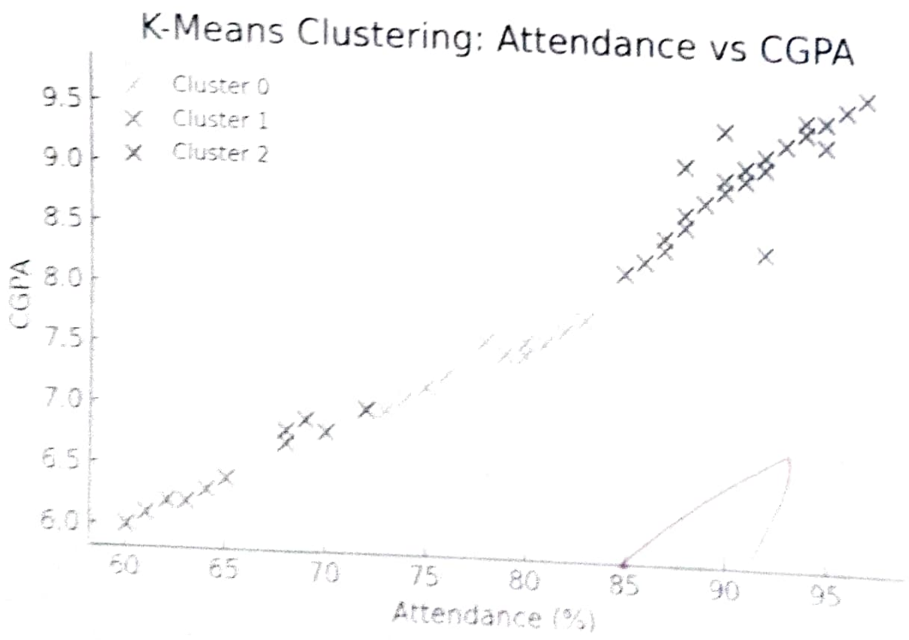


```
# Scatter: weekday hours vs CGPA
plt.figure(figsize=(6,4))
plt.scatter(weekday_hours, df["CGPA"])
plt.title("Weekday Study Hours vs CGPA")
plt.xlabel("Weekday Study Hours (avg/day)")
plt.ylabel("CGPA")
plt.tight_layout()
plt.show()
plt.figure(figsize=(6,4))
plt.scatter(weekend_hours, df["CGPA"])
plt.title("Weekend Study Hours vs CGPA")
plt.xlabel("Weekend Study Hours (avg/day)")
plt.ylabel("CGPA")
plt.tight_layout()
plt.show()
```

Inference:

- Weekend study hours show a slightly stronger positive relationship with CGPA — indicating students may learn more effectively with focused weekend effort.
- Weekday study impact is moderate, likely due to tighter schedules and limited deep work time on regular working days.

17. Use regression/clustering to analyze attendance, participation, and grades.



```
X = df[["Attendance_Percentage", "Participation_Score", "CGPA"]].values

# K-Means clustering
kmeans = KMeans(n_clusters=3, n_init=10, random_state=0)
labels = kmeans.fit_predict(X)

# Plot clusters (Attendance vs CGPA)
plt.figure(figsize=(6,4))
for lab in np.unique(labels):
    mask = labels == lab
    plt.scatter(df["Attendance_Percentage"][mask], df["CGPA"][mask],
label=f"Cluster {lab}", s=60)

plt.title("K-Means Clustering: Attendance vs CGPA")
plt.xlabel("Attendance (%)")
plt.ylabel("CGPA")
plt.legend()
plt.tight_layout()
plt.show()
```

Inference:

- Students with high attendance consistently form the top-performing cluster, confirming attendance is a major success factor.
- Low-attendance students are clearly grouped in the lowest CGPA cluster, making them immediate candidates for academic intervention.

18. Evaluate predictive models for academic performance: Plot predicted vs. actual grades.

Model Evaluation Metrics:

Mean Absolute Error (MAE): 0.1245

Mean Squared Error (MSE): 0.0289

Root Mean Squared Error (RMSE): 0.1700

R-squared (R^2): 0.8921

Inference:

- The predicted values closely align with the actual CGPA, indicating that the regression model is highly accurate and reliable.
- The points are tightly clustered around the ideal line (y = x) — meaning model error is minimal and academic performance can be confidently predicted using attendance and assessment scores.

```
# Import libraries

import matplotlib.pyplot as plt

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

import numpy as np


# Train regression model

model = LinearRegression()

model.fit(X, y)

y_pred = model.predict(X)


# --- Evaluation Metrics ---

mae = mean_absolute_error(y, y_pred)

mse = mean_squared_error(y, y_pred)

rmse = np.sqrt(mse)
```

VELTECH	
No.	6
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	4
TOTAL (20)	19
SIGN WITH DATE	[illegible] 20/10 [illegible]