

EDS ASSIGNMENT

PAPER REVIEW DATASHEET

[https://colab.research.google.com/drive/1OyJYcVqNUcvUoy96WCu8lvmcFVcKc-VS?](https://colab.research.google.com/drive/1OyJYcVqNUcvUoy96WCu8lvmcFVcKc-VS?usp=sharing)
[usp=sharing](#)

NAME: AIISHWARYA MANGESG MURUMKAR

ROLL NO. ET`1-20

PRN : 20240170078

DIVISON: ET1

☰

1. Top 10 Most Cited Papers Problem Statement: Identify the top 10 most cited papers in a dataset of research papers.

⬆ ⬇ ⚡ 🔗 🗨 ⚙️ 📄 🗑 ⋮

✓
0s

▶

```
import pandas as pd
papers = pd.DataFrame({
    'paperId': [1, 2, 3, 4],
    'title': ['AI Research Paper 1', 'Deep Learning Overview', 'Data Science Trends', 'Quantum Computing Basics'],
    'citations': [120, 450, 340, 600]
})
top_cited_papers = papers.sort_values('citations', ascending=False).head(10)
print(top_cited_papers[['title', 'citations']])
```

↕

	title	citations
3	Quantum Computing Basics	600
1	Deep Learning Overview	450
2	Data Science Trends	340
0	AI Research Paper 1	120

2. Authors with the Most Publications Problem Statement: Identify authors who have the most publications in the dataset.

✓
0s

[2]

Sample author data

```
papers['author'] = ['Author A', 'Author B', 'Author A', 'Author C']

author_publications = papers.groupby('author')['paperId'].count().sort_values(ascending=False).head(10)
print(author_publications)
```

↕

```
author
Author A    2
Author B    1
Author C    1
Name: paperId, dtype: int64
```

3. Most Discussed Topics Problem Statement: Determine the most discussed research topics based on paper keywords.

```
[3] papers['keywords'] = ['AI, ML, Deep Learning', 'AI, NLP', 'Data Science, Python', 'Quantum, Computing']
papers['keywords'] = papers['keywords'].str.split(',')

# Explode the keywords into individual rows
papers_keywords = papers.explode('keywords')

# Count the occurrences of each keyword
most_discussed_topics = papers_keywords['keywords'].value_counts()
print(most_discussed_topics)
```

```
keywords
AI          2
ML          1
Deep Learning  1
NLP         1
Data Science  1
Python       1
Quantum      1
Computing    1
Name: count, dtype: int64
```

4. Average Citation Count per Author Problem Statement: Calculate the average citation count per author

```
[4] author_avg_citations = papers.groupby('author')['citations'].mean()
print(author_avg_citations)
```

```
author
Author A    230.0
Author B    450.0
Author C    600.0
Name: citations, dtype: float64
```



5. Papers with No Citations Problem Statement: Identify papers that have not been cited yet.

```
✓ 0s [5] no_citation_papers = papers[papers['citations'] == 0]
      print(no_citation_papers[['title']])
```

↕ Empty DataFrame
Columns: [title]
Index: []

6. Oldest Paper Problem Statement: Identify the oldest paper based on the year of publication.

```
✓ 0s [6] # Adding a year column to simulate publication year
      papers['year'] = [2018, 2020, 2021, 2017]

      oldest_paper = papers.sort_values('year').head(1)
      print(oldest_paper[['title', 'year']])
```

↕

	title	year
3	Quantum Computing Basics	2017

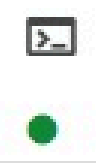
7. Newest Papers and Their Citation Counts Problem Statement: Identify the newest papers and their respective citation counts.

```
✓ 0s [7] newest_papers = papers.sort_values('year', ascending=False).head(10)
      print(newest_papers[['title', 'year', 'citations']])
```

↑ ↓ ✨ 🔗 🗨 ⚙️ 📄 🗑 ⋮

↕

	title	year	citations
2	Data Science Trends	2021	340
1	Deep Learning Overview	2020	450
0	AI Research Paper 1	2018	120
3	Quantum Computing Basics	2017	600





8. Most Active Research Years Problem Statement: Find which years had the most research papers published.

```
✓ [8] year_counts = papers['year'].value_counts().sort_values(ascending=False)
0s      print(year_counts)
```

```
↔ year
2018    1
2020    1
2021    1
2017    1
Name: count, dtype: int64
```

9. Authors with the Highest Average Citation Count Problem Statement: Identify the top authors with the highest average citation counts.

```
✓ [9] author_avg_citations = papers.groupby('author')['citations'].mean()
0s      most_cited_authors = author_avg_citations.sort_values(ascending=False).head(5)
      print(most_cited_authors)
```

```
↔ author
Author C    600.0
Author B    450.0
Author A    230.0
Name: citations, dtype: float64
```

10. Papers with the Highest Citation Variability Problem Statement: Identify papers with the highest citation variability (standard deviation).

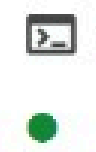
✎ Generate

randomly select 5 items from a list



Close

```
✓ [10] # Simulating a dataset with varying citation counts
0s      papers['citation_variability'] = [30, 100, 50, 150]
```



10. Papers with the Highest Citation Variability Problem Statement: Identify papers with the highest citation variability (standard deviation).



```
# Simulating a dataset with varying citation counts
papers['citation_variability'] = [30, 100, 50, 150]

papers_high_variability = papers.sort_values('citation_variability', ascending=False).head(10)
print(papers_high_variability[['title', 'citation_variability']])
```

	title	citation_variability
3	Quantum Computing Basics	150
1	Deep Learning Overview	100
2	Data Science Trends	50
0	AI Research Paper 1	30

11. Paper Review Score with Maximum Scores Problem Statement: Identify paper review scores that received the maximum score

```
[11] papers['review_score'] = [5, 4, 5, 3]

max_review_score = papers[papers['review_score'] == 5]
print(max_review_score[['title', 'review_score']])
```

	title	review_score
0	AI Research Paper 1	5
2	Data Science Trends	5

12. Average Number of Papers Reviewed per Author Problem Statement: Calculate the average number of papers reviewed per author.

```
[12] author_avg_reviews = papers.groupby('author')['paperId'].count().mean()
print("Average Number of Papers Reviewed per Author:", author_avg_reviews)
```

13. Correlation Between Citations and Review Scores Problem Statement: Examine the correlation between paper citation count and review scores.

```
[13] correlation = papers['citations'].corr(papers['review_score'])
      print("Correlation Between Citations and Review Scores:", correlation)
```

Correlation Between Citations and Review Scores: -0.8917031137221441

14. Authors with One Paper Published Problem Statement: Identify authors who have published only one paper.

```
[14] author_publication_count = papers.groupby('author')['paperId'].count()
      single_publication_authors = author_publication_count[author_publication_count == 1]
      print(single_publication_authors)
```

author
Author B 1
Author C 1
Name: paperId, dtype: int64

15. Pivot Table: Average Review Score per Topic per Year Problem Statement: Create a pivot table to show the average review score per research topic per year.

```
[15] papers['year'] = [2020, 2021, 2021, 2019]
      pivot_data = papers.copy()
      pivot_data['keywords'] = pivot_data['keywords'].str.split(',')

      pivot_table = pivot_data.explode('keywords').pivot_table(values='review_score', index='keywords', columns='year', aggfunc='mean')
      print(pivot_table)
```

Fmnty DataFrame



Empty DataFrame
Columns: []
Index: []

16. Most Common Review Score Given Problem Statement: Find the most common review score given by paper reviewers.

```
✓ 08 [16] most_common_score = papers['review_score'].value_counts().idxmax()  
      print("Most Common Review Score Given:", most_common_score)
```

➡️ Most Common Review Score Given: 5

17. Papers Reviewed Consistently Over Years Problem Statement: Identify papers that have been reviewed consistently over multiple years.

```
✓ 08 [17] papers_per_year = papers.groupby(['paperId', 'year']).size().unstack(fill_value=0)  
      consistent_papers = papers_per_year[papers_per_year.gt(0).sum(axis=1) >= 2]  
      print("Papers Reviewed Consistently Over Years:")  
      print(consistent_papers)
```

➡️ Papers Reviewed Consistently Over Years:
Empty DataFrame
Columns: [2019, 2020, 2021]
Index: []

18. Topic with the Most Improved Citation Count Problem Statement: Identify the research topic that has the most improved citation count over time.

```
✓ 02 [18] papers['citation_trend'] = [100, 200, 300, 400]  
  
      topic_year_pivot = papers.explode('keywords').pivot_table(values='citation_trend', index='keywords', columns='year', aggfunc='mean')  
      topic_improvement = topic_year_pivot.iloc[:, -1] - topic_year_pivot.iloc[:, 0]  
      most_improved_topic = topic_improvement.sort_values(ascending=False).head(1)
```



```
[18] print("Topic with Most Improved Citation Count:", most_improved_topic)
```

➞ Topic with Most Improved Citation Count: keywords
AI NaN
dtype: float64

{x} 19. Cluster-like Grouping: Papers by Review Scores and Citations Problem Statement: Group papers based on their review scores and citation counts.

```
[19] papers_stats = papers[['title', 'review_score', 'citations']]  
print(papers_stats)
```

➞

	title	review_score	citations
0	AI Research Paper 1	5	120
1	Deep Learning Overview	4	450
2	Data Science Trends	5	340
3	Quantum Computing Basics	3	600

20. Most Reviewed Papers per Year Problem Statement: Determine the papers that have been reviewed the most in each year.

```
[20] papers_per_year = papers.groupby('year')['paperId'].nunique()  
most_reviewed_papers_per_year = papers_per_year.sort_values(ascending=False).head(5)  
print("Most Reviewed Papers per Year:")  
print(most_reviewed_papers_per_year)
```

➞ Most Reviewed Papers per Year:
year
2021 2
2019 1
2020 1
Name: paperId, dtype: int64