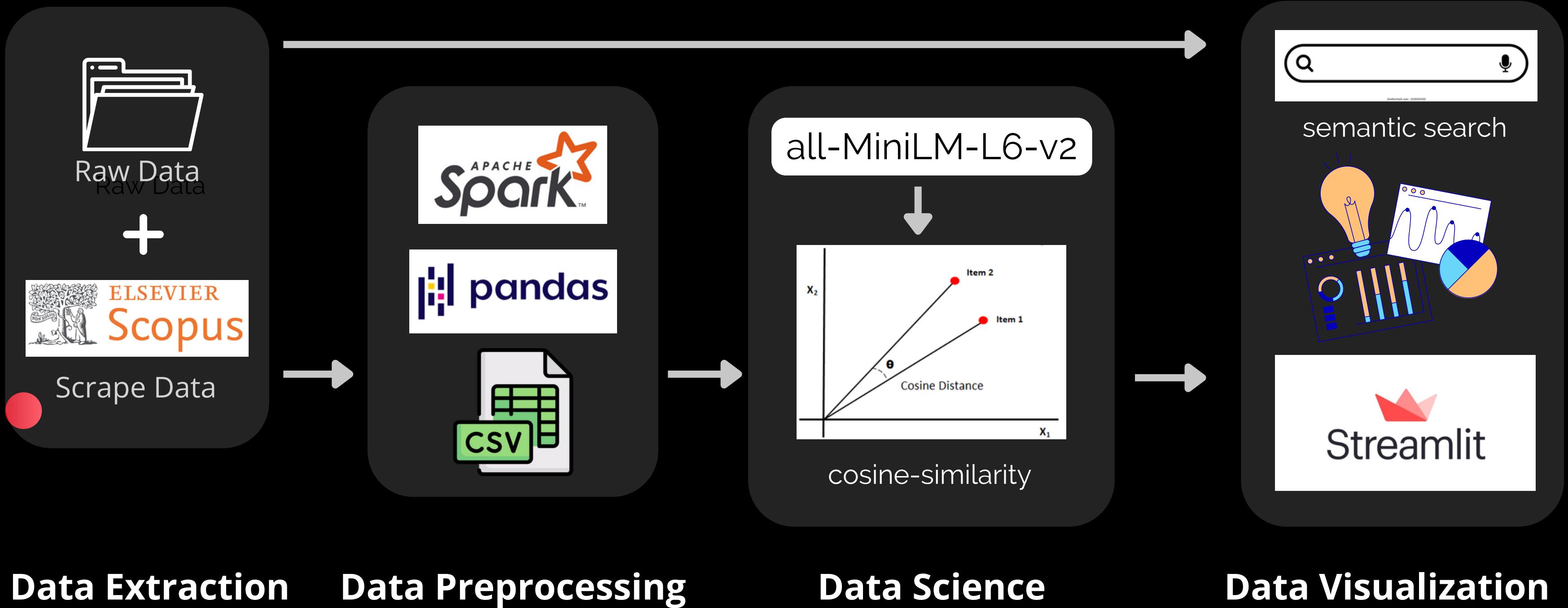


# DATA SCIENCE PROJECT

**Data Science for Scopus Dataset**

Ittikorn Pornchaipimolphant 6633293821  
Thaniya Kitworakiat 6633108421  
Pattanakarn Cheerasemanont 6633163921  
Pittaya Khunpitak 6633173121

# DIAGRAM



# DATA

- language
- title
- abstract
- authors
- keywords
- published\_date
- subject\_code
- citedby\_count
- publisher

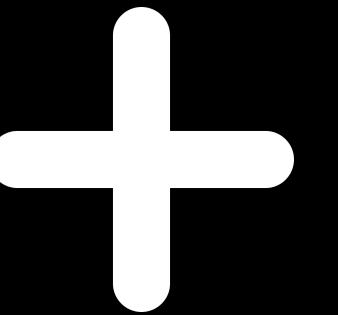


```
        "ce:seq": "6",
        "ce:initials": "T.",
        "@_fa": "true",
        "@type": "auth",
        "ce:surname": "Thilavech",
        "@auid": "56077336400",
        "ce:indexed-name": "Thilavech T."
    }]
},
"citation-title": "Dietary anthocyanins inhibit in",
"abstracts": "© 2022 Elsevier B.V. Insulin fibril fo",
"correspondence": {
    "affiliation": {
        "country": "Thailand",
        "postal-code": "10330",
        "@country": "tha",
        "city": "Bangkok",
        "organization": [
            {"$": "Phytochemical and Functi",
            {"$": "Department of Nutritio",
            {"$": "Faculty of Allied Healt",
            {"$": "Chulalongkorn University"
        ]
    }
}
```

# DATA



Raw data  
(2018 - 2023)



ELSEVIER  
**Scopus**

Scrape Data from Scopus API  
(2011 - 2017)

# DATA PREPROCESSING

Web Scraping

Understanding Data

Clean Data

Query

See details

# WEB SCRAPING

Next Page

```
File Selection View Go Run Terminal Help < > sparkjupyter M apipython ● { } searchresponse_2011.json  
PROJECT D E U ⚡  
data ● 2011 ● 2012  
( ) 201200001.json U  
( ) 201200002.json U  
( ) 201200003.json U  
( ) 201200004.json U  
( ) 201200005.json U  
( ) 201200006.json U  
( ) 201200007.json U  
( ) 201200008.json U  
( ) 201200009.json U  
( ) 201200010.json U  
( ) 201200011.json U  
( ) 201200012.json U  
( ) 201200013.json U  
( ) 201200014.json U  
( ) 201200015.json U  
( ) 201200016.json U  
( ) 201200017.json U  
( ) 201200018.json U  
( ) 201200019.json U  
( ) 201200020.json U  
( ) 201200021.json U  
( ) 201200022.json U  
( ) 201200023.json U  
( ) 201200024.json U  
( ) 201200025.json U  
( ) 201200026.json U  
( ) 201200027.json U  
...  
JIN  
LINE  
LINE  
● apipython > ⚡ import json  
Generate + Code + Markdown | ▶ Run All ⚡ Restart  
( ) 3m 19.3s  
... Retrieved data for year 2011: 200 entries  
Retrieved 200 scopus_id for year 2011: ['84455161  
Retrieved data for data/2011/2011000001.json  
Retrieved data for data/2011/2011000002.json  
Retrieved data for data/2011/2011000003.json  
Retrieved data for data/2011/2011000004.json  
Retrieved data for data/2011/2011000005.json  
Retrieved data for data/2011/2011000006.json  
Retrieved data for data/2011/2011000007.json  
Retrieved data for data/2011/2011000008.json  
Retrieved data for data/2011/2011000009.json  
Retrieved data for data/2011/2011000010.json  
Retrieved data for data/2011/2011000011.json  
Retrieved data for data/2011/2011000012.json  
Retrieved data for data/2011/2011000013.json  
Retrieved data for data/2011/2011000014.json  
Retrieved data for data/2011/2011000015.json  
Retrieved data for data/2011/2011000016.json  
Retrieved data for data/2011/2011000017.json  
Retrieved data for data/2011/2011000018.json  
Retrieved data for data/2011/2011000019.json  
Retrieved data for data/2011/2011000020.json  
Retrieved data for data/2011/2011000021.json  
Retrieved data for data/2011/2011000022.json  
Retrieved data for data/2011/2011000023.json  
...  
Retrieved data for data/2012/2012000070.json  
Retrieved data for data/2012/2012000071.json  
Retrieved data for data/2012/2012000072.json  
Retrieved data for data/2012/2012000073.json  
Output is truncated. View as a scrollable element or open in a
```



```
for year in range(2011, 2018):
    url = f"https://api.elsevier.com/content/search/scopus"
    params = {
        "apiKey": "YOUR_API_KEY",
        "insttoken": "YOUR_INST_TOKEN",
        "httpAccept": "application/json",
        "query": f"PUBYEAR = {year} AND AFFILORG('Chulalongkorn University')",
        "count": 200
    }

    response = requests.get(url, params=params)

    if response.status_code == 200:
        try:
            data = response.json()
            print(f"Retrieved data for year {year}: {len(data.get('search-results', {}).get('documentIndexes'))}")
            documentIndexes = []
            with open(f'searchresponse_{year}.json', 'w') as f:
```

# UNDERSTANDING DATA

```
|-- abstracts-retrieval-response: struct (nullable = true)
|  |-- affiliation: string (nullable = true)
|  |-- authkeywords: struct (nullable = true)
|    |-- author-keyword: string (nullable = true)
|  |-- authors: struct (nullable = true)
|    |-- author: array (nullable = true)
|      |-- element: struct (containsNull = true)
|        |-- @_fa: string (nullable = true)
|        |-- @auid: string (nullable = true)
|        |-- @seq: string (nullable = true)
|        |-- affiliation: string (nullable = true)
|        |-- author-url: string (nullable = true)
|        |-- ce:degrees: string (nullable = true)
|        |-- ce:given-name: string (nullable = true)
|        |-- ce:indexed-name: string (nullable = true)
|        |-- ce:initials: string (nullable = true)
|        |-- ce:suffix: string (nullable = true)
|        |-- ce:surname: string (nullable = true)
|        |-- preferred-name: struct (nullable = true)
|          |-- ce:given-name: string (nullable = true)
|          |-- ce:indexed-name: string (nullable = true)
|          |-- ce:initials: string (nullable = true)
|          |-- ce:surname: string (nullable = true)
|  |-- coredata: struct (nullable = true)
...
|  |  |  |  |-- @_fa: string (nullable = true)
```

```
-85118722833",
  "ion": "Usage of plastics in the form of personal pro
erDate": "2022-12-30",
  "egationType": "Journal",
  "": "https://api.elsevier.com/content/abstract/scopus\_id/85118722833",
  "": {"author": [
    {
      "given-name": "Selvakumar",
      "middle-name": "[REDACTED]",
      "family-name": "Dharmaraj",
      "indexed-name": "Selvakumar Dharmaraj S."
    }
  ],
  "count": 1,
  "affiliations": "S.",
  "is-primary": true,
  "orcid": {
    "id": "60109767",
    "uri": "https://api.elsevier.com/content/affiliation/60109767"
  }
}
```

**Next Page**

# QUERY DATA •

```
export_DF = paperDF.select(  
    col("abstracts-retrieval-response.coredata.dc:title").alias("title"),  
    col("abstracts-retrieval-response.coredata.dc:description").alias("abstract"),  
    col("abstracts-retrieval-response.coredata.dc:creator.author.ce:indexed-name").alias("authors"),  
    col("abstracts-retrieval-response.coredata.prism:coverDate").alias("published_date"),  
    col("abstracts-retrieval-response.subject-areas.subject-area.@abbrev").alias("subject_code"),  
    col("abstracts-retrieval-response.language.@xml:lang").alias("language"),  
    col("abstracts-retrieval-response.coredata.citedby-count").alias("citedby_count"),  
    col("abstracts-retrieval-response.coredata.dc:publisher").alias("publisher"),  
    col("abstracts-retrieval-response.authkeywords.author-keyword").alias("keywords"))
```

# CLEAN DATA

```
# Handle missing values
data_cleaned = data.dropna(subset=['title', 'abstract', 'published_date', 'subject_codes'])

# Drop rows where 'keywords' column is an empty list
data_cleaned = data_cleaned[data_cleaned['keywords'].apply(lambda x: x != [])]

# Fill missing values in the 'cited' column with 0
data_cleaned['citedby_count'] = data_cleaned['citedby_count'].fillna(0)

# Convert the list in 'authors' column to a tuple to make it hashable
data_cleaned['authors'] = data_cleaned['authors'].apply(tuple)

# Remove duplicate rows based on the title and authors combination, assuming they uniquely identify a document.
data_cleaned = data_cleaned.drop_duplicates(subset=['title', 'authors'])
```

```
def clean_subject_codes(subject_codes):
    # Remove duplicates and join back as a comma-separated string.
    unique_codes = ", ".join(sorted(set(subject_codes)))
    return unique_codes

# Apply the cleaning function to the 'subject_codes' column
data_cleaned['subject_codes'] = data_cleaned['subject_codes'].apply(clean_subject_codes)
```



# **DATA SCIENCE**

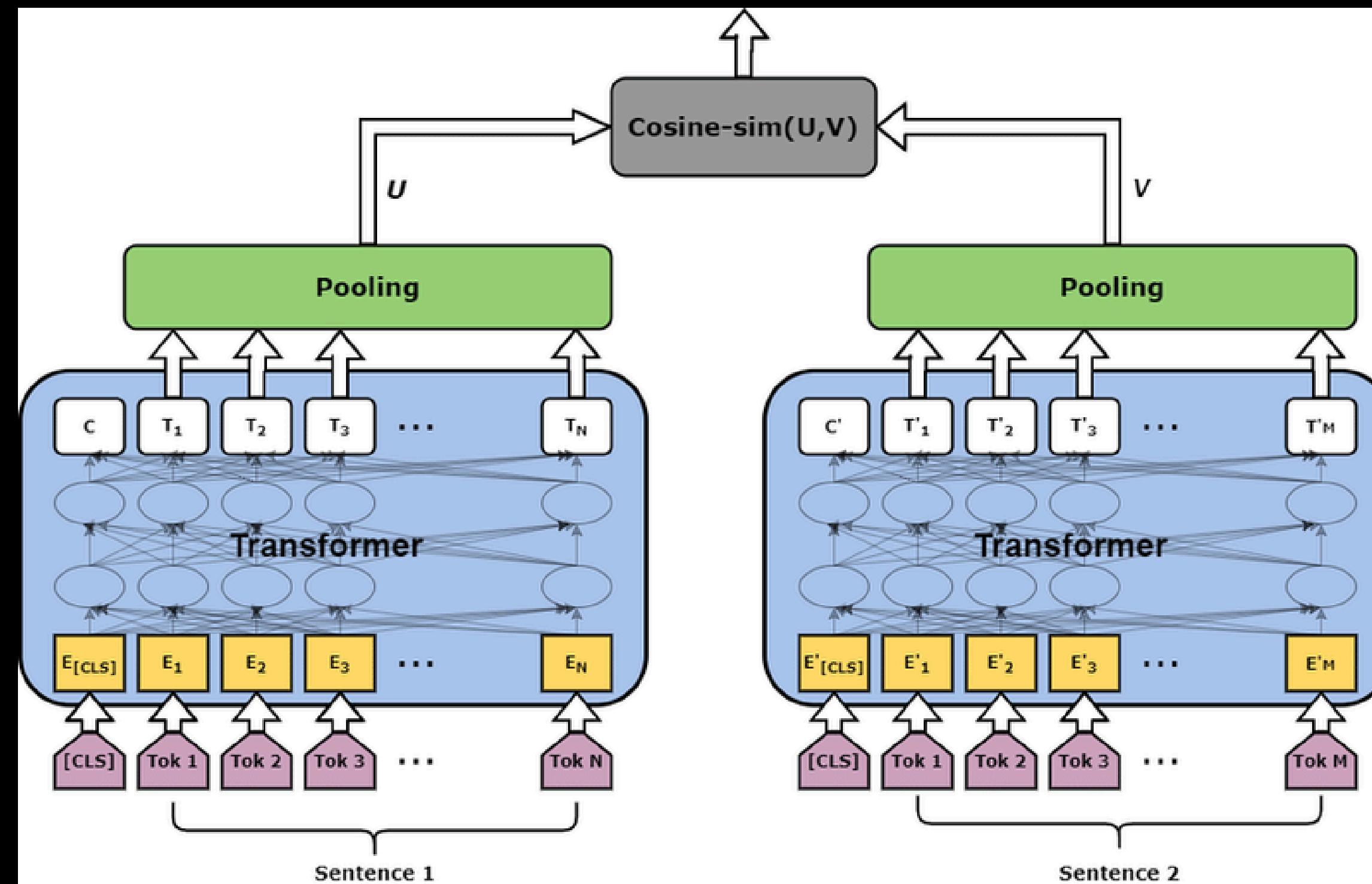
# **AI/ML**



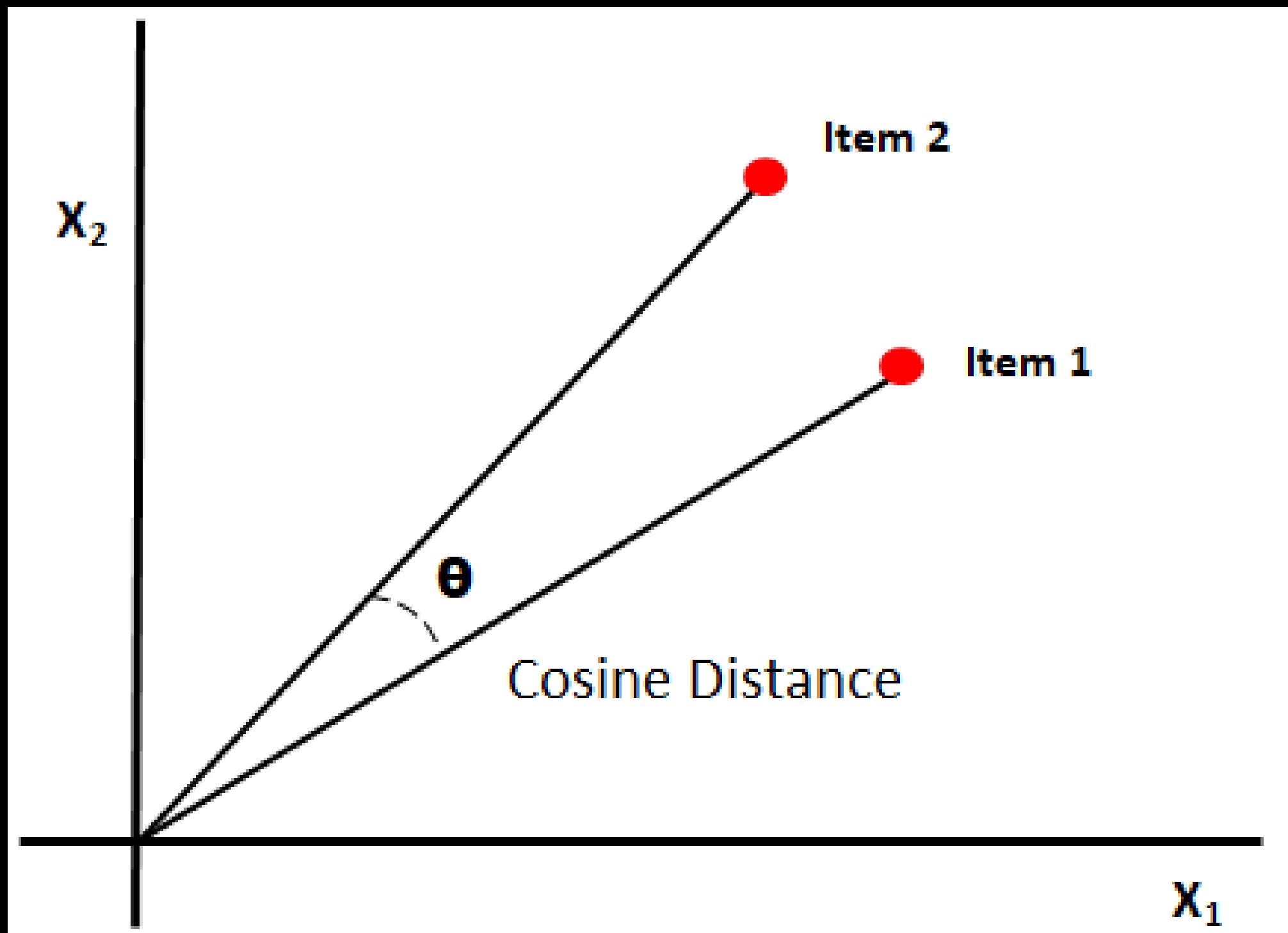
# SEMANTIC SEARCH



# SENTENCE-TRANSFORMERS

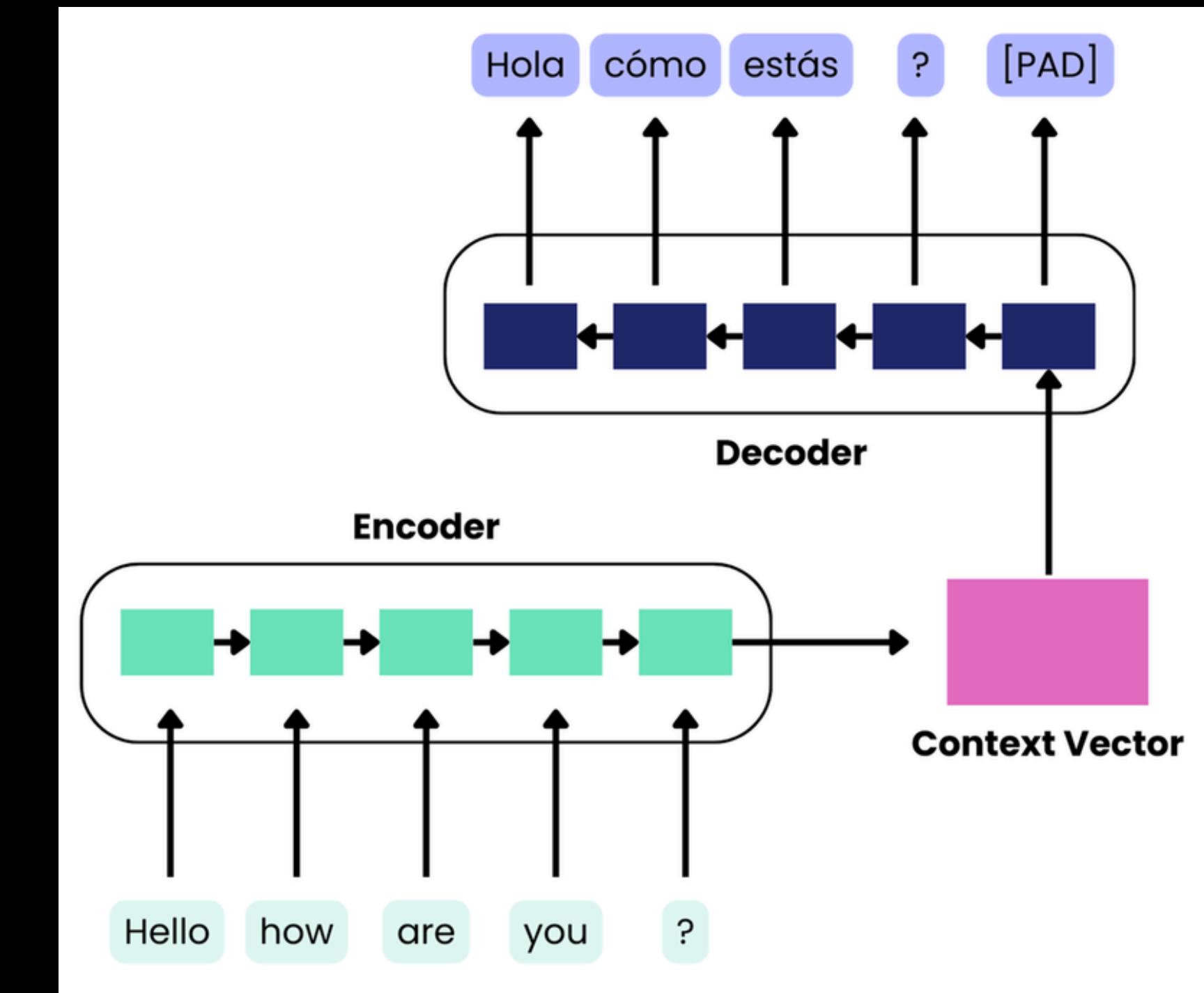


# COSINE-SIMILARITY



# MODEL

- all-MiniLM-L6-v2



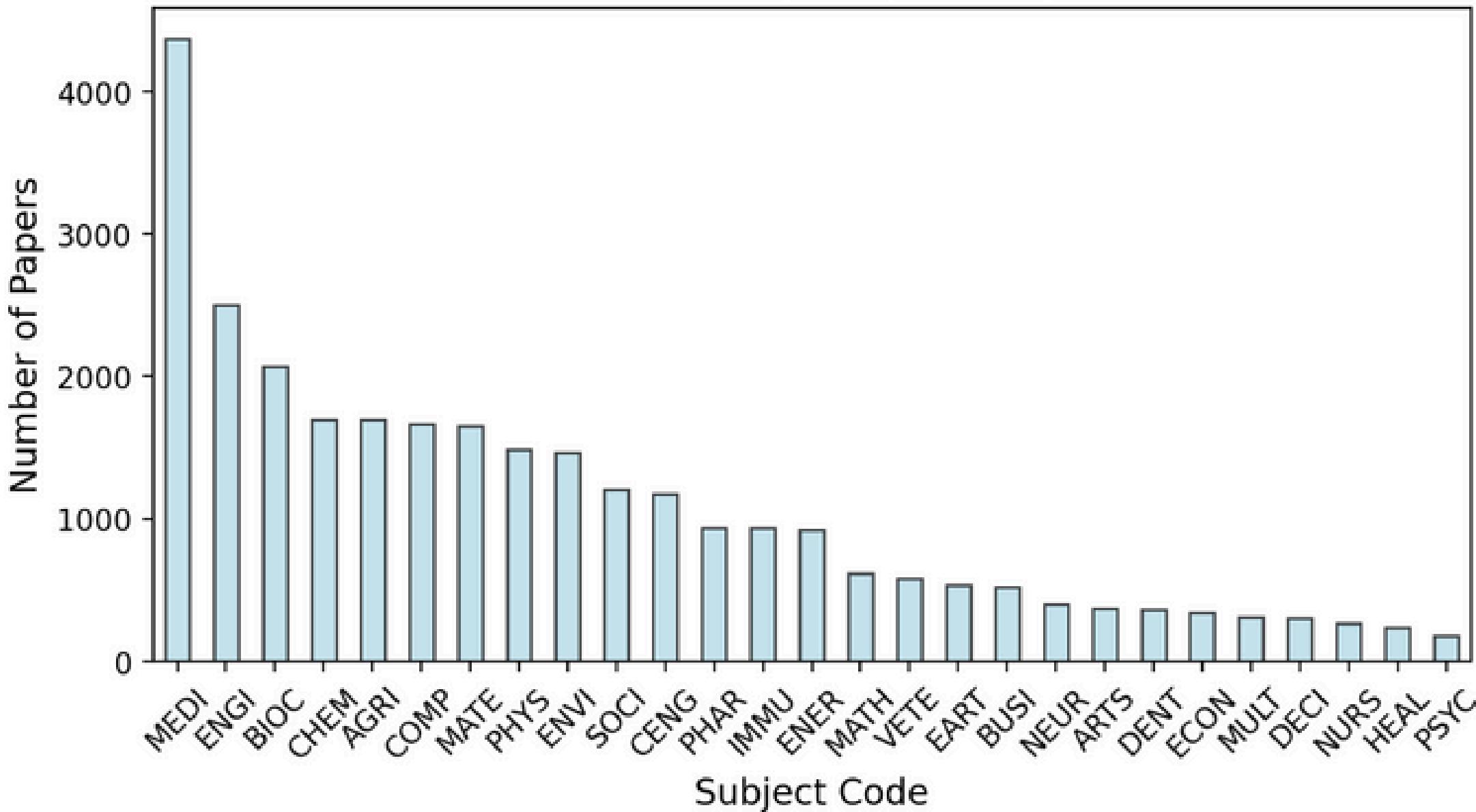
# DATA VISUALIZATION



# DATA OVERVIEW

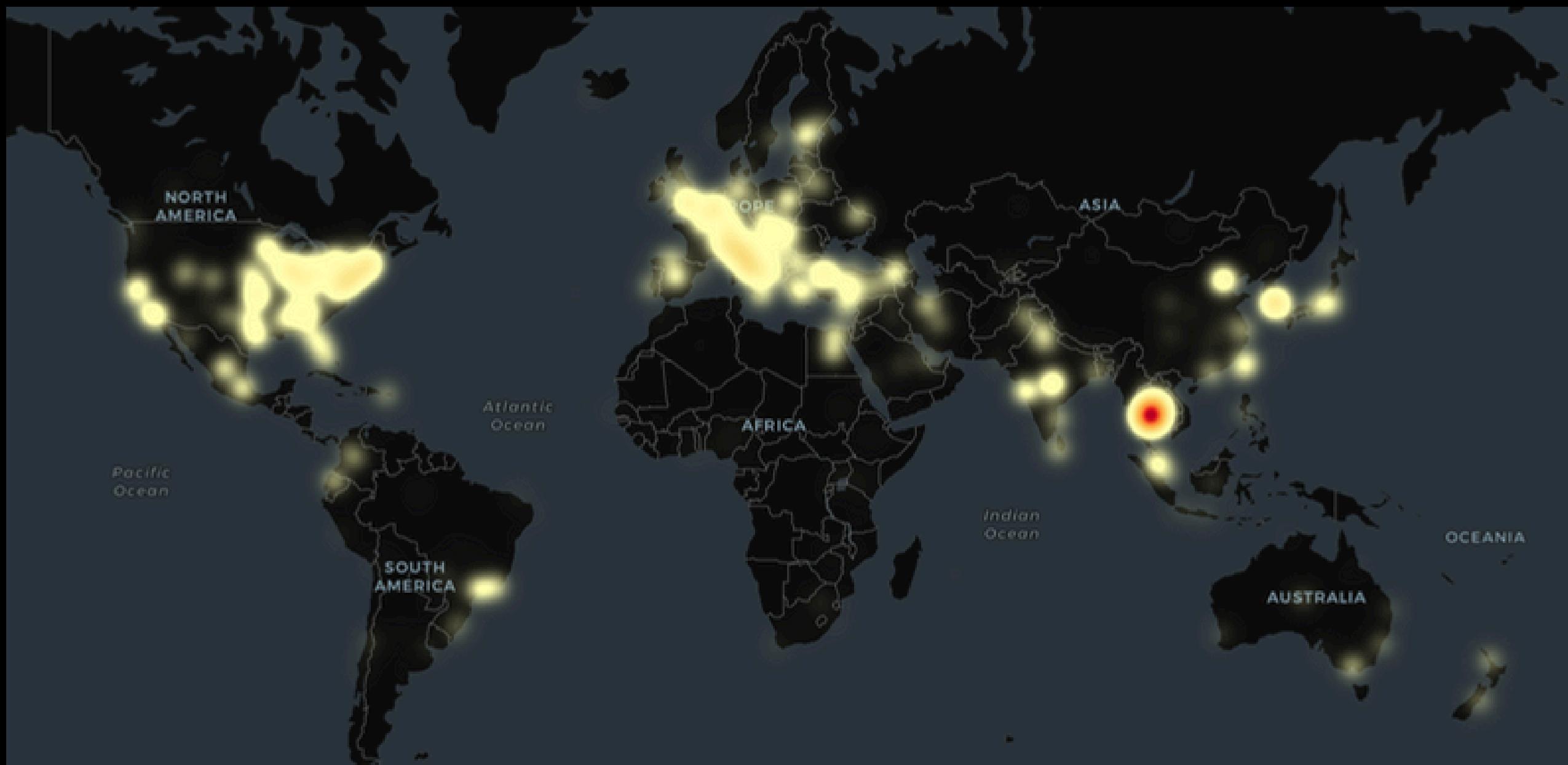
Visualization: Papers Per Subject ↗

Number of Papers per Subject



# DATA OVERVIEW

- Heatmap of location of authors



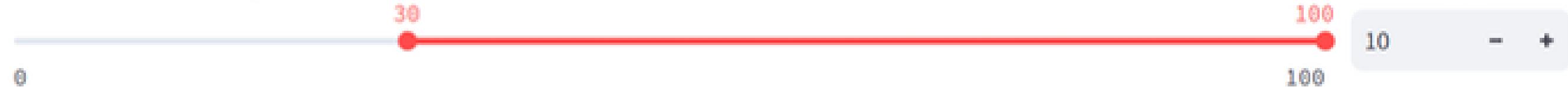
# SEARCH...

## Search Related Papers

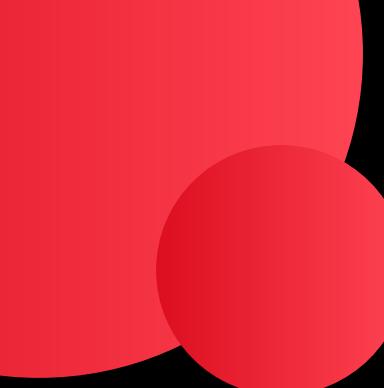
Enter sentence or word:

Search

Select Similarity Percentage:



# DEMO



Home

About

Contact

**Ensuring Data Accuracy and Reliability**

# **IMPORTANCE OF DATA CLEANING**

*Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed at ipsum  
vitae lacus lobortis lacinia. Donec tristique arcu massa, at pharetra  
tortor feugiat non.*

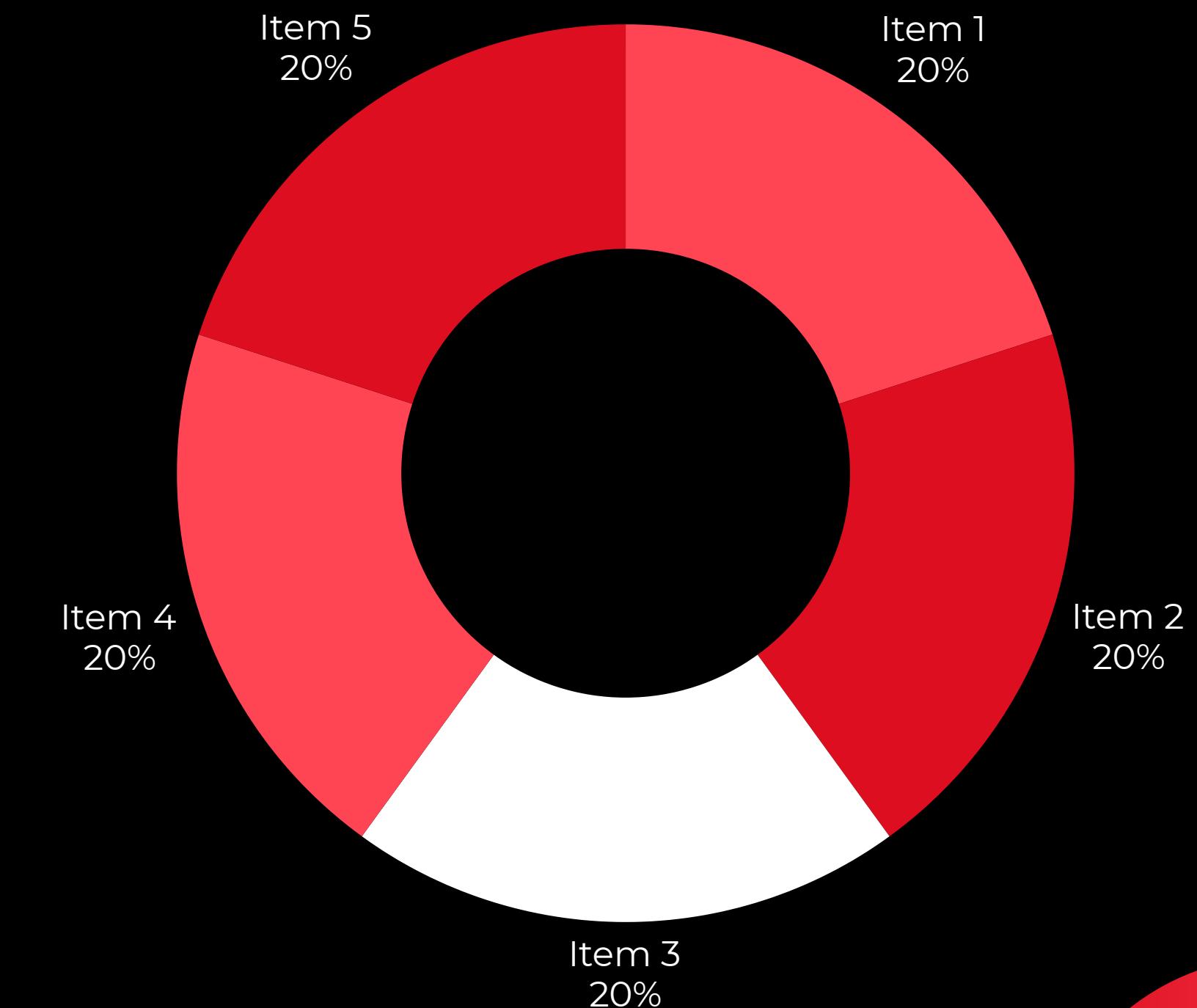
**Read More**



# CHALLENGES IN DATA ANALYSIS

## ● Overcoming Common Obstacles

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed at ipsum vitae lacinia. Donec tristique arcu massa, at pharetra tortor feugiat non.

[Read More](#)

# CONCLUSION

## ● The Future of Data Analysis

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed at ipsum vitae lacus lobortis lacinia. Donec tristique arcu massa, at pharetra tortor feugiat non.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed at ipsum vitae lacus lobortis lacinia. Donec tristique arcu massa, at pharetra tortor feugiat non.

Home

About

Contact

See You Next

# THANK YOU

2024 Data Analysis Presentation

