



Welcome to this session: Task Walkthrough - Tasks 11 - 12

The session will start shortly...

Questions? Drop them in the chat.
We'll have dedicated moderators
answering questions.



Safeguarding & Welfare

We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:



Ian Wyles
Designated Safeguarding
Lead



Simone Botes



Nurhaan Snyman



Rafiq Manan



Ronald Munodawafa



Tevin Pitts

Scan to report a
safeguarding concern



or email the Designated
Safeguarding Lead:
Ian Wyles

safeguarding@hyperiondev.com

Skills Bootcamp Data Science

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: [Questions](#)

Skills Bootcamp Data Science

- For all **non-academic questions**, please submit a query:
www.hyperiondev.com/support
- **Report a safeguarding incident:** **www.hyperiondev.com/safeguardreporting**
- We would love your feedback on lectures: **[Feedback on Lectures](#)**
- If you are hearing impaired, please kindly use your computer's function through Google chrome to enable captions.

Learning Outcomes

- ❖ **Load and explore datasets using pandas** to understand their structure and contents.
- ❖ **Perform basic data manipulations** on DataFrames, such as filtering, sorting, and summarizing data.
- ❖ **Create visualizations using matplotlib and seaborn** to identify trends, patterns, and relationships in data.
- ❖ **Combine DataFrame operations with visualizations** to generate comprehensive data analysis reports.

Task Walkthrough

For today's task you're a data scientist analyzing movie data for a streaming platform. Your task is to:

- ❖ Load and explore a dataset containing information about movies (e.g., title, genre, rating, revenue).
- ❖ Perform basic manipulations to answer key questions like:
 - Which genre has the highest average rating?
 - Which movie had the highest revenue?
 - How many movies were released each year?
- ❖ Visualize trends and insights using matplotlib and seaborn, such as:
 - A bar chart of average ratings by genre.
 - A line graph showing the number of movies released per year.
 - A scatter plot of revenue vs. rating to identify correlations.

What is a DataFrame in pandas?

- A. A table-like data structure with rows and columns
- B. A type of chart for visualising data
- C. A function to filter datasets
- D. A library for machine learning



Which Python library is commonly used for data visualisation?

- A. pandas
- B. seaborn
- C. NumPy
- D. TensorFlow

Datasets



Datasets

- ❖ A dataset is a **structured collection of information** relevant to a specific investigation or project
- ❖ In data science, they provide the **raw material** for **analysis** and **modeling**. Understanding different dataset formats ensures you can work with data from various sources (databases, online repositories, etc.).
- ❖ With the help of **Pandas DataFrames**, we can effortlessly manipulate data to suit our needs.

DataFrames



DataFrames

- ❖ A DataFrame is the way the **Pandas library** in Python represents **tabular data**. It's like a powerful *spreadsheet* within your code.
- ❖ **Rows**: Each row represents a **single observation** or data point (e.g., a person, a product, a transaction).
- ❖ **Columns**: Each column represents a **variable or feature** (e.g., height, price, date). Data within a column usually shares the **same data type** (numbers, text, etc.).

Pandas DataFrame



Pandas DataFrame

- ❖ The pandas' library documentation defines a DataFrame as a “two-dimensional, size-mutable, with labelled rows and columns.”

```
import pandas as pd
```

columns axis=1		column name	more columns to display						
	color	director_name	num_critic_for_reviews	duration	...	actor_2_facebook_likes	imdb_score	aspect_ratio	movie_facebook_likes
0	Color	James Cameron	723.0	178.0	...	936.0	7.9	1.78	33000
1	Color	Gore Verbinski	302.0	169.0	...	5000.0	7.1	2.35	0
2	Color	Sam Mendes	602.0	148.0	...	393.0	6.8	2.35	85000
3	Color	Christopher Nolan	813.0	164.0	...	23000.0	8.5	2.35	164000
4	NaN	Doug Walker	NaN	NaN	...	12.0	7.1	NaN	0

index label

index
axis=0

missing values

data
(values)

Anatomy of a DataFrame

Pandas DataFrame

- ❖ Pandas provides functions like `pd.read_csv()`, `pd.read_excel()`, `pd.read_sql()`, to bring your data directly into your coding environment as DataFrames.
- ❖ This is where you start turning your raw data into something easily workable.

```
import pandas as pd
```

```
# url = 'https://raw.githubusercontent.com/mwaskom/seaborn-data/master/iris.csv'  
# df = pd.read_csv(url)
```

```
iris = datasets.load_iris()
```

```
df = pd.DataFrame(iris.data, columns=iris.feature_names)
```

Exploring datasets

- ❖ **df.head(), df.tail():** Peek at the top and bottom rows for initial understanding

```
df.head()
```

✓ 0.0s

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

Exploring datasets

- ❖ **df.head(), df.tail()**: Peek at the top and bottom rows for initial understanding

```
df.tail()
```

✓ 0.0s

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

Exploring datasets

- ❖ **df.shape:** Tells you the dimensions (rows, columns) of your data.

```
df.shape
```

```
✓ 0.0s
```

```
(150, 5)
```

Exploring datasets

- ❖ **df.info():** Gives the **data types** of each column, and if columns have missing values

```
df.info()
✓ 0.0s

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
#   Column                Non-Null Count  Dtype
---  -
0   sepal length (cm)     150 non-null   float64
1   sepal width (cm)      150 non-null   float64
2   petal length (cm)     150 non-null   float64
3   petal width (cm)      150 non-null   float64
4   species               150 non-null   int64
dtypes: float64(4), int64(1)
memory usage: 6.0 KB
```

Exploring datasets

- ❖ **df.describe():** Quick summary statistics for numerical columns.

```
df.describe()
```

✓ 0.0s

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333	1.000000
std	0.828066	0.435866	1.765298	0.762238	0.819232
min	4.300000	2.000000	1.000000	0.100000	0.000000
25%	5.100000	2.800000	1.600000	0.300000	0.000000
50%	5.800000	3.000000	4.350000	1.300000	1.000000
75%	6.400000	3.300000	5.100000	1.800000	2.000000
max	7.900000	4.400000	6.900000	2.500000	2.000000

Manipulating Data



Manipulating Data

- ❖ **Selecting Columns:** You often work with a **subset of features**.
- ❖ Using `df[['column1', 'column2']]` gets you only specific columns.

```
df.columns
```

```
✓ 0.0s
```

```
Index(['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)',  
      'petal width (cm)', 'species'],  
      dtype='object')
```

```
# Select specific columns
```

```
df_selected = df[['species', 'petal length (cm)', 'petal width (cm)']]
```

```
✓ 0.0s
```


Manipulating Data

- ❖ **Filtering Rows:** Focus on specific subsets meeting certain conditions, e.g., `df[df['species'] == 'setosa']`

```
# Filter by flower species
```

```
df_setosa = df[df['species'] == 'setosa']
```

✓ 0.0s

Manipulating Data

- ❖ **Creating New Columns:** Derived features, e.g., calculating area from length and width.

```
# Create a new calculated column
```

```
df['petal area (cm^2)'] = df['petal length (cm)'] * df['petal width (cm)']
```

```
✓ 0.0s
```


Manipulating Data

- ❖ **Renaming/Dropping:** Improve clarity or get rid of unneeded data.

```
# Rename a column
```

```
df = df.rename(columns={'sepal length (cm)': 'sepal_len'})
```

```
✓ 0.0s
```

- ❖ Data manipulation gives you a **highly customized DataFrame** focused on your exact analysis needs.

Built-in Methods

- ❖ Pandas offers a toolbox of functions for calculations:
 - **mean()** - Computes the mean for each column.
 - **min()** - Computes the minimum for each column.
 - **max()** - Computes the maximum for each column.
 - **std()** - Computes the standard deviation for each column.
 - **var()** - Computes the variance for each column.
 - **unique()** - Computes the number of unique values in each column.
- ❖ This is the start of understanding the characteristics of your data.

Grouping and Aggregation

- ❖ `df.groupby()`: Divide your data **based on categories** in a column (e.g., group by species).

```
print(df['petal area (cm^2)'].mean())  
print(df['species'].nunique())  
print(df.groupby('species')['petal length (cm)'].std())
```

✓ 0.0s

5.794066666666667

3

species

0 0.173664

1 0.469911

2 0.551895

Name: petal length (cm), dtype: float64

Grouping and Aggregation

- ❖ **.agg()**: Apply calculations within each group (e.g., average length, maximum width).

```
df.groupby('species').agg(  
    mean_petal_length=('petal length (cm)', 'mean'),  
    max_sepal_width=('sepal width (cm)', 'max')  
)
```

✓ 0.0s

	mean_petal_length	max_sepal_width
species		
0	1.462	4.4
1	4.260	3.4
2	5.552	3.8

Matplotlib



Matplotlib

Installation

```
pip install matplotlib
```

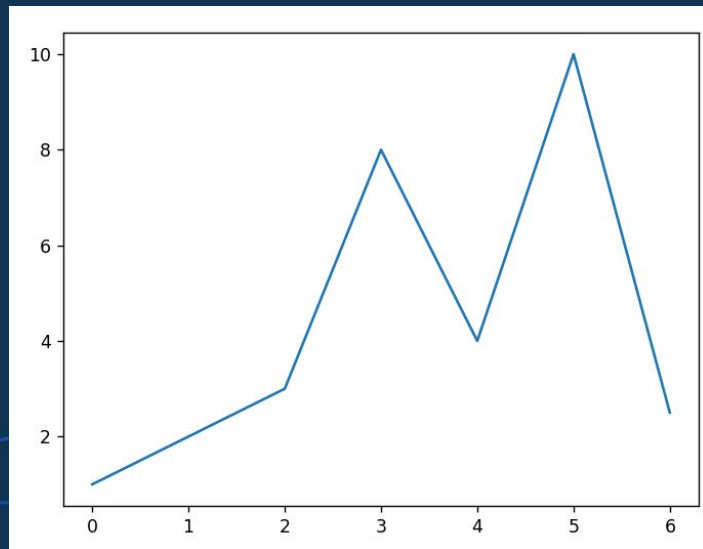
Importing

```
import matplotlib.pyplot as plt
```

```
import matplotlib.pyplot as plt
import numpy as np

ypoints = np.array([1, 2, 3, 8, 4, 10, 2.5])

plt.plot(ypoints)
plt.show()
```



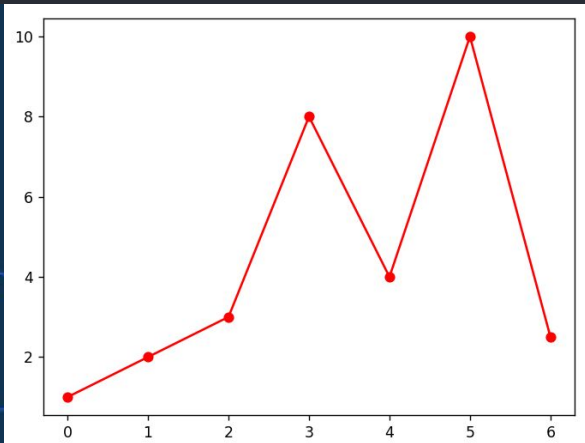
Matplotlib

```
import matplotlib.pyplot as plt
import numpy as np

ypoints = np.array([1, 2, 3, 8, 4, 10, 2.5])

plt.plot(ypoints, 'o-r')

plt.show()
```



Markers

o = Circle, * = Star,
. = Point, x = Cross,
s = Square, d = diamond

Linestyle ls

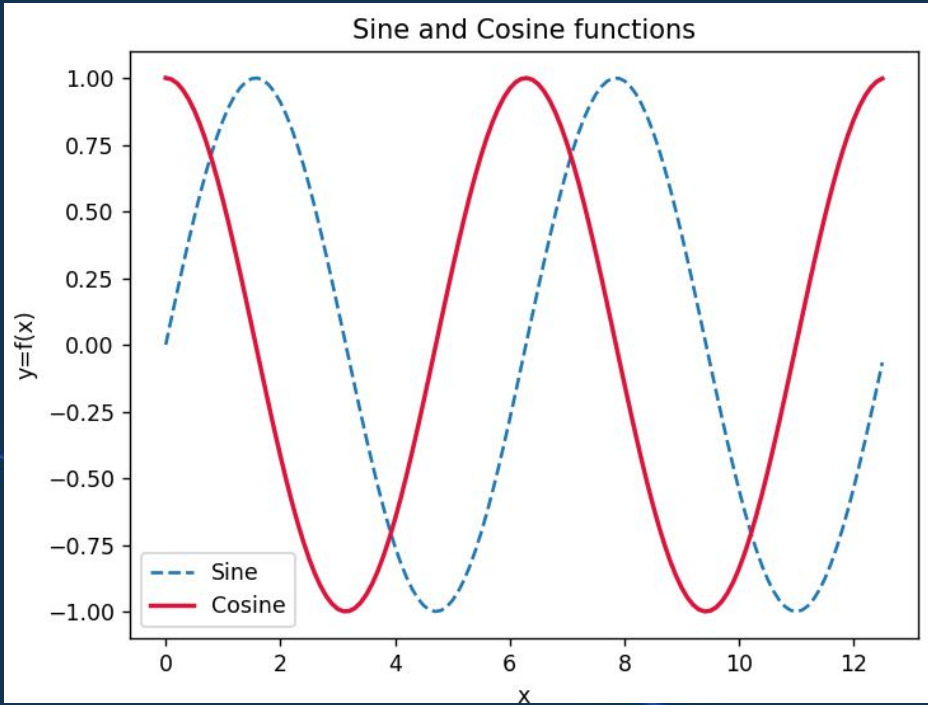
- Solid line
: Dotted line
-- Dashed line
-. Dashed/dotted line

Colour

List of colours

b = Blue, r = Red, g = Green,
c = Cyan, m = Magenta, y =
Yellow, k = Black, w = White

Matplotlib



```
import matplotlib.pyplot as plt
import numpy as np
```

```
pi = np.pi
```

```
# Creating x axis with range
```

```
#and y axis with sine/cosine
```

```
x = np.arange(0, 4*pi, 0.1)
```

```
y1 = np.sin(x)
```

```
y2 = np.cos(x)
```

```
#Plotting sine/cosine on the same plot
```

```
plt.plot(x,y1, label='Sine', ls='--')
```

```
plt.plot(x,y2, label='Cosine', lw=2, c='crimson')
```

```
#x-axis, y-axis label, and plot title
```

```
plt.xlabel('x')
```

```
plt.ylabel('y=f(x)')
```

```
plt.title('Sine and Cosine functions')
```

```
#Legend for the two curves
```

```
plt.legend()
```

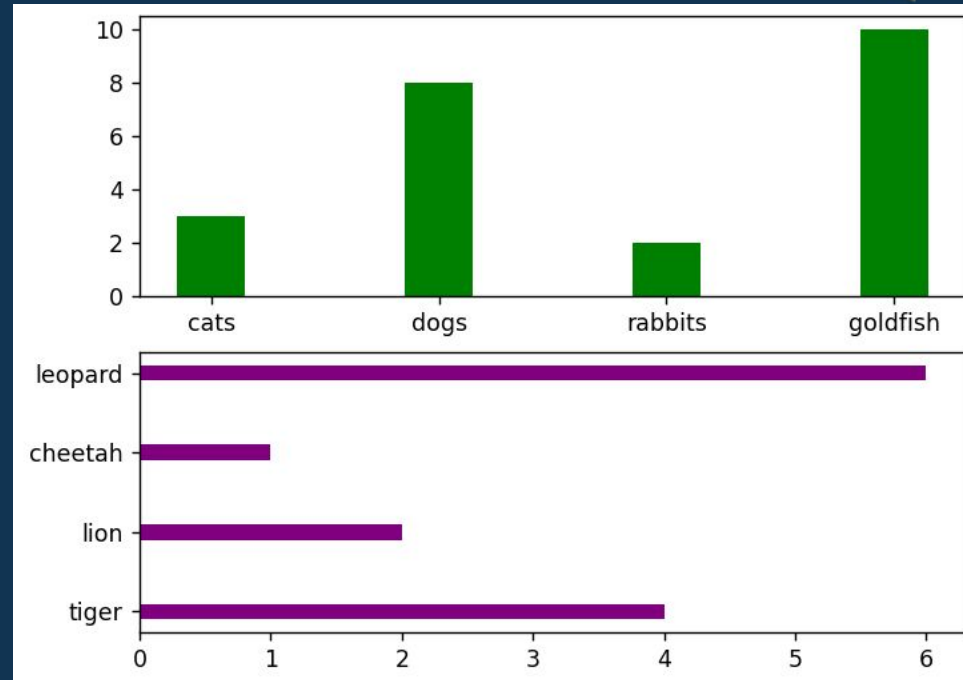
```
plt.show() #plt.savefig('sine.png')
```


Matplotlib: Barplot

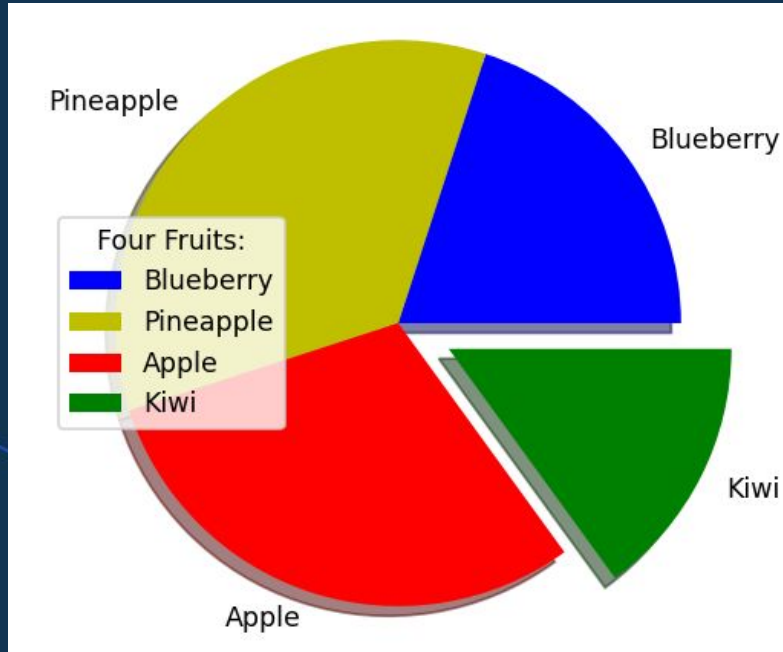
```
import matplotlib.pyplot as plt
import numpy as np

#x,y for 1st and 2nd barplots
x1 = np.array(["cats", "dogs", "rabbits", "goldfish"])
y1 = np.array([3, 8, 2, 10])
x2 = np.array(["tiger", "lion", "cheetah", "leopard"])
y2 = np.array([4, 2, 1, 6])

#Subplot, parameters(rows, columns, index of current plot)
plt.subplot(2,1,1)
plt.bar(x1, y1, color = 'g', width=0.3)
plt.subplot(2,1,2)
plt.barh(x2, y2, color = '#800080', height=0.2)
plt.show()
```



Matplotlib: Pie chart



```
import matplotlib.pyplot as plt
import numpy as np

y = np.array([20, 35, 30, 15])
mylabels = ['Blueberry', 'Pineapple', 'Apple', 'Kiwi']
mycolors = ['b', 'y', 'r', 'g']
myexplode = [0, 0, 0, 0.2]

plt.pie(y, labels = mylabels, colors = mycolors,
        explode = myexplode, shadow = True)
plt.legend(loc='center left', title = "Four Fruits:")
plt.show()
```

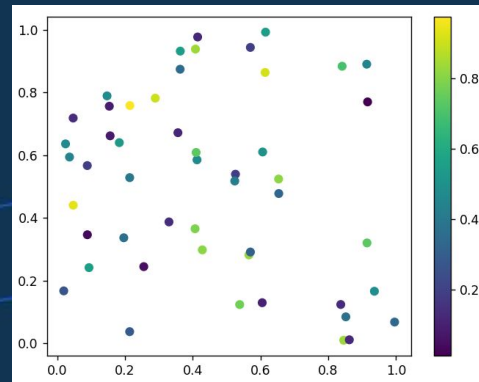
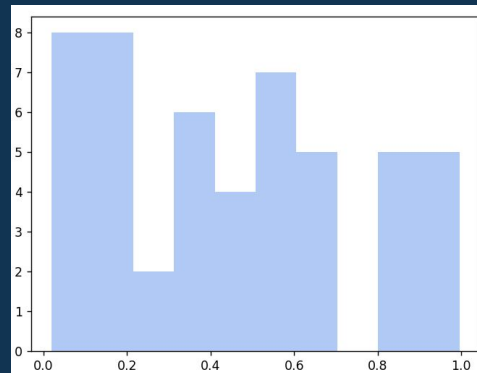
Histograms & Scatterplots

A graph showing frequency distributions, the number of observations within each given interval.

```
N = 50
x = np.random.rand(N)
y = np.random.rand(N)
colors = np.random.rand(N)

plt.hist(x, color='cornflowerblue', alpha=0.5)
plt.show()

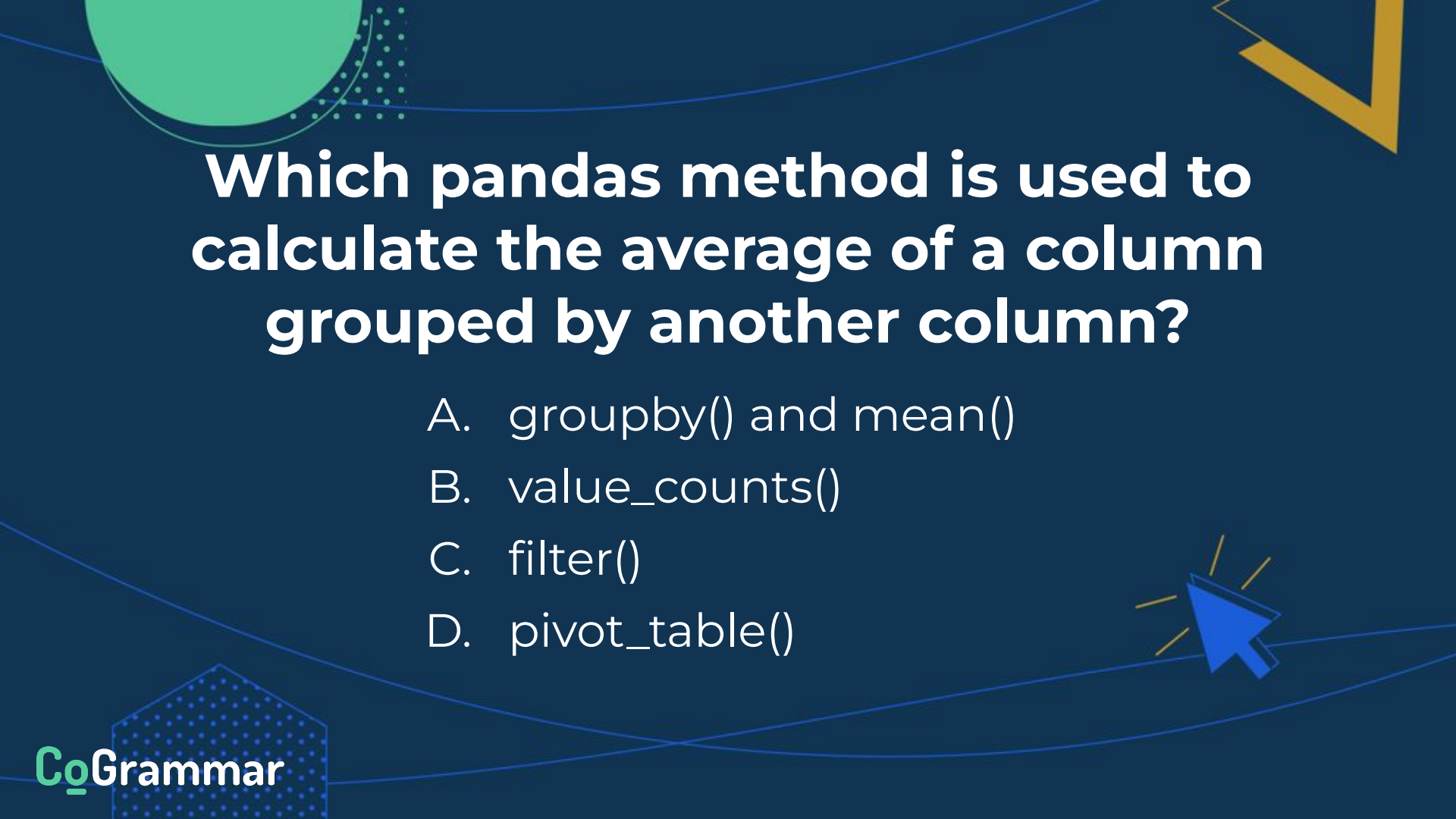
plt.scatter(x, y, c=colors)
plt.colorbar()
plt.show()
```



Task Walkthrough

For today's task you're a data scientist analyzing movie data for a streaming platform. Your task is to:

- ❖ Load and explore a dataset containing information about movies (e.g., title, genre, rating, revenue).
- ❖ Perform basic manipulations to answer key questions like:
 - Which genre has the highest average rating?
 - Which movie had the highest revenue?
 - How many movies were released each year?
- ❖ Visualize trends and insights using matplotlib and seaborn, such as:
 - A bar chart of average ratings by genre.
 - A line graph showing the number of movies released per year.
 - A scatter plot of revenue vs. rating to identify correlations.



Which pandas method is used to calculate the average of a column grouped by another column?

- A. `groupby()` and `mean()`
- B. `value_counts()`
- C. `filter()`
- D. `pivot_table()`



What is the purpose of a scatter plot?

- A. To compare categories
- B. To show trends over time
- C. To display relationships between two numerical variables
- D. To count occurrences of data points

Summary

- ★ **User-Defined Functions:**
Modularizing operations like data validation and analysis.
- ★ **Strings:**
Validating email formats and formatting output.
- ★ **Lists and Dictionaries:**
Storing participant objects and managing survey questions and responses.
- ★ **File I/O:**
Reading and saving data to/from files for persistence.
- ★ **OOP Classes:**
Modeling real-world entities and encapsulating behavior with methods.

CoGrammar

Q & A SECTION

**Please use this time to ask
any questions relating to the
topic, should you have any.**

Thank you for attending



CoGrammar



Department
for Education