

# **EDS Theory Activity 1**

**Dataset :- Opin Rank Review**

**Name: Shravani**

**Kulkarni**

**Roll no. :- CS3-16**

**PRN: 202401040136**

**Division: CS3**

# Python Code for Given Dataset:

```
Opin Rank Review Dataset.ipynb
File Edit View Insert Runtime Tools Help

Q Commands + Code + Text

import pandas as pd
import numpy as np

np.random.seed(42)

data = {
    'Review_ID': range(1, 9),
    'Car_Model': np.random.choice(['Toyota Camry', 'Honda Accord', 'Ford Focus', 'Tesla Model 3'], 8),
    'Author': np.random.choice(['Alice', 'Bob', 'Charlie', 'David', 'Eva'], 8),
    'Review_Date': pd.date_range(start='2024-01-01', periods=8, freq='W'),
    'Review_Content': np.random.choice([
        'Amazing car with superb comfort!',
        'Good fuel economy and stylish design.',
        'Performance is top-notch.',
        'Interior could be better.',
        'Great value for money.'
    ], 8),
    'Overall_Rating': np.random.randint(1, 6, 8),
    'Comfort': np.random.randint(1, 6, 8),
    'Performance': np.random.randint(1, 6, 8),
    'Fuel_Economy': np.random.randint(1, 6, 8),
    'Value_for_Money': np.random.randint(1, 6, 8),
    'Exterior_Styling': np.random.randint(1, 6, 8),
    'Interior_Design': np.random.randint(1, 6, 8),
    'Features': np.random.randint(1, 6, 8)
}

df = pd.DataFrame(data)

print("Initial OpinRank-like Dataset:")
print(df)
```

## Output:

Initial OpinRank-like Dataset:

	Review_ID	Car_Model	Author	Review_Date	\
0	1	Ford Focus	Bob	2024-01-07	
1	2	Tesla Model 3	Charlie	2024-01-14	
2	3	Toyota Camry	Charlie	2024-01-21	
3	4	Ford Focus	Charlie	2024-01-28	
4	5	Ford Focus	Eva	2024-02-04	
5	6	Tesla Model 3	David	2024-02-11	
6	7	Toyota Camry	Charlie	2024-02-18	
7	8	Toyota Camry	Eva	2024-02-25	

	Review_Content	Overall_Rating	Comfort	\
0	Good fuel economy and stylish design.	5	4	
1	Interior could be better.	4	3	
2	Good fuel economy and stylish design.	1	4	
3	Interior could be better.	1	4	
4	Great value for money.	3	1	
5	Amazing car with superb comfort!	3	3	
6	Interior could be better.	2	5	
7	Good fuel economy and stylish design.	4	3	


	Performance	Fuel_Economy	Value_for_Money	Exterior_Styling	\
0	5	1	5	4	
1	1	2	3	5	
2	2	5	1	2	
3	4	2	4	2	
4	1	4	2	4	
5	4	4	4	2	
6	2	4	2	2	
7	2	4	2	4	


	Interior_Design	Features
0	4	4
1	1	4
2	5	4
3	5	5
4	2	1
5	5	5
6	2	5
7	1	1

# Problem Statements:

## Problem 1:


Find the total number of reviews.


```
0s  print("\nProblem 1: Total number of reviews:")
print(len(df))
```

```
 Problem 1: Total number of reviews:
8
```

## Problem 2:


Find unique car models reviewed.


```
0s  print("\nProblem 2: Unique car models:")
print(df['Car_Model'].unique())
```

```
 Problem 2: Unique car models:
['Ford Focus' 'Tesla Model 3' 'Toyota Camry']
```

## Problem 3:


Find the average Overall Rating.


```
0s  print("\nProblem 3: Average Overall Rating:")
print(df['Overall_Rating'].mean())
```

```
 Problem 3: Average Overall Rating:
2.875
```

#### Problem 4:

Find the maximum Comfort rating.


```
0s  print("\nProblem 4: Maximum Comfort Rating:")
print(df['Comfort'].max())
```

```
 Problem 4: Maximum Comfort Rating:
5
```

#### Problem 5:

Find the minimum Fuel Economy rating.


```
0s [7] print("\nProblem 5: Minimum Fuel Economy Rating:")
print(df['Fuel_Economy'].min())
```

```
 Problem 5: Minimum Fuel Economy Rating:
1
```

#### Problem 6:

Find how many reviews were written by 'Alice'.

```
0s [8] print("\nProblem 6: Number of reviews by Alice:")
print(df[df['Author'] == 'Alice'].shape[0])
```

```
 Problem 6: Number of reviews by Alice:
0
```

## Problem 7:

List all reviews with Overall Rating greater than 3.

```
[9] print("\nProblem 7: Reviews with Overall Rating > 3:")
    print(df[df['Overall_Rating'] > 3])
```

↔

Problem 7: Reviews with Overall Rating > 3:

	Review_ID	Car_Model	Author	Review_Date	\
0	1	Ford Focus	Bob	2024-01-07	
1	2	Tesla Model 3	Charlie	2024-01-14	
7	8	Toyota Camry	Eva	2024-02-25	

	Review_Content	Overall_Rating	Comfort	\
0	Good fuel economy and stylish design.	5	4	
1	Interior could be better.	4	3	
7	Good fuel economy and stylish design.	4	3	

	Performance	Fuel_Economy	Value_for_Money	Exterior_Styling	\
0	5	1	5	4	
1	1	2	3	5	
7	2	4	2	4	

	Interior_Design	Features
0	4	4
1	1	4
7	1	1

## Problem 8:

Sort the dataset by Review\_Date in descending order.

```
[10] print("\nProblem 8: Dataset sorted by Review_Date (Descending):")
print(df.sort_values('Review_Date', ascending=False))
```

Problem 8: Dataset sorted by Review\_Date (Descending):

	Review_ID	Car_Model	Author	Review_Date	\
7	8	Toyota Camry	Eva	2024-02-25	
6	7	Toyota Camry	Charlie	2024-02-18	
5	6	Tesla Model 3	David	2024-02-11	
4	5	Ford Focus	Eva	2024-02-04	
3	4	Ford Focus	Charlie	2024-01-28	
2	3	Toyota Camry	Charlie	2024-01-21	
1	2	Tesla Model 3	Charlie	2024-01-14	
0	1	Ford Focus	Bob	2024-01-07	

	Review_Content	Overall_Rating	Comfort	\
7	Good fuel economy and stylish design.	4	3	
6	Interior could be better.	2	5	
5	Amazing car with superb comfort!	3	3	
4	Great value for money.	3	1	
3	Interior could be better.	1	4	
2	Good fuel economy and stylish design.	1	4	
1	Interior could be better.	4	3	
0	Good fuel economy and stylish design.	5	4	

	Performance	Fuel_Economy	Value_for_Money	Exterior_Styling	\
7	2	4	2	4	
6	2	4	2	2	
5	4	4	4	2	
4	1	4	2	4	
3	4	2	4	2	
2	2	5	1	2	
1	1	2	3	5	
0	5	1	5	4	

	Interior_Design	Features
7	1	1
6	2	5
5	5	5
4	2	1
3	5	5
2	5	4
1	1	4
0	4	4

### Problem 9:

Find the average Fuel Economy rating for Tesla Model 3.

```
[11] print("\nProblem 9: Average Fuel Economy for Tesla Model 3:")
      print(df[df['Car_Model'] == 'Tesla Model 3']['Fuel_Economy'].mean())
```

Problem 9: Average Fuel Economy for Tesla Model 3:  
3.0

### Problem 10:

Get the review with the highest Overall Rating.

```
[12] print("\nProblem 10: Review with Highest Overall Rating:")
      print(df[df['Overall_Rating'] == df['Overall_Rating'].max()])
```

Problem 10: Review with Highest Overall Rating:

Review_ID	Car_Model	Author	Review_Date	
0	1	Ford Focus	Bob	2024-01-07

	Review_Content	Overall_Rating	Comfort	
0	Good fuel economy and stylish design.	5	4	

	Performance	Fuel_Economy	Value_for_Money	Exterior_Styling	
0	5	1	5	4	

	Interior_Design	Features
0	4	4

### Problem 11:

Add a new column "Total\_Score" = Sum of all feature ratings  
(Comfort, Performance, etc.)

```
0s df['Total_Score'] = df[['Comfort', 'Performance', 'Fuel_Economy', 'Value_for_Money', 'Exterior_Styling', 'Interior_Design', 'Features']].sum(axis=1)

print("\nProblem 11: Dataset with Total_Score column:")
print(df[['Review_ID', 'Total_Score']])
```

Problem 11: Dataset with Total\_Score column:

	Review_ID	Total_Score
0	1	27
1	2	19
2	3	23
3	4	26
4	5	15
5	6	27
6	7	22
7	8	17

### Problem 12:

Find the author who gave the worst Overall Rating.

```
0s print("\nProblem 12: Author with worst Overall Rating:")
print(df[df['Overall_Rating'] == df['Overall_Rating'].min()]['Author'])
```

Problem 12: Author with worst Overall Rating:

2	Charlie
3	Charlie

Name: Author, dtype: object

### Problem 13:

Find the average Value for Money rating.

```
0s [15] print("\nProblem 13: Average Value for Money Rating:")
print(df['Value_for_Money'].mean())
```

Problem 13: Average Value for Money Rating:

2.875



### Problem 14:

Count the number of reviewers per car model.

```
✓ 0s [17] print("\nProblem 14: Number of reviewers per car model:")
print(df['Car_Model'].value_counts())
```

⇒

```
Problem 14: Number of reviewers per car model:
Car_Model
Ford Focus      3
Toyota Camry    3
Tesla Model 3    2
Name: count, dtype: int64
```

### Problem 15:

Find reviews where Comfort and Performance both are greater than 3.

```
✓ 0s [17] print("\nProblem 15: Reviews with Comfort > 3 and Performance > 3:")
print(df[(df['Comfort'] > 3) & (df['Performance'] > 3)])
```

⇒

```
Problem 15: Reviews with Comfort > 3 and Performance > 3:
  Review_ID  Car_Model  Author  Review_Date \
0          1  Ford Focus    Bob   2024-01-07
3          4  Ford Focus  Charlie 2024-01-28

  Review_Content  Overall_Rating  Comfort \
0  Good fuel economy and stylish design.    5    4
3          Interior could be better.    1    4

  Performance  Fuel_Economy  Value_for_Money  Exterior_Styling \
0            5            1            5            4
3            4            2            4            2

  Interior_Design  Features  Total_Score
0                4        4            27
3                5        5            26
```

### Problem 16:

Find the earliest review date.

```
[18] print("\nProblem 16: Earliest review date:")
      print(df['Review_Date'].min())
```

Problem 16: Earliest review date:  
2024-01-07 00:00:00

### Problem 17:

Get top 3 reviews by Total\_Score.

```
print("\nProblem 17: Top 3 reviews by Total_Score:")
print(df.sort_values('Total_Score', ascending=False).head(3))
```

Problem 17: Top 3 reviews by Total\_Score:

	Review_ID	Car_Model	Author	Review_Date	\
0	1	Ford Focus	Bob	2024-01-07	
5	6	Tesla Model 3	David	2024-02-11	
3	4	Ford Focus	Charlie	2024-01-28	

	Review_Content	Overall_Rating	Comfort	\
0	Good fuel economy and stylish design.	5	4	
5	Amazing car with superb comfort!	3	3	
3	Interior could be better.	1	4	

	Performance	Fuel_Economy	Value_for_Money	Exterior_Styling	\
0	5	1	5	4	
5	4	4	4	2	
3	4	2	4	2	

	Interior_Design	Features	Total_Score
0	4	4	27
5	5	5	27
3	5	5	26

### Problem 18:

Replace all Overall Ratings less than 3 with 'Low'.

```
0s df['Rating_Label'] = np.where(df['Overall_Rating'] < 3, 'Low', 'Good')

print("\nProblem 18: Dataset with Rating_Label column:")
print(df[['Review_ID', 'Overall_Rating', 'Rating_Label']])
```

Problem 18: Dataset with Rating\_Label column:

	Review_ID	Overall_Rating	Rating_Label
0	1	5	Good
1	2	4	Good
2	3	1	Low
3	4	1	Low
4	5	3	Good
5	6	3	Good
6	7	2	Low
7	8	4	Good

### Problem 19:

Find mean ratings for each car model grouped together.

```
0s [21] print("\nProblem 19: Mean Overall Ratings grouped by Car Model:")
      print(df.groupby('Car_Model')['Overall_Rating'].mean())
```

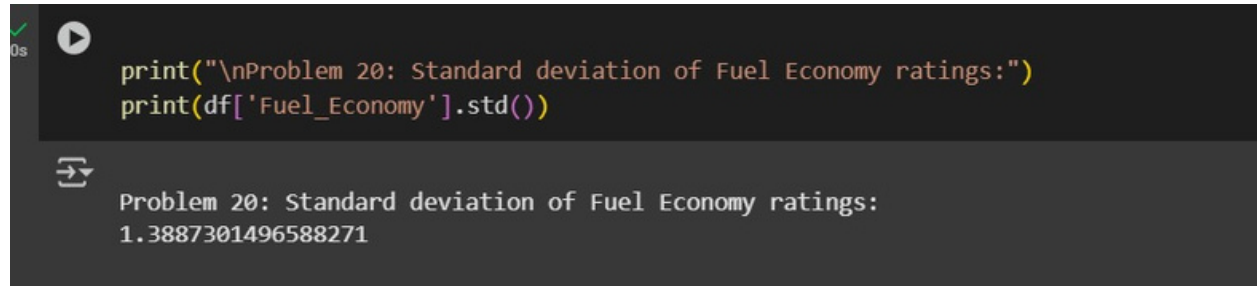
Problem 19: Mean Overall Ratings grouped by Car Model:

Car_Model	
Ford Focus	3.000000
Tesla Model 3	3.500000
Toyota Camry	2.333333

Name: Overall\_Rating, dtype: float64

### Problem 20:

Find standard deviation of Fuel Economy ratings.



```
print("\nProblem 20: Standard deviation of Fuel Economy ratings:")
print(df['Fuel_Economy'].std())
```

Problem 20: Standard deviation of Fuel Economy ratings:  
1.3887301496588271