Project Report

Group 6: Pruthvi Billa, Afia Simeen, Zizheng Zhang, Thanmai Reddy

Introduction:

The dataset selected for this project is derived from Stack Exchange, a network of question-and-answer (Q&A) websites covering diverse topics. The primary motivation behind choosing this dataset lies in the potential to develop a predictive model for accurately classifying posts into their respective topics. The overarching business question driving this project is whether we can predict the topic of a question based on its content. The practical application of such a predictive model is to enable Stack Exchange to monitor user posts effectively and flag those that are irrelevant to the topic. This, in turn, aims to enhance the content quality of the website.

Additionally, Stack Exchange's reputation award process and self-moderating nature make it a valuable source for understanding user interactions and content dynamics. For example, this dataset provides information on user behavior, post frequencies, and contributions across different topics. Through exploratory analysis, we intend to investigate user behavior patterns, including post frequencies and correlations between user behavior and topic preferences. The implementation of a predictive model for topic classification aligns with the goal of improving user experience by facilitating targeted content monitoring and fostering business growth through more effective advertising. The results of this study have the potential to positively impact both the user experience and the business outcomes of Stack Exchange.

Description of the Dataset:

The dataset is substantial, with a size of 3.05GB, containing approximately 2 million rows. The data encompasses a diverse range of topics, including AI, gaming, history, movies, music, and software engineering. It is structured with diverse columns of various types. For example, in addition to Integer columns to represent numerical values, it also has DateTime columns to capture temporal aspects, and Categorical Variables like enums to categorize entries into specific classes.

Tools used in this project:

1. AWS S3, EC2

2. HDFS, Pig

3. PySpark, Colab, PyTorch

We utilized Amazon Web Services: S3 for data storage and file transfer, and EC2 to execute several Python scripts. For big data applications, we employed Hadoop HDFS for data storage and Pig in MapReduce mode for data processing. Data processing with Pig commands involved loading data from HDFS. Descriptive analytics were performed using PySpark, and for predictive modeling, we used Colab and Torch.

Cleaning and Transformation Process:

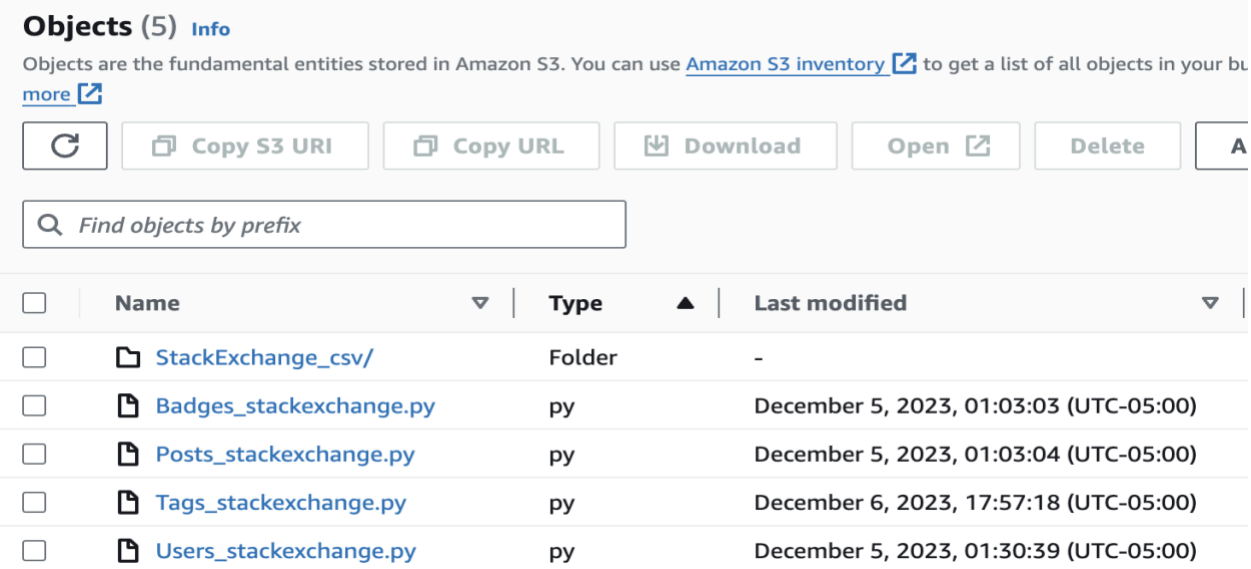
1. File conversions from xml to csv: All dataset files were initially in XML format. Using the 'stackexchange-xml-converter' CLI tool, we converted them into CSV files to facilitate easier data processing and analytics.

```
(base) pruthvishyambilla@Pruthvis-MBP-2 stackexchange-xml-converter % find /Users/pruthvish
ackExchange/ai.stackexchange.com -type f -name "*.xml" -exec ./stackexchange-xml-converter
-source-path {} -store-to-dir=/Users/pruthvishyambilla/Desktop/StackExchange_csv/ \;
2023/12/06 16:59:03 Total 1 file(s) to convert
2023/12/06 16:59:03 [Comments] Converting is started
2023/12/06 16:59:03 [Comments] File is converted. 27,350 of 27,350 row(s) has been processe
2023/12/06 16:59:03 Total 1 file(s) to convert
2023/12/06 16:59:03 [Users] Converting is started
2023/12/06 16:59:04 [Users] File is converted. 66,629 of 66,629 row(s) has been processed s
2023/12/06 16:59:04 Total 1 file(s) to convert
2023/12/06 16:59:04 [Votes] Converting is started
2023/12/06 16:59:04 [Votes] File is converted. 88,548 of 88,548 row(s) has been processed s
2023/12/06 16:59:04 Total 1 file(s) to convert
2023/12/06 16:59:04 [Tags] Converting is started
2023/12/06 16:59:04 [Tags] File is converted. 983 of 983 row(s) has been processed successf
2023/12/06 16:59:04 Total 1 file(s) to convert
2023/12/06 16:59:04 [PostHistory] Converting is started
2023/12/06 16:59:05 [PostHistory] File is converted. 100,535 of 100,535 row(s) has been prc
2023/12/06 16:59:05 Total 1 file(s) to convert
2023/12/06 16:59:05 [PostLinks] Converting is started
2023/12/06 16:59:05 [PostLinks] File is converted. 2,356 of 2,356 row(s) has been processed
2023/12/06 16:59:05 Total 1 file(s) to convert
2023/12/06 16:59:05 [Posts] Converting is started
2023/12/06 16:59:06 [Posts] File is converted. 25,296 of 25,296 row(s) has been processed s
2023/12/06 16:59:06 Total 1 file(s) to convert
2023/12/06 16:59:06 [Badges] Converting is started
2023/12/06 16:59:06 [Badges] File is converted. 58,820 of 58,820 row(s) has been processed
(base) pruthvishyambilla@Pruthvis-MBP-2 stackexchange-xml-converter %
```

2. Data Transfer and Storage:

We started by uploading files to AWS S3. Then, we used the AWS S3 sync command to transfer files from S3 to an EC2 instance. Subsequently, all these files were copied to HDFS for further processing.

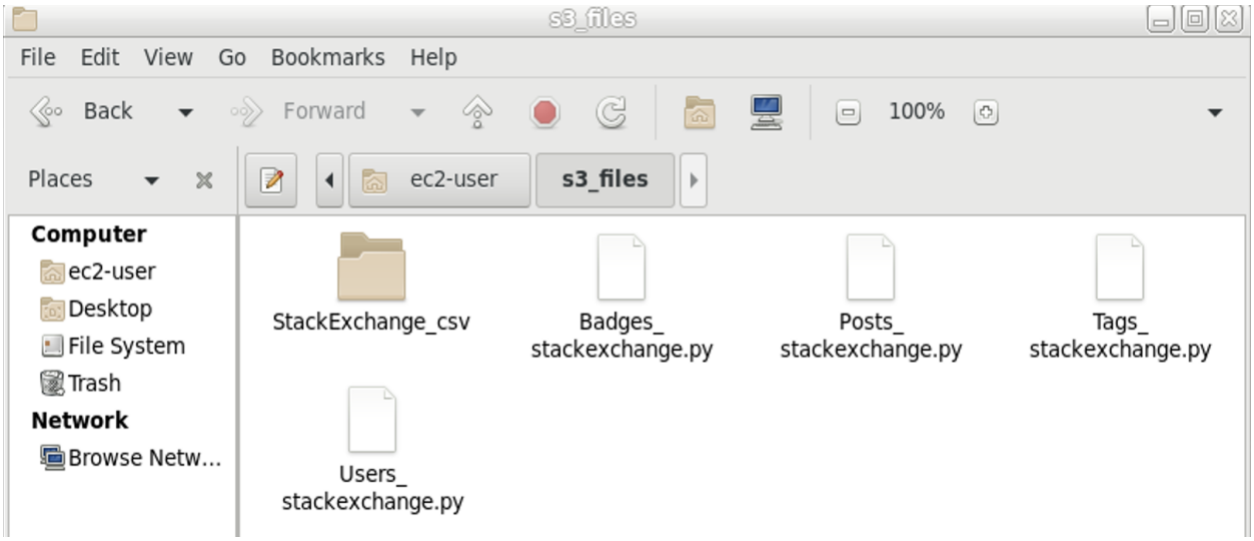
AWS S3:



Command to transfer from S3 to EC2:

```
[ec2-user@ip-172-31-61-143 ~]$ aws s3 sync s3://budt758-billa-330pm ./s3_files
```

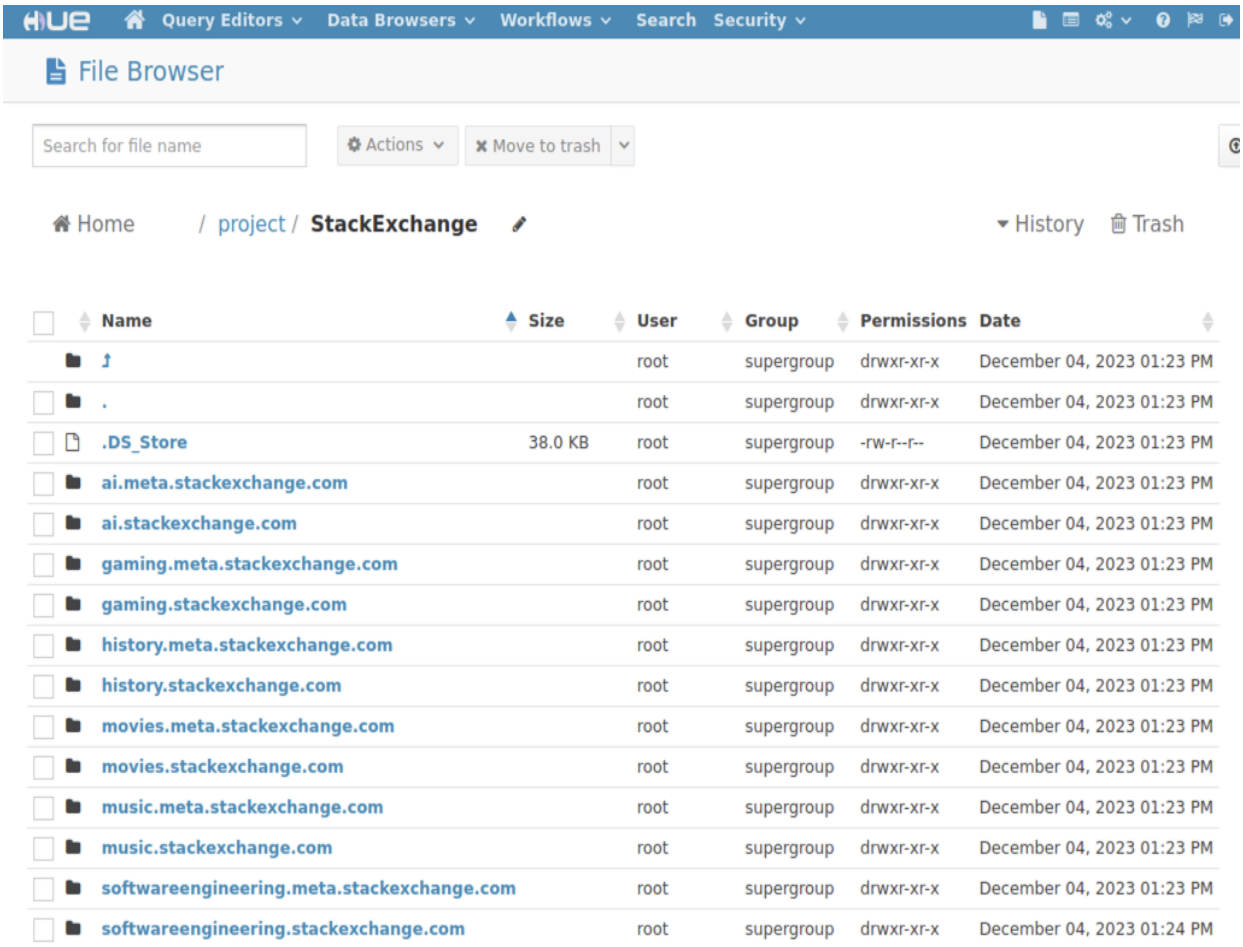
AWS EC2:



Docker:

```
[root@quickstart s3_files]# ls
 StackExchange
[root@quickstart s3_files]# hdfs dfs -put StackExchange/ /project/
 [root@quickstart s3_files]#
```

HDFS:



Name	Size	User	Group	Permissions	Date
..		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
.		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
.DS_Store	38.0 KB	root	supergroup	-rw-r--r--	December 04, 2023 01:23 PM
ai.meta.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
ai.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
gaming.meta.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
gaming.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
history.meta.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
history.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
movies.meta.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
movies.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
music.meta.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
music.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
softwareengineering.meta.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:23 PM
softwareengineering.stackexchange.com		root	supergroup	drwxr-xr-x	December 04, 2023 01:24 PM

3. Pig commands:

Pig, a platform for processing and analyzing large datasets, was utilized to implement the ETL scripts for duplicate removal. Pig simplifies the process of working with Hadoop through a high-level scripting language, making it effective for big data processing tasks. The initial step involved the removal of duplicate rows from the dataset. This was achieved through ETL (Extract, Transform, Load) scripts implemented in Pig on a docker container.

In the process of validating badge distribution analysis across various topics on Stack Exchange, data processing with Apache Pig played a crucial role. Initially, the `badges_etl.pig` script was

employed to eliminate any duplicate entries from each topic's badge dataset. This cleansing ensured the integrity of the subsequent analysis.

File: badges_etl.pig

```
-- Load Badges.csv file using PigStorage and define schema
badges_data = LOAD '/project/StackExchange/ai.stackexchange.com/Badges.csv'
 USING PigStorage(',') AS (Id:int, UserId:int, Class:int,
 Name:chararray, TagBased:int, Date:chararray );

-- Project only Name column
badges = FOREACH badges_data GENERATE Name;

badges_distinct = DISTINCT badges;

STORE badges_distinct INTO '/project/StackExchange/Badge/ai.csv' USING PigStorage(',');
```

Terminal output:

```
HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.7.0  0.12.0-cdh5.7.0 root    2023-12-05 04:38:36     2023-12-05 04:40:43     DISTIN

Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTIme      AvgMapTime      MedianMapTime   MaxRed
job_1701749287690_0011  1       1       8       8       8       8       5       5       5
job_1701749287690_0012  1       1       8       8       8       8       6       6       6
job_1701749287690_0013  1       1       9       9       9       9       5       5       5
job_1701749287690_0014  1       1       9       9       9       9       5       5       5
job_1701749287690_0015  1       1       8       8       8       8       4       4       4
job_1701749287690_0016  1       1       6       6       6       6       5       5       5

Input(s):
Successfully read 93566 records (5158763 bytes) from: "/project/StackExchange/history.stackexc
Successfully read 120944 records (6690392 bytes) from: "/project/StackExchange/music.stackexch
Successfully read 442783 records (25287755 bytes) from: "/project/StackExchange/gaming.stackex
Successfully read 614541 records (34861761 bytes) from: "/project/StackExchange/softwareengine
Successfully read 156478 records (8731946 bytes) from: "/project/StackExchange/movies.stackexc
Successfully read 58821 records (3232443 bytes) from: "/project/StackExchange/ai.stackexchange

Output(s):
Successfully stored 137 records (1489 bytes) in: "/project/StackExchange/Badge/history.csv"
Successfully stored 173 records (1831 bytes) in: "/project/StackExchange/Badge/music.csv"
Successfully stored 186 records (2564 bytes) in: "/project/StackExchange/Badge/gaming.csv"
Successfully stored 203 records (2242 bytes) in: "/project/StackExchange/Badge/softwreengineer
Successfully stored 113 records (1287 bytes) in: "/project/StackExchange/Badge/movies.csv"
Successfully stored 97 records (1122 bytes) in: "/project/StackExchange/Badge/ai.csv"

Counters:
Total records written : 909
Total bytes written : 10535
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1701749287690_0011
job_1701749287690_0012
job_1701749287690_0013
job_1701749287690_0014
job_1701749287690_0015
job_1701749287690_0016
```

Following this, the `common_badges.etl` script was used to identify badges that were common across all topics. The outcome was a refined list of 75 unique records, laying a solid foundation for conducting a comprehensive badge distribution analysis and interpreting the implications of the results.

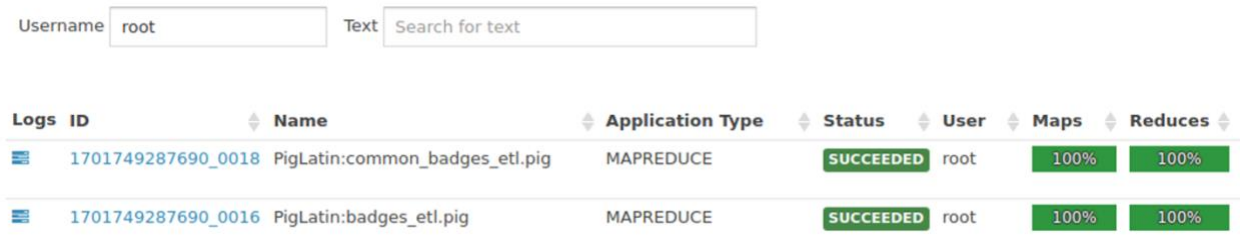
File: common_badges_etl.pig

```
-- Load the badge names from both files
badges_ai = LOAD '/project/StackExchange/Badges/badges_ai.csv' USING PigStorage(',') AS (badge:chararray);
badges_gaming = LOAD '/project/StackExchange/Badges/badges_gaming.csv' USING PigStorage(',') AS (badge:chararray);
badges_history = LOAD '/project/StackExchange/Badges/badges_history.csv' USING PigStorage(',') AS (badge:chararray);
badges_movies = LOAD '/project/StackExchange/Badges/badges_movies.csv' USING PigStorage(',') AS (badge:chararray);
badges_music = LOAD '/project/StackExchange/Badges/badges_music.csv' USING PigStorage(',') AS (badge:chararray);
badges_softwareengineering = LOAD '/project/StackExchange/Badges/badges_softwareengineering.csv'
 USING PigStorage(',') AS (badge:chararray);

-- Remove the 'Name' value from both datasets
badges_ai_filtered = FILTER badges_ai BY badge != 'Name';
badges_gaming_filtered = FILTER badges_gaming BY badge != 'Name';
badges_history_filtered = FILTER badges_history BY badge != 'Name';
badges_movies_filtered = FILTER badges_movies BY badge != 'Name';
badges_music_filtered = FILTER badges_music BY badge != 'Name';
badges_softwareengineering_filtered = FILTER badges_softwareengineering BY badge != 'Name';

-- Find common values
common_badges = JOIN badges_ai_filtered BY badge, badges_gaming_filtered BY badge,
 badges_history_filtered BY badge, badges_movies_filtered BY badge, badges_music_filtered BY badge,
 badges_softwareengineering_filtered BY badge;

-- Extract the common values
common_values = FOREACH common_badges GENERATE badges_ai_filtered::badge AS common_badge;

-- Store the common values in a new file
STORE common_values INTO '/project/StackExchange/Badges/Common_Badges.csv' USING PigStorage(',');
```

Terminal output:

```
HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.7.0  0.12.0-cdh5.7.0 root    2023-12-05 04:59:57     2023-12-05 05:00:39     HASH_JOIN,

Success!

Job Stats (time in seconds):
JobId   Maps    Reduces MaxMapTime      MinMapTIme      AvgMapTime      MedianMapTime   MaxReduceT
job_1701749287690_0018  6       1       16      13      15      16      7       7       7       7
c_filtered,badges_softwareengineering,badges_softwareengineering_filtered,common_badges,common_val

Input(s):
Successfully read 113 records from: "/project/StackExchange/Badges/badges_movies.csv"
Successfully read 137 records from: "/project/StackExchange/Badges/badges_history.csv"
Successfully read 186 records from: "/project/StackExchange/Badges/badges_gaming.csv"
Successfully read 203 records from: "/project/StackExchange/Badges/badges_softwareengineering.csv"
Successfully read 97 records from: "/project/StackExchange/Badges/badges_ai.csv"
Successfully read 173 records from: "/project/StackExchange/Badges/badges_music.csv"

Output(s):
Successfully stored 75 records (811 bytes) in: "/project/StackExchange/Badges/Common_Badges.csv"

Counters:
Total records written : 75
Total bytes written : 811
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1701749287690_0018
```

Hue Browser for the pig jobs:



4. Python:

A Python script incorporating a data handling function was utilized to process cross-topic community user data, effectively managing duplicates and missing values.

```
def data_handling(df):
    # Remove duplicates in 'AccountId' column
    df.drop_duplicates(subset=['AccountId'], inplace=True)

    # Handling missing values
    df.dropna(subset=['AccountId'], inplace=True)
    return df
```

Descriptive Analysis:

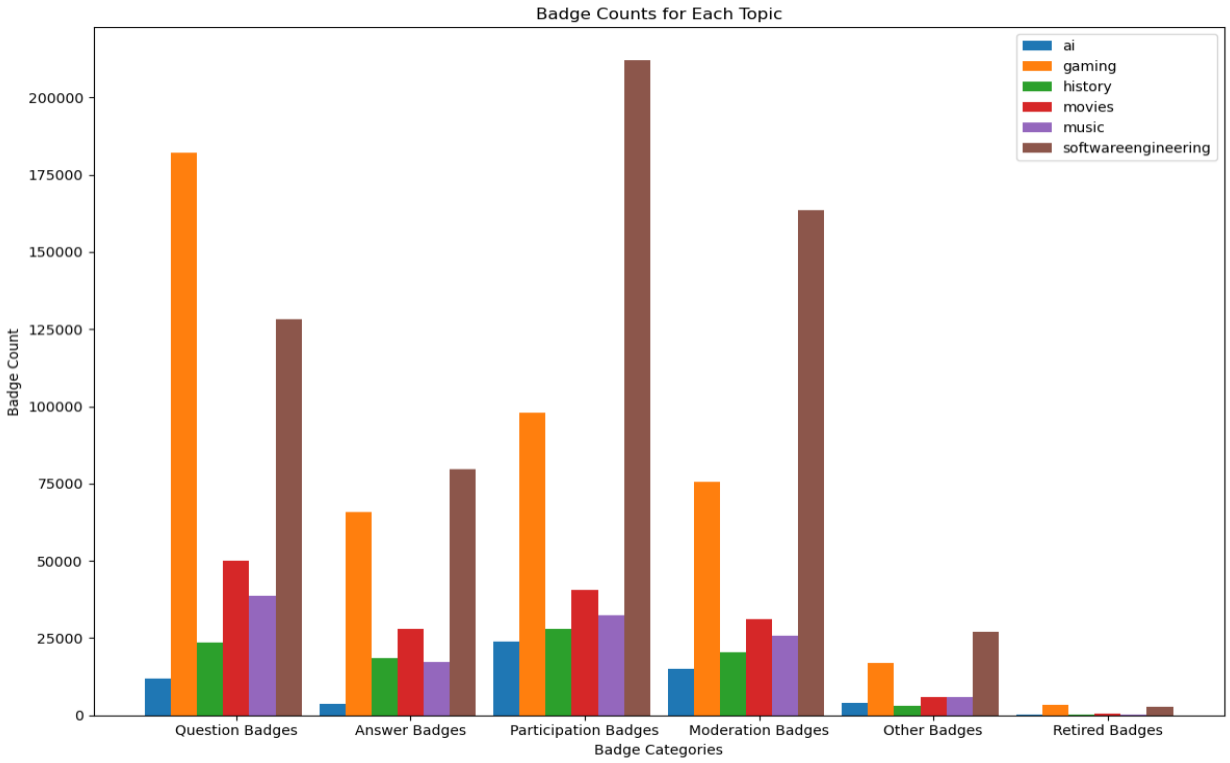
The choice of the Stack Exchange dataset was driven by the anticipation that it could provide insights into several key business questions. The expectation was that the dataset's comprehensive coverage of diverse topics and rich user-contributed content would offer valuable insights into these business questions, aiding in strategic decision-making and platform optimization.

Python script files were executed using python3 in ec2.

Business Questions:

1. **Platform Engagement and Growth**: Understanding how badge distribution influences user participation and expertise recognition.

```
# Set the width of the bars
bar_width = 0.15
bar_positions = np.arange(6)

# Colors for the bars
colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22', '#17becf']

# Initialize a figure and axis
fig, ax = plt.subplots(figsize=(12, 8))

# Iterate through each topic and create a clustered bar chart
for i, topic in enumerate(topics_list):
    badges_df = pd.read_csv(f"/home/ec2-user/s3_files/StackExchange/{topic}.stackexchange.com/Badges.csv")['Name']

    categories_count = {
        "Question Badges": 0,
        "Answer Badges": 0,
        "Participation Badges": 0,
        "Moderation Badges": 0,
        "Other Badges": 0,
        "Retired Badges": 0
    }
    for badge in badges_df:
        category = badge_to_category.get(badge, None)
        if category:
            categories_count[category] += 1

    # Get category counts and labels
    counts = list(categories_count.values())
    categories = list(categories_count.keys())

    # Plotting the clustered bar chart for each topic
    ax.bar(bar_positions + i * bar_width, counts, bar_width, label=topic, color=colors[i % len(colors)])

# Set the labels and title
ax.set_xlabel('Badge Categories')
ax.set_ylabel('Badge Count')
ax.set_title('Badge Counts for Each Topic')
ax.set_xticks(bar_positions + (len(topics_list) / 2) * bar_width)
ax.set_xticklabels(categories)
ax.legend()
plt.tight_layout()
plt.savefig('Badges_topic.png')
plt.show()
```



Analysis: The plot appears to show the distribution of various badge types across six Stack Exchange topics: AI, Gaming, History, Movies, Music, and Software Engineering. The badge categories include Question Badges, Answer Badges, Participation Badges, Moderation Badges, Other Badges, and Retired Badges.

Software Engineering has the highest count of Participation Badges, suggesting active user involvement. Gaming is notable for a high count of Question Badges, indicating a lot of queries are raised by users. Answer Badges are relatively lower across all topics, which could imply a need for more expert answers or incentives for answering questions. The lower count of Moderation Badges across most topics except Software Engineering and Gaming could reflect less user participation in community moderation activities. Other Badges and Retired Badges are minimal, which might be due to their specific nature or less frequent awarding. This distribution can help identify which communities are most engaged and where Stack Exchange might focus efforts to increase activity, such as encouraging more answers or moderating discussions.

Recommendations: Enhance community engagement by offering incentives such as increased visibility and recognition, or even physical rewards for top answers. To boost participation across various topics, implement collaborative events, peer sessions, and dedicated platforms, mirroring the high engagement found in software engineering.

2. **Cross-Community Engagement**: Exploring ways to improve community-building across different Stack Exchange topics. The dataset, encompassing multiple domains, was expected to offer insights into cross-topic engagement strategies and identify commonalities that foster community interaction.

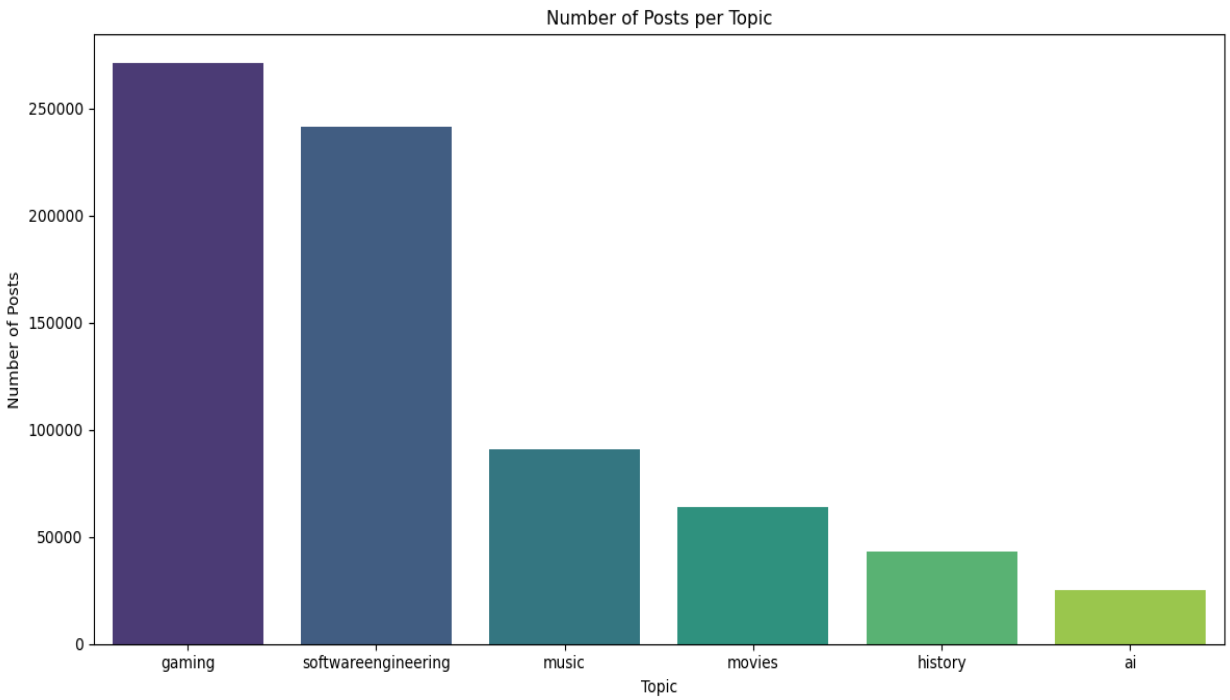
```
[[ec2-user@ip-172-31-61-143 ~]$ python3 Users_stackexchange.py
                        ai  gaming history  movies   music softwareengineering
ai                    100%   8.81%   6.15%   7.10%   6.87%              26.69%
gaming               2.83%    100%   4.68%   7.59%   5.21%              20.02%
history              9.34%  22.11%    100%  21.24%  14.96%              32.06%
movies               6.40%  21.26%  12.60%    100%  12.05%              30.27%
music                6.74%  15.91%   9.66%  13.12%    100%              28.73%
softwareengineering  4.82%  11.25%   3.81%   6.07%   5.29%                100%
Total number of users in ai: 66622
Total number of users in gaming: 207116
Total number of users in history: 43831
Total number of users in movies: 73902
Total number of users in music: 67872
Total number of users in softwareengineering: 368617
There are 1071 number of users who are active in all the 6 topics.
There are a total of 822605 users in all the 6 topics.
```

Analysis: The above displays a cross-topic user engagement analysis on Stack Exchange. The percentages indicate the overlap of users active across different topics. Software engineering has the highest internal user overlap, while AI shows significant cross-engagement with software engineering. The output indicates that 26.69% of users active in AI are also active in software engineering, suggesting a strong cross-engagement between these two technical fields. Conversely, 2.83% of users from gaming are active in AI, indicating a smaller, yet notable, overlap. This data highlights opportunities for tailored content and community strategies that bridge user interests between AI and gaming. A small subset of users (1,071) is active in all six topics, out of a total user base of 822,605. This suggests opportunities for cross-topic community building and targeted content strategies to increase engagement.

Recommendations: Initiate cross-topic events, challenges, or discussions to cater to users with diverse interests. Establish collaborative spaces to facilitate connections among users with shared interests.

3. **Content and Support Strategy**: Analyzing user post volume to enhance content and support strategies. By examining engagement metrics, especially in high-traffic areas like gaming and software engineering, the dataset could inform resource allocation and support strategies for optimal user satisfaction.

```
#!/usr/bin/env python
# coding: utf-8

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns


num_posts_list = []
topics_list = ['ai', 'gaming', 'history', 'movies', 'music', 'softwareengineering']

for topic in topics_list:
    # Load Posts.csv for the current topic
    posts_df = pd.read_csv(f"/home/ec2-user/s3_files/StackExchange/{topic}.stackexchange.com/Posts.csv")
    # Get the number of posts
    num_posts = len(posts_df)
    num_posts_list.append(num_posts)

# Create a DataFrame for visualization
data = {'Topic': topics_list, 'Number of Posts': num_posts_list}
df = pd.DataFrame(data)
# Sort the DataFrame by 'Number of Posts' in descending order
df = df.sort_values(by='Number of Posts', ascending=False)
# Create a bar plot using Seaborn
plt.figure(figsize=(12, 6))
sns.barplot(data=df, x='Topic', y='Number of Posts', palette='viridis')
plt.xlabel('Topic')
plt.ylabel('Number of Posts')
plt.title('Number of Posts per Topic')
plt.tight_layout()
# Save the plot
plt.savefig("Posts_topic.png")
# Show the bar plot
plt.show()
```

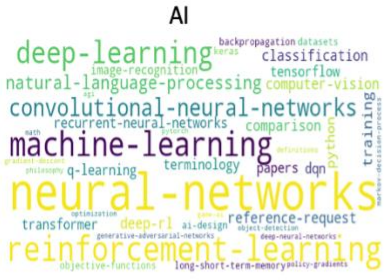


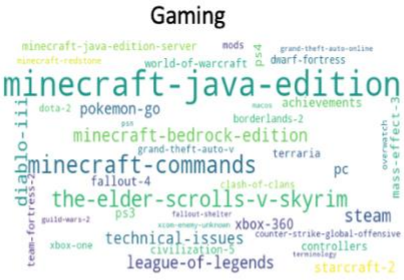
Analysis: The plot presents the number of posts per topic on Stack Exchange, indicating varying levels of activity across different fields. Gaming and software engineering lead with the highest number of posts, suggesting they are the most active communities. Music, movies, and history have moderate activity, while AI has the fewest posts, which could point towards a newer or more niche community. This data can inform content strategy and community support efforts by highlighting where user engagement is most concentrated.

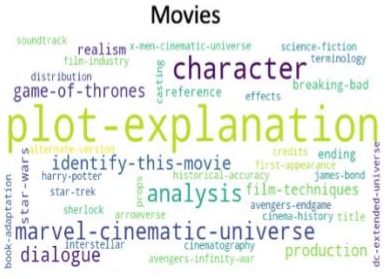
Recommendations: Allocate resources like time, staff, and funds strategically to improve content and support in highly engaged areas, such as gaming and software engineering, to efficiently meet user needs.

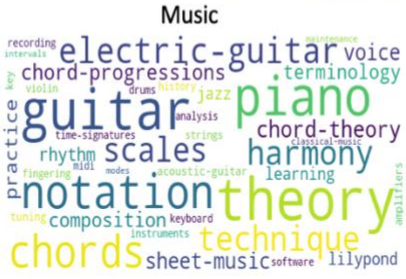
4. **User Experience Optimization through Tag Analysis**: Examining tag frequency to refine user experience. The dataset, containing information about tags associated with discussions, was seen as a valuable resource to understand user preferences and optimize search and recommendation functionalities.

```python
#!/usr/bin/env python
# coding: utf-8

from pyspark.sql import SparkSession
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
from nltk.corpus import stopwords
from collections import Counter
from IPython.display import display, Image, HTML

# Initialize Spark Session
spark = SparkSession.builder.appName("Wordcloud Processing").getOrCreate()
topics_list = ['ai', 'gaming', 'history', 'movies', 'music', 'softwareengineering']

def generate_wordcloud(topic):
    # Load Tags.csv for the current topic
    tags_df = pd.read_csv(f"/home/ec2-user/s3_files/StackExchange/{topic}.stackexchange.com/Tags.csv")
    # Create an array with duplicates for each tag based on their count
    repeated_tags = np.concatenate([np.repeat(tag, count) for tag, count in zip(tags_df['TagName'], tags_df['Count'])])
    # Count occurrences of each word
    word_counts = Counter(repeated_tags)
    # Generate a WordCloud object from the word counts
    wordcloud = WordCloud(background_color='white', width=600, height=300, max_font_size=75, max_words=40)
    wordcloud.generate_from_frequencies(word_counts)
    # Save the wordcloud to an image
    wordcloud.to_file(f"{topic}_wordcloud.png")
    # Access the words and their frequencies
    common_words = wordcloud.words_
    print("Top 5 most frequent words for: "+topic)
    # Print the top 5 most common words and their frequencies
    for idx, (word, freq) in enumerate(common_words.items()):
        if idx < 5:
            print(f'{word}: {freq}')
        else:
            break
    print('\n')

# Parallelize the task using Spark and collect results locally
spark.sparkContext.parallelize(topics_list).foreach(generate_wordcloud)
# Display the wordcloud images with titles
for topic in topics_list:
    display(HTML(f"<h2 style='text-align:left;'>Wordcloud for {topic.capitalize()}</h2>"))
    display(Image(filename=f"{topic}_wordcloud.png"))
```









Analysis: The terminal output displays the most frequent words from various Stack Exchange topics analyzed through a word cloud in PySpark. For instance, in 'movies,' the word 'plot-explanation' dominates, suggesting a focus on understanding storylines. Similarly, 'neural-networks' is prevalent in 'ai,' indicating a strong interest in this specific area of AI. Each topic's top words reflect its community's primary interests and could guide content creation, community engagement, and targeted advertising.

Recommendations: Enhance search functionalities to prominently feature discussions with popular tags, improving information discovery. By visually emphasizing frequent tags using larger fonts or unique colors and showcasing them in a 'trending' section, the platform can streamline navigation and provide a more intuitive user experience.

Predictive Analysis

Intended Use Case: The model is intended to predict the topic of every newly published post on the Stack Exchange websites. Whenever the model prediction is inconsistent with the topic under which the post was published, censors should then follow up and investigate if the post is irrelevant or otherwise harmful. And if so, actions can then be taken against the post, for instance, deleting, modifying, or blocking. Since Stack Exchange is an online community of Q&A nature, content quality is at its core of business value. A conditional censorship mechanism is, therefore, beneficial to the websites' growth.

Deliverable:

- A fine-tuned large language model (BERT) to predict the topic of Stack Exchange user posts. The model achieved an accuracy of 97%, showing significantly good performance.
- The model has been pushed to Hugging Face Hub at: https://huggingface.co/Chaconne/BDAI
- The training scripts for this model are available on Google Colab at: https://colab.research.google.com/drive/1XmEYekLYy_nXV4yX3oLM9A-JnWJ2o_Wi?authuser=1#scrollTo=8l6KMrmH0nFs

Business Value:

- Introduced an automatic anomaly detection stage in the traditional human-based censorship workflow, which significantly reduces human labor cost.
- Utilizing the state-of-the-art AI technology to acquire the best model performance.
- Improvement of content quality through the application of this topic classification workflow.

New Insights:

- We observed that the model is prone to make mistakes when trying to predict posts from closely related topics, such as gaming and history, and AI and software engineering. This is because the linguistic features of such posts are similar. For instance, the same group of words may appear frequently in both AI and software engineering, which makes it difficult for the model to distinguish between such topics.

Evaluation:

- Performant GPUs on Google Colab have been unavailable during the training period for unknown reasons. A technical compromise was thus made to accommodate a lower-tier GPU - just a small portion (10%) of the original dataset was used for training.
- The model has nonetheless achieved an accuracy of 96%, indicating very high applicability.

- In this project there was just a subset of all the Stack Exchange websites being chosen for analysis. The same training workflow for topic classification can be generalized to all the Stack Exchange websites.

Conclusion:

The Stack Exchange dataset represents a diverse repository of knowledge and community interactions. We chose this dataset due to its unique compilation of user-contributed content from various Stack Exchange sites, covering topics such as AI, Gaming, History, Movies, Music, and Software Engineering. The dataset is a testament to the collaborative efforts of users who share their knowledge, experiences, and insights across these diverse domains. The decision to explore and analyze this dataset was motivated by the desire to uncover patterns, trends, and insights within these communities. By delving into the dataset's intricacies, one can gain a deeper understanding of user behaviors, community dynamics, and the evolution of discussions across different topics. By analyzing interactions among users in various communities, we gained insights into the dynamic nature of online collaborations and discovered common interests. Examining post volume and popularity within specific domains helped us tailor content and support strategies to meet user needs effectively. Additionally, tag frequency analysis enhanced the user experience by optimizing search and recommendation systems, simplifying content discovery. Understanding how badge distribution influences participation informed strategies for recognizing and incentivizing valuable contributions, fostering a strong sense of community and expertise.

While the analysis provided deep insights into user behavior and engagement, it fell short in some respects. Expanding the dataset to include more temporal data and user interactions would be invaluable. We acknowledge the need for further investigation into specific areas where complete answers were not immediately apparent. Our findings have nonetheless provided actionable insights for informed decision-making and platform optimization, laying the groundwork for future explorations to refine our understanding.

References:

1. https://archive.org/details/stackexchange
2. https://github.com/SkobelevIgor/stackexchange-xml-converter