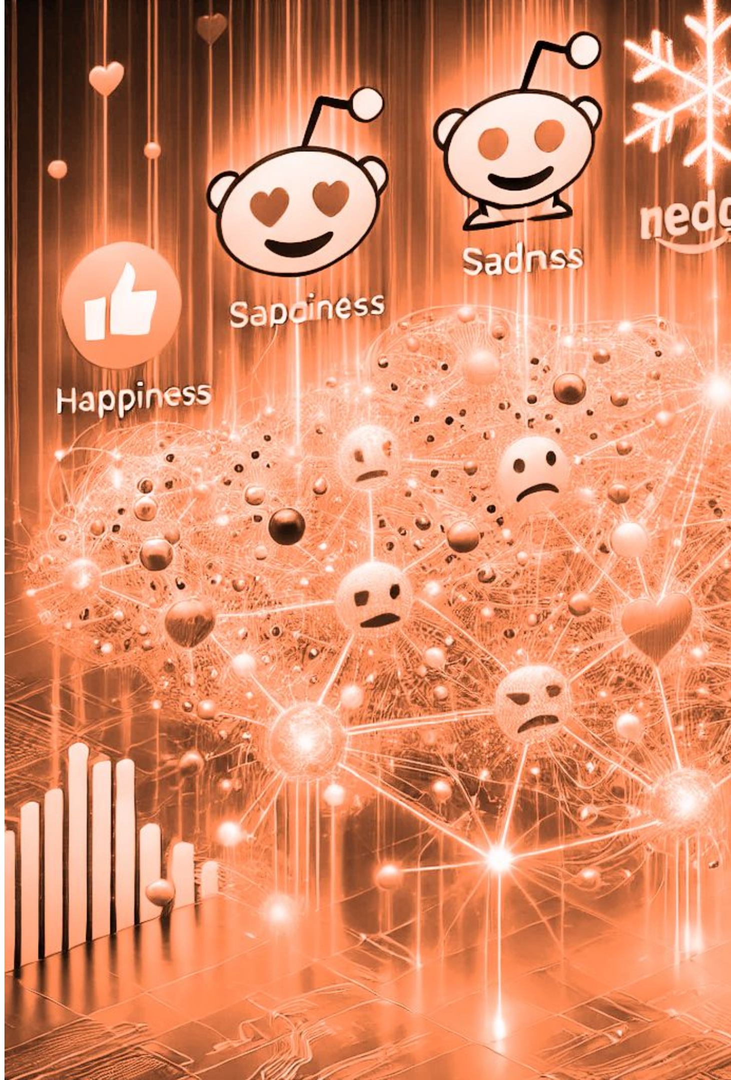
Reddit Sentiment Analysis

December 2, 2024

Project 3 - Ahshaam Abu, Gavin MacPherson, Kyle McDonald, Patrick Mora, Shekhar Das

Executive Summary

Our project aims to analyze and compare different communities. By leveraging machine learning models, NLP models, and Snowflake/AWS we focused on extracting insights from textual data. The primary goal was to use a pre-trained transformer from hugging face to analyze different communities on reddit through sentiment analysis on comments.

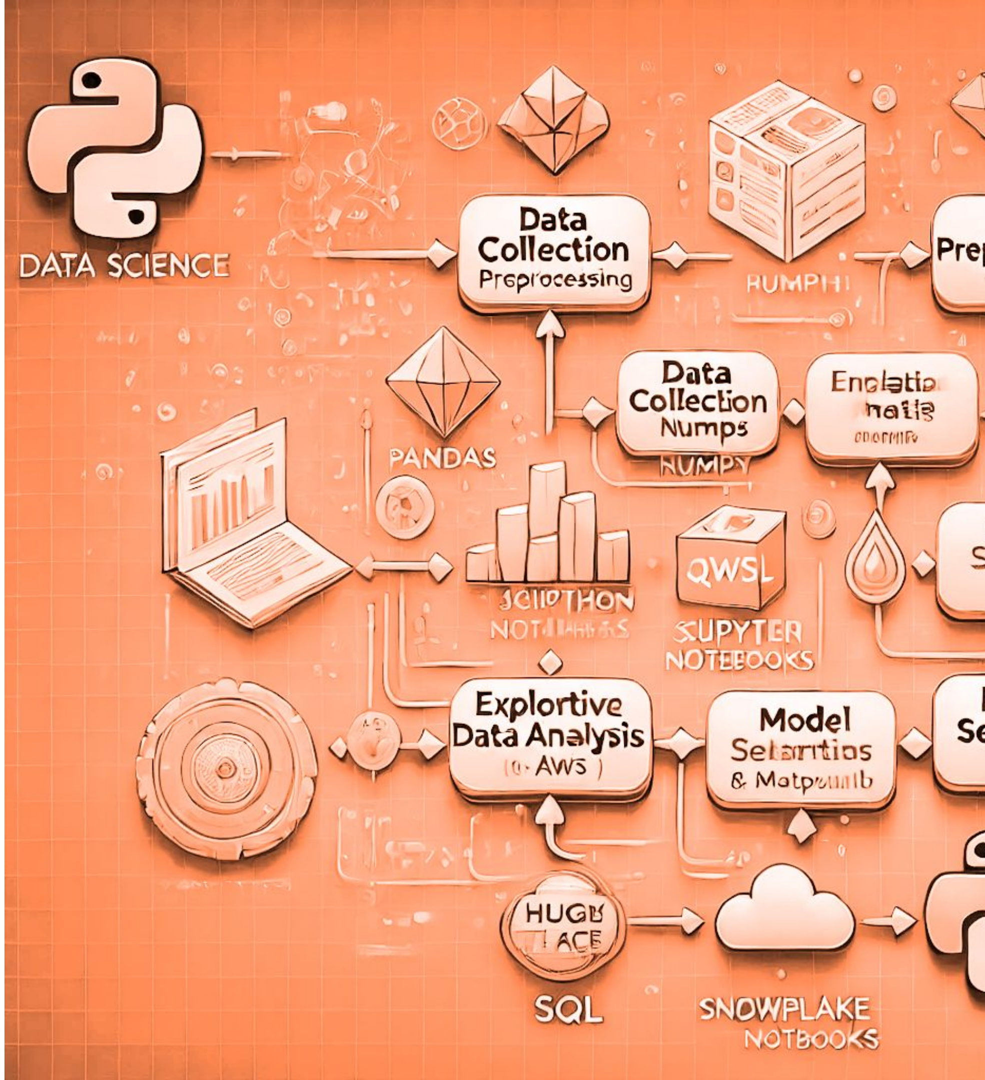


Approach to Achieve Goals

Workflow:

1. Data collection and preprocessing.
2. Exploratory data analysis for insights.
3. Model selection through Hugging Face.
4. Testing, evaluation, and comparison using real-world examples.

Tools Used: Python (Pandas, NumPy, Scikit-learn, Matplotlib), Jupyter Notebooks, Snowflake (SQL, AWS), Hugging Face (DistilBERT, utrobinmv), Reddit API.



Data Collection

- ➔ **Source**
  Data was gathered from communities using Reddit API
- ➔ **Volume**
  Current dataset includes [1200 entries, 5 features]
- ➔ **Rationale**
  These platforms were selected due to their high volume of user-generated content and relevance to our analysis goals.

Data Collection & Data Cleanup

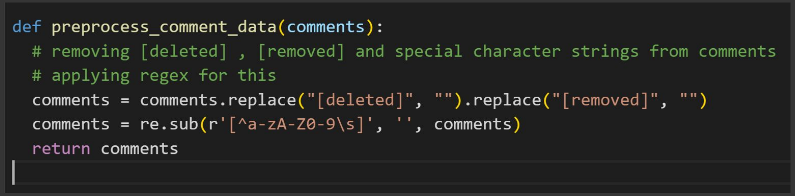
Data Cleanup:

- Handled missing values (deleted/removed comments)
- Removed special characters from comments
- Standardized text formats (e.g., lowercasing, punctuation removal).

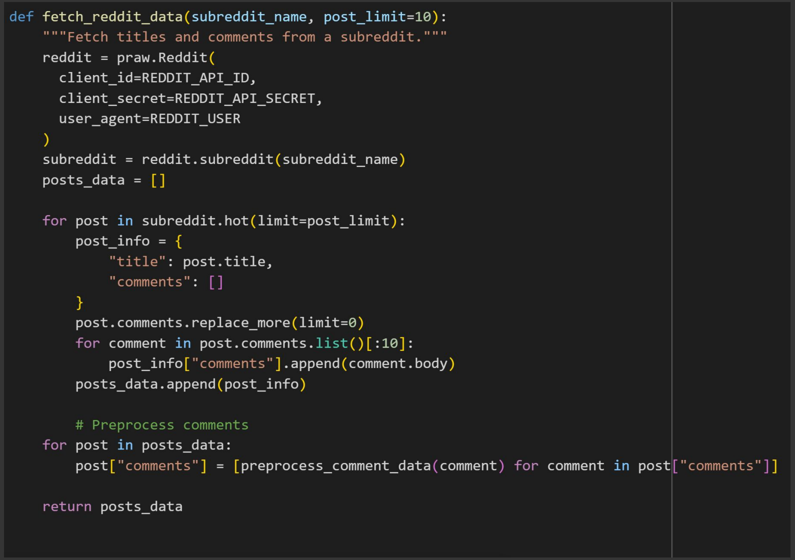
Data Collection:

- Created a function that uses the Reddit API to pull comments from a given number of posts from a given community

Challenges: Removed or deleted comments, and removal of special characters



```python
def preprocess_comment_data(comments):
  # removing [deleted] , [removed] and special character strings from comments
  # applying regex for this
  comments = comments.replace("[deleted]", "").replace("[removed]", "")
  comments = re.sub(r'[^a-zA-Z0-9\s]', '', comments)
  return comments
```



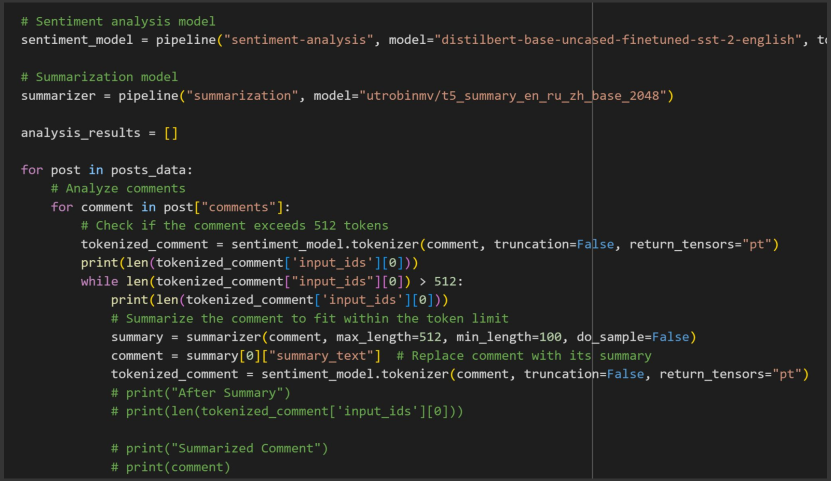
```python
def fetch_reddit_data(subreddit_name, post_limit=10):
    """Fetch titles and comments from a subreddit."""
    reddit = praw.Reddit(
      client_id=REDDIT_API_ID,
      client_secret=REDDIT_API_SECRET,
      user_agent=REDDIT_USER
    )
    subreddit = reddit.subreddit(subreddit_name)
    posts_data = []

    for post in subreddit.hot(limit=post_limit):
        post_info = {
            "title": post.title,
            "comments": []
        }
        post.comments.replace_more(limit=0)
        for comment in post.comments.list()[:10]:
            post_info["comments"].append(comment.body)
        posts_data.append(post_info)

        # Preprocess comments
    for post in posts_data:
        post["comments"] = [preprocess_comment_data(comment) for comment in post["comments"]]

    return posts_data
```

Sentiment Analysis

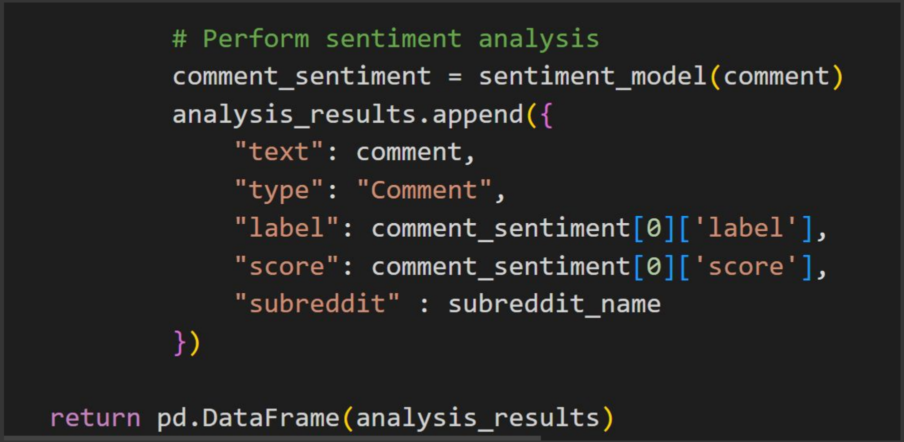
Sentiment Analysis:

- Created a function that returned a dataframe with the predicted sentiments attached to each comment and their features.
- Used distilBERT for sentiment analysis, and utrobin

Challenges: Could only take in 512 words, so if a comment had too many words, we would use the utrobinmv summarization model from Hugging Face.



```python
# Sentiment analysis model
sentiment_model = pipeline("sentiment-analysis", model="distilbert-base-uncased-finetuned-sst-2-english", to

# Summarization model
summarizer = pipeline("summarization", model="utrobinmv/t5_summary_en_ru_zh_base_2048")

analysis_results = []

for post in posts_data:
    # Analyze comments
    for comment in post["comments"]:
        # Check if the comment exceeds 512 tokens
        tokenized_comment = sentiment_model.tokenizer(comment, truncation=False, return_tensors="pt")
        print(len(tokenized_comment['input_ids'][0]))
        while len(tokenized_comment["input_ids"][0]) > 512:
            print(len(tokenized_comment['input_ids'][0]))
            # Summarize the comment to fit within the token limit
            summary = summarizer(comment, max_length=512, min_length=100, do_sample=False)
            comment = summary[0]["summary_text"]  # Replace comment with its summary
            tokenized_comment = sentiment_model.tokenizer(comment, truncation=False, return_tensors="pt")
            # print("After Summary")
            # print(len(tokenized_comment['input_ids'][0]))

            # print("Summarized Comment")
            # print(comment)
```



```python
        # Perform sentiment analysis
        comment_sentiment = sentiment_model(comment)
        analysis_results.append({
            "text": comment,
            "type": "Comment",
            "label": comment_sentiment[0]['label'],
            "score": comment_sentiment[0]['score'],
            "subreddit" : subreddit_name
        })

return pd.DataFrame(analysis_results)
```

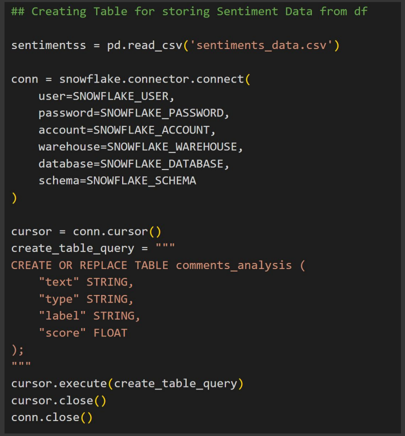
Final Program & Snowflake

Everything Brought Together:

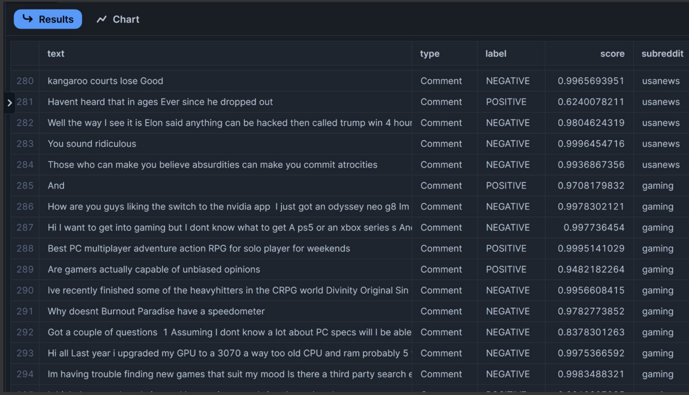
- Created a main function that asks user to input a subreddit topic and specify the number of top posts to analyze, then the number of top posts to be analyzed from that community

Snowflake/Data Storage:

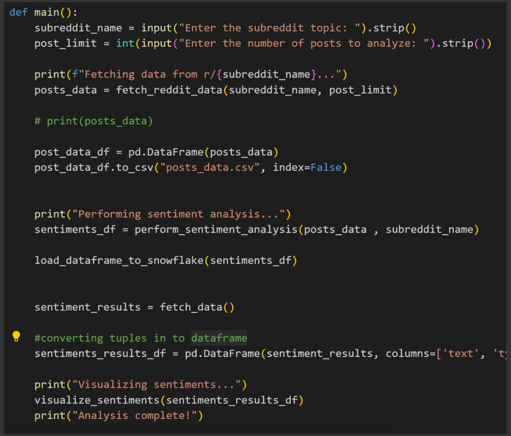
- Used Snowflake to store data in one area every time the program is ran.
- The cloud we used was AWS.



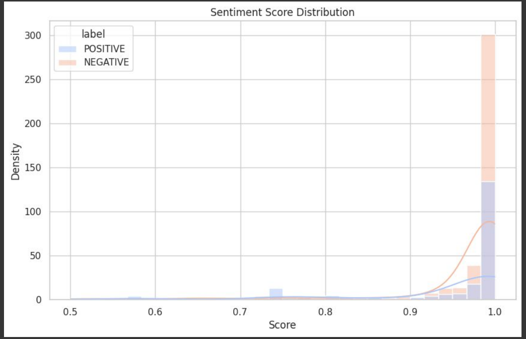
```python
## Creating Table for storing Sentiment Data from df

sentimentss = pd.read_csv('sentiments_data.csv')

conn = snowflake.connector.connect(
    user=SNOWFLAKE_USER,
    password=SNOWFLAKE_PASSWORD,
    account=SNOWFLAKE_ACCOUNT,
    warehouse=SNOWFLAKE_WAREHOUSE,
    database=SNOWFLAKE_DATABASE,
    schema=SNOWFLAKE_SCHEMA
)

cursor = conn.cursor()
create_table_query = """
CREATE OR REPLACE TABLE comments_analysis (
    "text" STRING,
    "type" STRING,
    "label" STRING,
    "score" FLOAT
);
"""
cursor.execute(create_table_query)
cursor.close()
conn.close()
```



	text	type	label	score	subreddit
280	kangaroo courts lose Good	Comment	NEGATIVE	0.9965693951	usanews
281	Havent heard that in ages Ever since he dropped out	Comment	POSITIVE	0.6240078211	usanews
282	Well the way I see it is Elon said anything can be hacked then called trump win 4 hour	Comment	NEGATIVE	0.9804624319	usanews
283	You sound ridiculous	Comment	NEGATIVE	0.9996454716	usanews
284	Those who can make you believe absurdities can make you commit atrocities	Comment	NEGATIVE	0.9936867356	usanews
285	And	Comment	POSITIVE	0.9708179832	gaming
286	How are you guys liking the switch to the nvidia app I just got an odyssey neo g8 Im	Comment	NEGATIVE	0.9978302121	gaming
287	Hi I want to get into gaming but I dont know what to get A ps5 or an xbox series s An	Comment	NEGATIVE	0.997736454	gaming
288	Best PC multiplayer adventure action RPG for solo player for weekends	Comment	POSITIVE	0.9995141029	gaming
289	Are gamers actually capable of unbiased opinions	Comment	POSITIVE	0.9482182264	gaming
290	Ive recently finished some of the heavyhitters in the CRPG world Divinity Original Sin	Comment	NEGATIVE	0.9956608415	gaming
291	Why doesnt Burnout Paradise have a speedometer	Comment	NEGATIVE	0.9782773852	gaming
292	Got a couple of questions 1 Assuming I dont know a lot about PC specs will I be able	Comment	NEGATIVE	0.8378301263	gaming
293	Hi all Last year i upgraded my GPU to a 3070 a way too old CPU and ram probably 5	Comment	NEGATIVE	0.9975366592	gaming
294	Im having trouble finding new games that suit my mood Is there a third party search e	Comment	NEGATIVE	0.9983488321	gaming



```python
def main():
    subreddit_name = input("Enter the subreddit topic: ").strip()
    post_limit = int(input("Enter the number of posts to analyze: ").strip())

    print(f"Fetching data from r/{subreddit_name}...")
    posts_data = fetch_reddit_data(subreddit_name, post_limit)

    # print(posts_data)

    post_data_df = pd.DataFrame(posts_data)
    post_data_df.to_csv("posts_data.csv", index=False)


    print("Performing sentiment analysis...")
    sentiments_df = perform_sentiment_analysis(posts_data , subreddit_name)

    load_dataframe_to_snowflake(sentiments_df)


    sentiment_results = fetch_data()

    #converting tuples in to dataframe
    sentiments_results_df = pd.DataFrame(sentiment_results, columns=['text', 'ty
    
    print("Visualizing sentiments...")
    visualize_sentiments(sentiments_results_df)
    print("Analysis complete!")
```

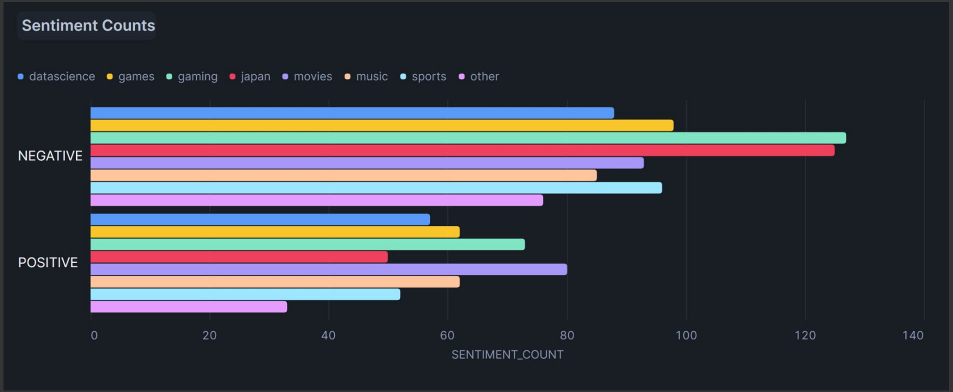


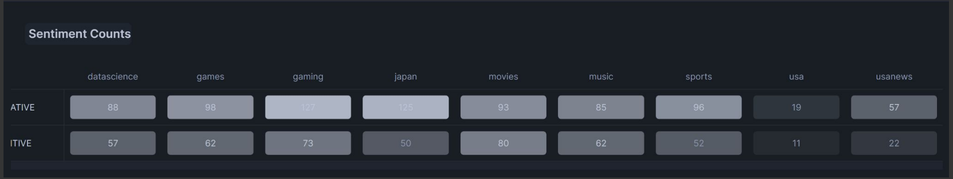
Findings

- There always seemed to be proportionally more negative comments than positive in each community.
    a. People may be more inclined to interact or comment if they share opposite opinions.

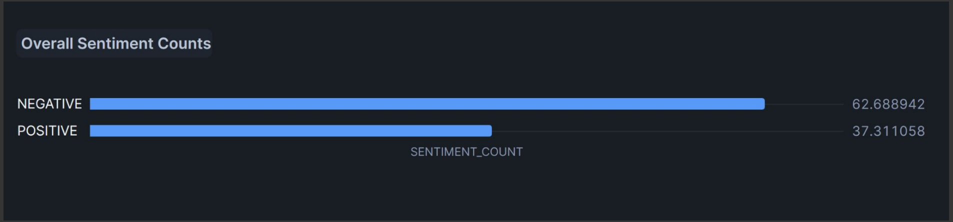
Challenges:

- Wanted to attach dates to comments and analyze the spike in comments based on events but would have had to go back and change lists to dictionaries, which we didn't have enough time to do.
- Model didn't take context into account




Sentiment Counts

	datascience	games	gaming	japan	movies	music	sports	usa	usanews
ATIVE	88	98	127	125	93	85	96	19	57
ITIVE	57	62	73	50	80	62	52	11	22



Future Development

Unanswered Questions:

- How can we improve sentiment analysis accuracy further? (95.8%, but most likely lower)
- What additional data sources could enhance results?

Next Steps:

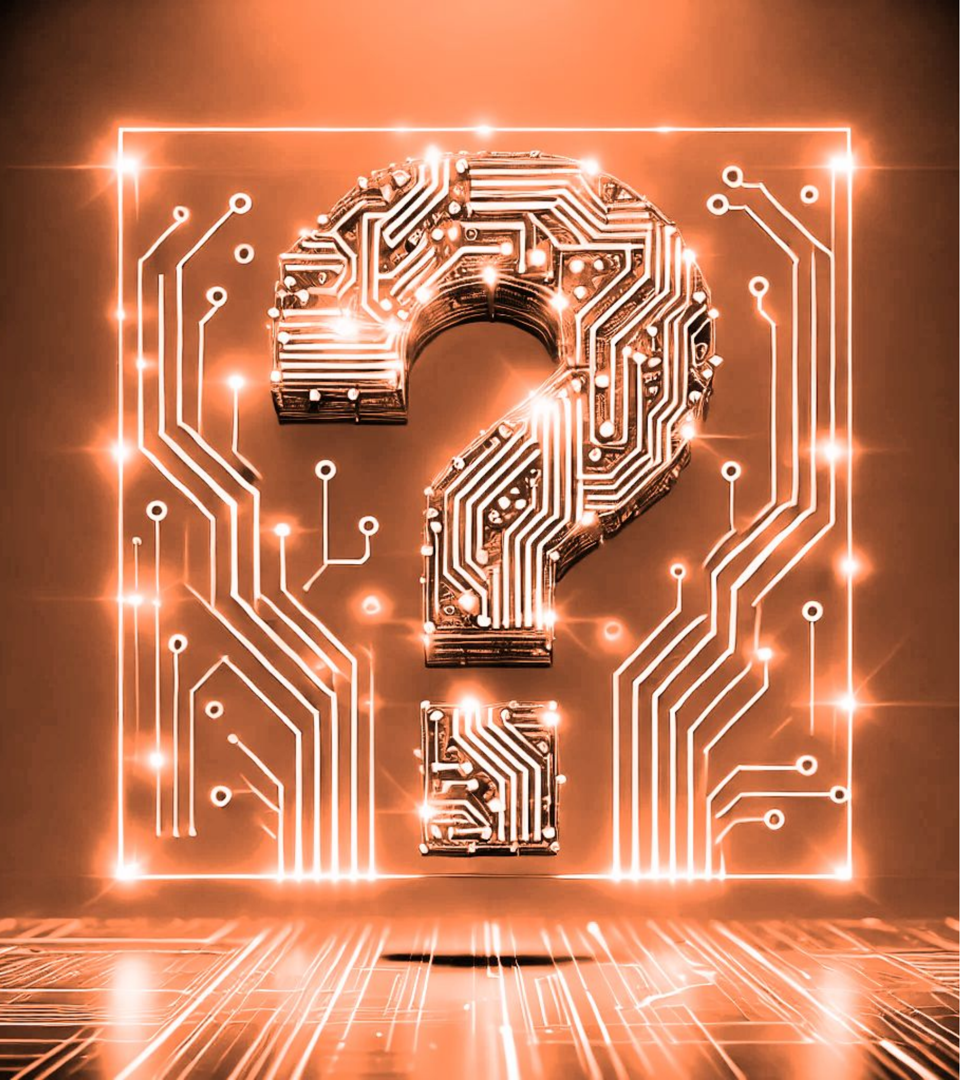
- Extend analysis to more features such as dates.
- Extend analysis to other platforms or languages.
- Use techniques to take in and incorporate context into sentiment analysis.
- Learn more of what Snowflake and SQL have to offer

Acknowledgments

Thank you to our instructors for guidance.

Acknowledgment to our team members for collaboration.

Tools and libraries that supported the project: Python, Jupyter, Scikit-learn, etc.



Questions & Answers

We are happy to answer any questions about our project!