AUTONOMOUS TAGGING OF STACK OVERFLOW QUESTIONS

Submitted by

PROJECT ID: SPS-PRO-3348

TEAM LEADER: SAMUEL SOLOMON

TEAM MEMBER: MONISHA T

TEAM MEMBER: RANJANA P

TEAM MEMBER: ASWATHNARAAYANAN S

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1PROJECT OVERVIEW:

Autonomous tagging of Stack Overflow questions refers to the process of automatically assigning appropriate tags or categories to questions posted on the Stack Overflow platform. Stack Overflow is a popular online community for programmers and developers to ask and answer questions related to programming and software development.

Traditionally, Stack Overflow relies on its user community to manually tag questions, which can be a time- consuming task. However, in recent years, efforts have been made to automate this process using machine learning and natural language processing techniques.

Stack Overflow is a popular online community and question-and-answer platform specifically designed for programmers and developers. It serves as a valuable resource for technical knowledge, problem-solving, and collaboration within the programming community.

Stack Overflow allows users to post questions related to programming, software development, algorithms, frameworks, and other technical topics. These questions can range from beginner-level queries to complex, specialized issues.

Stack Overflow's strength lies in its active community of developers who voluntarily contribute their knowledge and expertise. Members can provide answers, suggestions, and explanations to questions posted by others.

Users can vote on the quality and helpfulness of questions and answers, which helps in determining their visibility and credibility. Stack Overflow employs a reputation system.

1.2 PURPOSE:

The purpose of Stack Overflow is to serve as an online community and knowledge-sharing platform for programmers and developers. It was created to provide a platform where developers can ask questions, share knowledge, and collaborate with their peers in solving programming-related problems.

The purpose of Stack Overflow is to create a collaborative environment that empowers developers to ask questions, share knowledge, and learn from one another, ultimately fostering growth and improvement within the programming community.

Stack Overflow allows developers to ask questions about programming, software development, and related topics. Other members of the community can provide answers, solutions, and insights based on their expertise and experience.

Stack Overflow aims to create a vast repository of programming knowledge. Questions and answers posted on the platform become accessible to anyone who may encounter similar problems in the future, helping them find solutions and learn from the collective expertise of the community.

Stack Overflow fosters an active and engaged community of developers who can connect with like-minded individuals, share ideas, and collaborate on projects. Users can follow specific tags or topics of interest and participate in discussions through comments, votes, and edits.

Developers often encounter challenges and errors while working on projects. Stack Overflow provides a platform where they can seek help in troubleshooting and resolving these issues by leveraging the expertise of the community.

Stack Overflow offers a platform for developers to enhance their skills and knowledge. By participating in discussions, answering questions, and sharing insights.

CHAPTER 2

IDEATION & PROPOSED SOLUTION

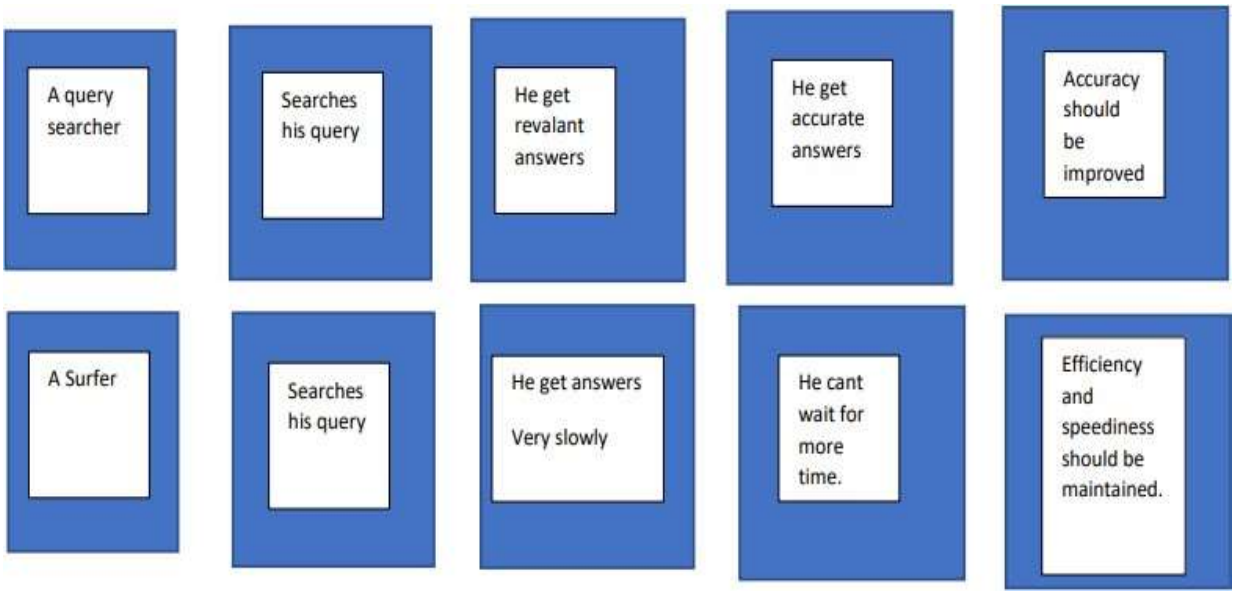
2.1 PROBLEM STATEMENT DEFINITION:

Create a problem statement to understand your customer's point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love.

A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face.

Throughout the process, you'll also be able to empathize with your customers, which helps you better understand how they perceive your product or service.

Problem definitions for Stack Overflow typically revolve around challenges faced by developers and programmers while working on software development projects or encountering programming-related issues.

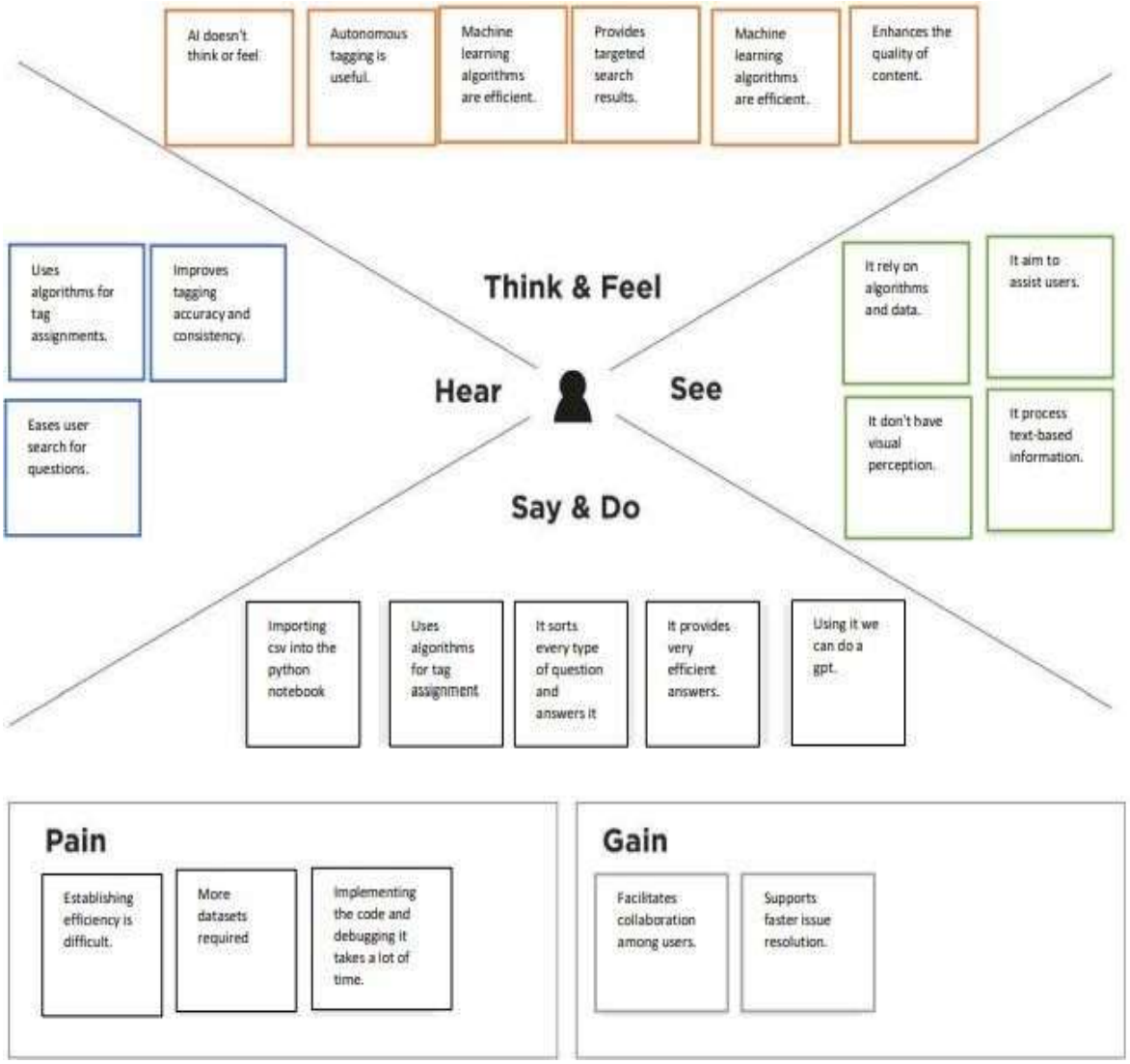


The platform caters to a wide range of programming-related challenges, and users can ask questions specific to their unique scenarios or seek general advice from the community.

2.2 EMPATHY MAP CANVAS:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user’s behaviours and attitudes.

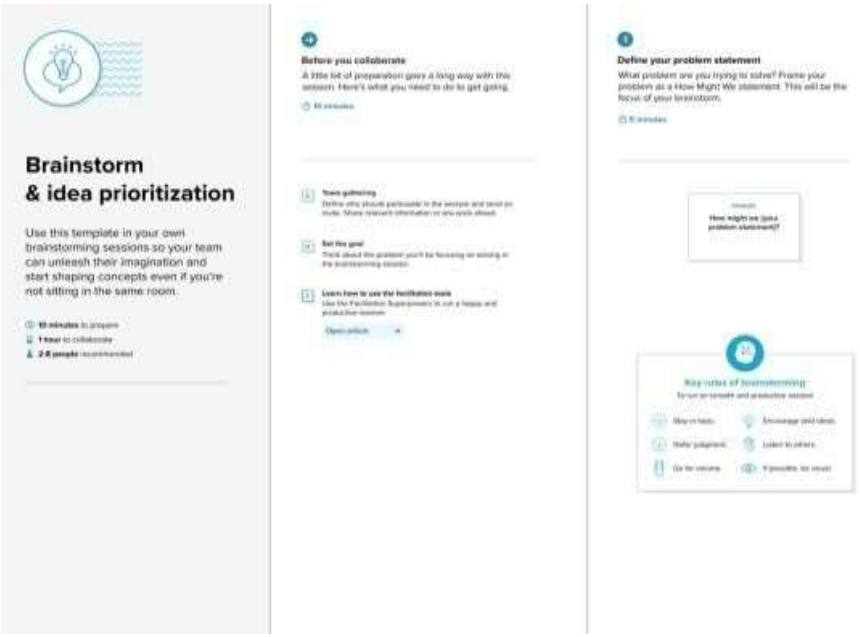
Understanding the user's perspective through an empathy map helps guide the development and improvement of Stack Overflow, ensuring that it meets the needs and expectations of its users effectively.



2.3 IDEATION & BRAINSTORMING:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of- the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

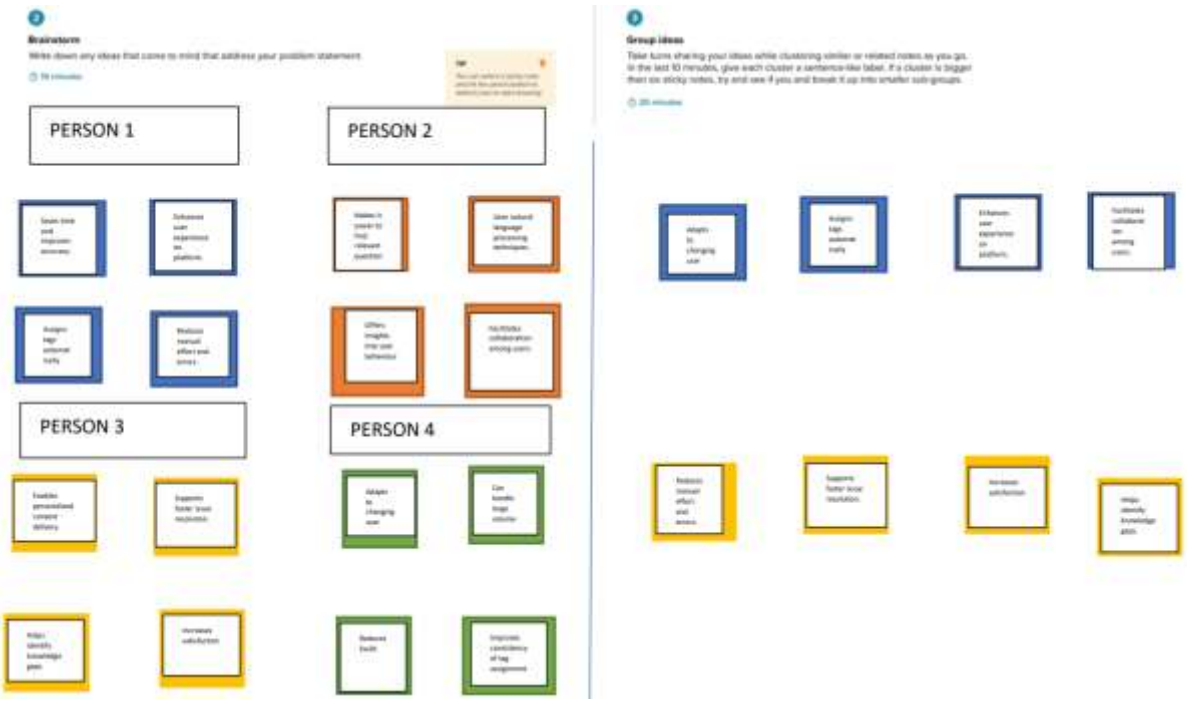
Step-1: Team Gathering, Collaboration and Select the Problem Statement



Problem

The real problem is using quora, stack overflow etc. Will not give correct answers. They will provide relevant answers only.

Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization

2.4 PROPOSED SOLUTION:

S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	Question and answer sessions are now frequently held on information-sharing platforms. Reddit, StackOverflow, Quora, and OpenEDX are a few examples. Although the amount of information on these websites has multiplied, there is no effective, automatic method for classifying data as such. The majority of these websites need users to tag their inquiries, which is not a natural way to do so.
2.	Idea / Solution description	The user experience can be enhanced by a system that allows for autonomous tagging by grouping information into distinct common subjects. Another advantage is that the user may receive suggestions for searches that are relevant to his own issue and could aid in his quick and accurate solution search.

3.	Novelty / Uniqueness	Algorithms for classification like LinearSVC, SGD classifier, and Logistic regression will be used. With these methods, we will train and evaluate the data. The best model from this set is chosen, saved, and used to integrate the model into the flask.
4.	Social Impact / Customer Satisfaction	Saves time and effort. Improves tagging accuracy and consistency. Eases user search for questions. Involves natural language processing techniques.
5.	Business Model (Revenue Model)	The user may also receive queries that are suggested for him based on his own issue, which could assist him in quickly and accurately determining the solution. The strategy for question and-answer platforms described in this project anticipates tags for a specific query.
6.	Scalability of the Solution	This technology can be used everywhere and can replace the gpt. This technology is more efficient and its convenient used by everyone.

CHAPTER 3
REQUIREMENT ANALYSIS

3.1 FUNCTIONAL REQUIREMENTS:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Real time detection	A method of detecting stack overflows is to create a canary space at the end of each task. This space is filled with some known data. If this data is ever modified, then the application has written past the end of the stack.
FR-2	Accuracy	The content quality of shared knowledge in Stack Overflow (SO) is crucial in supporting software developers with their programming problems. We use many algorithms to produce finest results.
FR-3	Data storage and analysis	Stack Overflow helps people find the answers they need, when they need them. We're best known for our public Q&A platform that over 100 million people visit every month to ask questions, learn, and share technical knowledge. We are implementing it thorugh the IBM cloud platform.

3.2 NON-FUNCTIONAL REQUIREMENTS:

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	A method of detecting stack overflows is to create a canary space at the end of each task. This space is filled with some known data. If this data is ever modified, then the application has written past the end of the stack
NFR-2	Security	A program susceptible to stack overflows can expose security vulnerabilities that hackers can exploit. By overwriting the call stack, they can insert their own executable code, which could have a significant impact on how the program works or how it is accessed.
NFR-3	Reliability	There are many techniques that can be used to detect stack overflows. Some make use of hardware while some are performed entirely in software
NFR-4	Performance	I have an object detector and now I have to decide which confidence threshold to use for each class

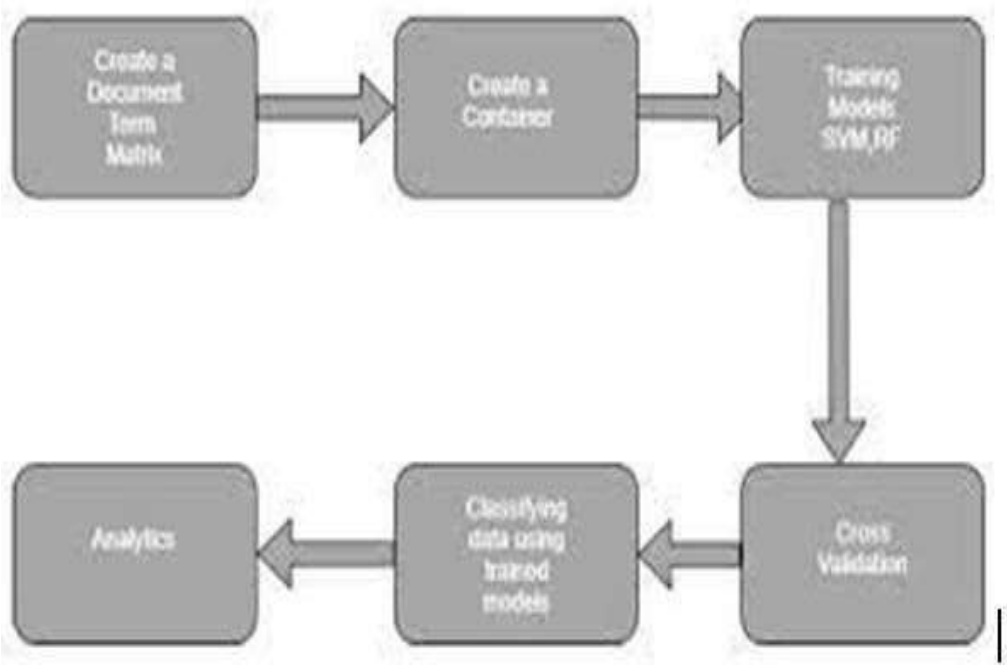
NFR-5	Availability	You must have a base case where the function stops make new recursive calls. If there is no base case then the function calls will never
		stop and eventually a stack overflow will occur
NFR-6	Scalability	The Stack Exchange team operate in a fully remote manner, and even if team members are co- located, they are encouraged to act as if they were not

CHAPTER 4

PROJECT DESIGN

4.1 DATA FLOW DIAGRAMS:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



4.2 SOLUTION &TECHNICAL ARCHITECTURE:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.

● Define features, development phases, and solution requirements.

● Provide specifications according to which the solution is defined, managed, and delivered.

Example - Solution Architecture Diagram:

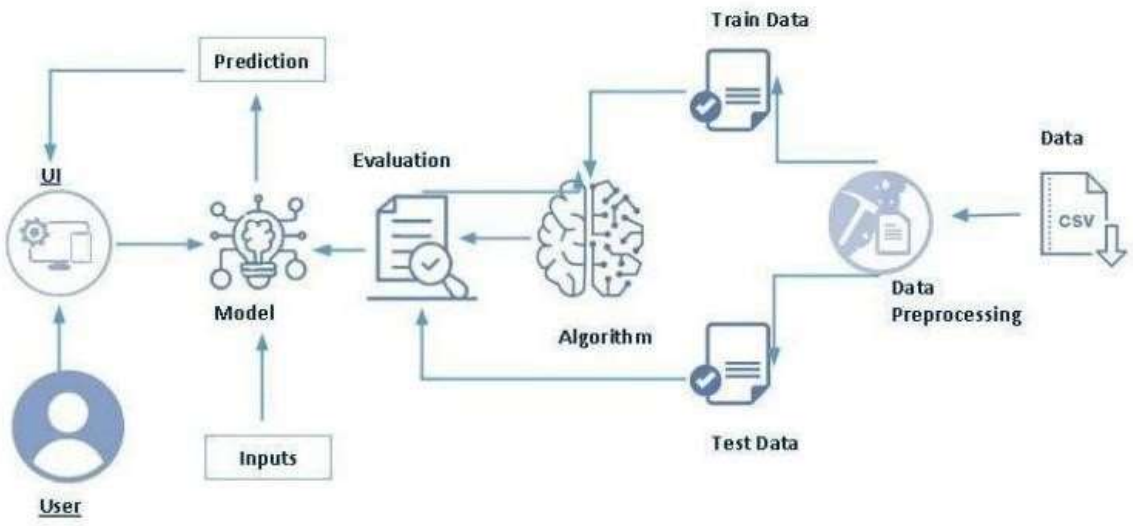


Figure 1: Architecture and data flow of Autonomous tagging of stack overflow question.

4.3 USER STORIES:

User stories in the context of autonomous Stack Overflow could be framed around the needs and interactions of users with the autonomous tagging system. Here are a few examples:

- As a Stack Overflow user, I want the autonomous tagging system to accurately assign relevant tags to my question, so that it reaches the right audience and increases the chances of receiving helpful answers.
- As a Stack Overflow user, I want the autonomous tagging system to suggest additional relevant tags based on my question's content, so that I can provide more contexts and improve the visibility of my question.

- As a Stack Overflow user, I want the autonomous tagging system to adapt and improve over time based on user feedback and interactions, so that it becomes more accurate and efficient in assigning tags to questions.
- As a Stack Overflow user, I want the autonomous tagging system to consider the programming language or framework mentioned in my question and assign appropriate language-specific tags, so that users with expertise in that specific technology can easily discover and answer my question.
- As a Stack Overflow user, I want the autonomous tagging system to suggest related questions or answers based on the assigned tags, so that I can explore similar topics and potentially find additional information or solutions.
- As a Stack Overflow moderator, I want the autonomous tagging system to flag or highlight potential spam or inappropriate tags assigned to questions, so that I can review and take appropriate actions to maintain the quality and integrity of the platform.

CHAPTER 5

CODING & SOLUTION

5.1 FEATURE 1:

TAGS:

To scrape tags from Stack Overflow using Python, you can utilize the requests and Beautiful Soup libraries. The following code demonstrates how to retrieve the tags from the Stack Overflow homepage:

```
import requests

from bs4 import BeautifulSoup


# Send a GET request to the Stack Overflow homepage response

= requests.get("https://stackoverflow.com")


# Create a BeautifulSoup object to parse the HTML content soup

= BeautifulSoup(response.text, "html.parser")


# Find the tags by locating the relevant HTML elements tag_elements

= soup.find_all("a", class_="post-tag")


# Extract the tag names from the HTML elements tags

= [tag.text for tag in tag_elements]


# Print the tags
for tag in tags: print(tag)
```

QUESTIONS:

To retrieve questions from Stack Overflow using Python, you can utilize the Stack Exchange API. Here's an example of how you can fetch the most recent questions:

```
import requests

# Set the base URL and parameters for the API request base_url

= "https://api.stackexchange.com/2.3/questions" params = {

    "site": "stackoverflow",

    "order": "desc",

    "sort": "creation",

    "tagged": "python",

    "pagesize": 10

}

# Send a GET request to the Stack Exchange API

response = requests.get(base_url, params=params) data

= response.json()


# Extract the questions from the response questions

= data["items"]


# Print the titles of the questions for

question in questions:

print(question["title"])
```

BODY:

To retrieve the body of a Stack Overflow question using the Stack Exchange API in Python, you can modify the previous code snippet. Here's an example:

```
import requests

# Set the base URL and parameters for the API request base_url
= "https://api.stackexchange.com/2.3/questions" params = {
    "site": "stackoverflow",
    "order": "desc",
    "sort": "creation",
    "tagged": "python",
    "pagesize": 1,
    "filter": "!9Z(-wzftf" # Filter to include question body

}

# Send a GET request to the Stack Exchange API
response = requests.get(base_url, params=params) data
= response.json()

# Extract the question from the response question
= data["items"][0]

# Print the question title and body print("Title:",
question["title"])                    print("Body:",
question["body"])
```

In this example, we set the pagesize parameter to 1 to retrieve

only one question. We also include a filter parameter in the params dictionary with the value "!9Z(-wzftf".

This filter instructs the API to include the body of the question in the response.After parsing the JSON response, we extract the first question from the items list and print its title and body.

The parameters in the params dictionary specify the sorting order, the tagged parameter to filter by tags, and the pagesize parameter to limit the number of questions returned.

The API response is in JSON format, so we parse it using the json() method provided by the response object. We extract the list of questions from the JSON response and print the titles of the questions.

The API response is in JSON format, so we parse it using the json() method provided by the response object. We extract the body of the first answer from the JSON response and print it. If no answers are found for the question, an appropriate message is displayed.

The API parameters in the params dictionary specify the sorting order, filter for retrieving the answer bodies, and the site parameter set to "stackoverflow" to target Stack Overflow.

Remember to respect the Stack Exchange API usage guidelines, such as making reasonable and appropriate use of the API and adhering to rate limits.

5.2 FEATURE 2:

TAGS:

	Id	Tag
0	80	flex
1	80	actionscript-3
2	80	air
3	90	svn
4	90	tortoisesvn

QUESTIONS:

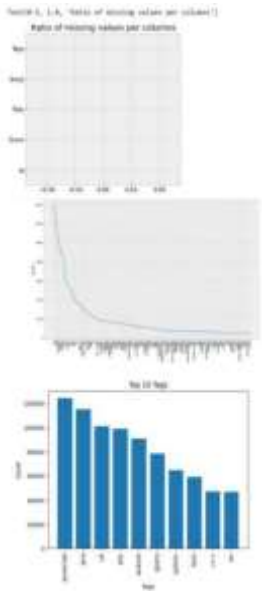
	Id	OwnerUserId	CreationDate	ClosedDate	Score	Title	Body
0	80	26.0	2008-08-01T13:57:07Z	NaN	26	SQLStatement.execute() - multiple queries in o...	<p>I've written a database generation script i...
1	90	58.0	2008-08-01T14:41:24Z	2012-12-26T03:45:49Z	144	Good branching and merging tutorials for Torto...	<p>Are there any really good tutorials explain...
2	120	83.0	2008-08-01T15:50:08Z	NaN	21	ASP.NET Site Maps	<p>Has anyone got experience creating ...
3	180	2089740.0	2008-08-01T18:42:19Z	NaN	53	Function for creating color wheels	<p>This is something I've pseudo-solved many t...
4	260	91.0	2008-08-01T23:22:08Z	NaN	49	Adding scripting functionality to .NET applica...	<p>I have a little game written in C#. It uses...

ANSWERS:

	Id	OwnerUserId	CreationDate	ParentId	Score	Bod
0	92	61.0	2008-08-01T14:45:37Z	90	13	<p><a href="http://svnbook.rebean.com/">Vers
1	124	26.0	2008-08-01T16:09:47Z	80	12	<p>I wound up using this. It is a kind of ha
2	199	50.0	2008-08-01T19:36:46Z	180	1	<p>I've read somewhere the human e can't dis
3	269	91.0	2008-08-01T23:49:57Z	260	4	<p>Yes, I thought about that, but I soo figu
4	307	49.0	2008-08-02T01:49:46Z	260	28	<p><href="http://www.codeproject.com/Article

CHAPTER 6
RESULT

6.1 PERFORMANCE METRICS:

S.No.	Parameter	Values	Screenshot
1.	Tune the Model+ Metrics	**Regression Model:** MAE - , MSE - , RMSE - , R2 score – **Classification Model:** Confusion Matrix - , Accuray Score- & Classification Report – Hyperparameter Tuning - Validation Method	Classifier: Logistic Regression Hamming Loss: 0.1014961053764802 Precision: 0.8683652755244097 Recall: 0.7309219283523352 Classifier: MultinomialNB Hamming Loss: 0.1123298081185485 Precision: 0.8435391453577921 Recall: 0.7102899473349152 Classifier: LinearSVC Hamming Loss: 0.11232980811854854 Precision: 0.8435391453577921 Recall: 0.7102899473349152 Classifier: RandomForestClassifier Hamming Loss: 0.10622348173009942 Precision: 0.8949286398330041 Recall: 0.6793622316106256
2.	Exploratory Data Analysis + Handling missing values and cleansing the tags and body column	Dropping the missing values Dependency Graph Tags Vs Count	

CHAPTER 7

ADVANTAGES & DISADVANTAGES

ADVANTAGES:

Improved Tag Accuracy:

Autonomous tagging leverages machine learning and natural language processing techniques to analyze the content of questions and assign appropriate tags. This results in more accurate and relevant tagging compared to manual tagging, reducing the chances of misclassification or missing important tags.

Time and Effort Saving:

Autonomous tagging automates the process of assigning tags to questions, eliminating the need for manual tagging by users. This saves time and effort for both the question askers and the community, allowing them to focus on answering questions and engaging in discussions.

Enhanced Discoverability:

Accurate and relevant tagging improves the discoverability of questions. Users can easily search and filter questions based on specific tags or topics of interest, increasing the chances of finding relevant questions and providing targeted answers.

Reduced Duplicate Questions:

Autonomous tagging can help identify similar or duplicate questions by analyzing their content and assigned tags. This helps prevent the proliferation of duplicate questions, allowing users to find existing answers and reducing redundancy in the platform.

Enriched User Engagement:

By providing accurate tags, autonomous tagging encourages

experts and users with relevant knowledge to discover questions within their areas of expertise. This leads to increased user engagement, as experts are more likely to provide helpful answers and insights to questions that match their expertise.

Continuous Improvement:

Autonomous tagging systems can be refined and improved over time. User feedback and interactions with the system can be used to train and update the tagging models, ensuring that the system adapts to evolving user needs and provides increasingly accurate and relevant tags.

Scalability and Consistency:

Autonomous tagging allows for scalability, as it can handle large volumes of questions efficiently and consistently. It ensures that all questions receive appropriate tags based on their content, regardless of the number of questions being posted.

Overall, autonomous tagging in Stack Overflow offers advantages such as improved tag accuracy, time and effort savings, enhanced discoverability, reduced duplicates, enriched user engagement, continuous improvement, and scalability. These benefits contribute to a more efficient and effective platform for knowledge sharing and collaboration among developers and programmers.

DISADVANTAGES:

Tagging Errors:

Autonomous tagging systems may occasionally assign incorrect or irrelevant tags to questions. The models rely on patterns and training data, which may not always capture the nuances or specific context of a question accurately. This can result in misclassified tags and potentially lead to confusion or hinder the discoverability of questions.

Lack of Human Interpretation:

Autonomous tagging lacks the human judgment and interpretation that manual tagging provides. Human taggers can understand the nuances, context, and intent of a question better, leading to more accurate and nuanced tagging. Autonomous systems may struggle to capture these subtleties effectively.

Challenges with Ambiguous or Uncommon Topics:

Autonomous tagging can struggle with questions related to ambiguous or less common topics. If the training data is primarily focused on popular or mainstream programming languages and frameworks, it may not perform well when tagging questions related to niche or emerging technologies.

Inability to Capture Changing Language Trends:

Programming languages, frameworks, and technologies evolve over time, and new ones emerge. Autonomous tagging systems may face challenges in keeping up with these changes, as they rely on historical data and pre-existing patterns. This can result in outdated or incomplete tagging for newer technologies.

Overreliance on Tagged Questions:

Autonomous tagging systems heavily rely on previously tagged questions for training. If the training data is biased, incomplete, or contains errors, it can affect the performance and accuracy of the tagging system. This can lead to a perpetuation of incorrect or biased tags in the platform.

Limited Adaptability to User Preferences:

Autonomous tagging systems may not effectively capture user preferences or subjective factors when assigning tags. Users may have specific criteria or preferences for the tags they expect, which may not align with the system's automated tagging decisions.

Lack of Transparency:

Users may not fully understand how autonomous tagging

systems work or the specific criteria used for tag assignment. This lack of transparency can lead to confusion or mistrust regarding the accuracy and relevance of the assigned tags.

It's important to note that these disadvantages can be mitigated through continuous refinement, user feedback, and a combination of autonomous tagging with human moderation and intervention to ensure accurate and reliable tag assignment.

CHAPTER 8

CONCLUSION

In conclusion, autonomous tagging of Stack Overflow questions offers several advantages and brings efficiency to the platform. It improves tag accuracy, saves time and effort, enhances discoverability, reduces duplicate questions, fosters user engagement, allows for continuous improvement, and ensures scalability. However, there are also potential disadvantages to consider, such as tagging errors, the lack of human interpretation, challenges with ambiguous or uncommon topics, difficulty capturing changing language trends, reliance on biased or incomplete training data, limited adaptability to user preferences, and a lack of transparency. These drawbacks can be addressed through continuous refinement, user feedback, and a combination of autonomous tagging with human moderation. By striking the right balance between automation and human intervention, Stack Overflow can leverage autonomous tagging to improve the user experience, facilitate knowledge sharing, and promote effective collaboration within the programming community.

However, there are also potential disadvantages to consider, such as tagging errors, lack of human interpretation, challenges with ambiguous or uncommon topics, inability to capture changing language trends, overreliance on tagged questions, limited adaptability to user preferences, and a lack of transparency. These drawbacks highlight the importance of continuous refinement, user feedback, and a combination of autonomous tagging with human moderation to ensure accurate and reliable tag assignment.

Overall, while autonomous tagging systems can greatly assist in organizing and categorizing the vast amount of content on Stack Overflow, it is essential to strike a balance between automation and human involvement to maintain the integrity, accuracy, and relevance of the tags assigned to questions.

By leveraging the strengths of both autonomous tagging and human moderation, Stack Overflow can continue to be a valuable resource for developers worldwide.

CHAPTER 9

FUTURE SCOPE

The future scope for autonomous tagging of Stack Overflow involves advancements and enhancements in several areas. Here are some potential future directions:

Improved Tagging Models:

Continued research and development in natural language processing (NLP) and machine learning can lead to more advanced tagging models. These models can better understand the context, intent, and nuances of questions, resulting in even more accurate and relevant tag assignments.

Integration of User Feedback:

Leveraging user feedback and interactions to improve the tagging system is crucial. Future developments could involve mechanisms to allow users to provide feedback on the assigned tags, report misclassifications, and suggest alternative or additional tags. This feedback loop can enhance the system's learning and adaptability.

Enhanced Topic Coverage:

Autonomous tagging can expand its coverage to include a broader range of programming languages, frameworks, and emerging technologies. This involves continuously updating the training data and models to keep pace with the evolving programming landscape.

Customization and Personalization:

Providing users with the ability to customize and personalize the tagging system can further enhance the user experience. This could include options to define preferred tags, prioritize certain topics, or adjust the tag assignment algorithm based on individual preferences.

Multilingual Support: Expanding autonomous tagging to support multiple languages can facilitate broader global participation and knowledge sharing on Stack Overflow. This would involve training the models on multilingual datasets and ensuring accurate tag assignment across different programming languages.

Advanced Tag Recommendation:

Autonomous tagging systems can evolve to provide intelligent tag recommendations while users compose their questions. These recommendations can be based on the question content, similar questions, user history, or other contextual information, helping users assign appropriate tags more effectively.

Collaboration with Human Moderators:

A collaborative approach that combines autonomous tagging with human moderation can lead to optimal results. Human moderators can review and validate tag assignments, intervene when necessary, and ensure the overall quality and relevance of the tagging system.

Continuous Learning and Updates:

Autonomous tagging systems should be designed to learn from user interactions, feedback, and evolving trends. Regular updates to the tagging models and algorithms can improve their performance and adapt to changing user needs and preferences.

The future of autonomous tagging on Stack Overflow lies in a combination of advanced machine learning techniques, user involvement, customization options, and collaboration with human moderators. By harnessing these advancements, Stack Overflow can enhance the accuracy, efficiency, and overall user experience of its autonomous tagging system.

CHAPTER 10

APPENDIX

IMPORTING NECESSARY LIBRARIES

```
import pandas as pd import
numpy as np
import matplotlib.pyplot as plt import
matplotlib.lines as mlines import
seaborn as sns
import warnings import pickle
import time import re from bs4
import BeautifulSoup import
nltk from nltk.tokenize
import ToktokTokenizer from nltk.stem.wordnet
import WordNetLemmatizer from nltk.corpus import
stopwords from string
import punctuation
import sklearn.linear_model from sklearn.feature_extraction.text
import TfidfVectorizer from sklearn.decomposition import
LatentDirichletAllocation from sklearn.preprocessing import
MultiLabelBinarizer from sklearn.model_selection import
train_test_split from sklearn.model_selection import
learning_curve from sklearn.model_selection import ShuffleSplit
from sklearn.naive_bayes import MultinomialNB from
sklearn.linear_model
```

```
import LinearRegression from
sklearn.linear_model
import SGDClassifier from sklearn.linear_model
import LogisticRegression from sklearn.svm
import LinearSVC from sklearn.linear_model
import Perceptron from sklearn.neural_network
import MLPClassifier from sklearn.ensemble
import RandomForestClassifier from sklearn
import model_selection from sklearn.metrics
import make_scorer from sklearn.metrics
import hamming_loss import logging from
scipy.sparse import hstack
warnings.filterwarnings("ignore")
np.random.seed(seed=11)
```

READ THE DATASET

In 1: ques=pd.read_csv(r"Questions.csv", encoding="ISO-8859-1") ques.head()

Out[1]:

	Id	OwnerUserId	CreationDate	ClosedDate	Score	Title	Body
0	80	26.0	2008-08-01T13:57:07Z	NaN	26	SQLStatement.execute() - multiple queries in o...	<p>I've written a database generation script i...
1	90	58.0	2008-08-01T14:41:24Z	2012-12-26T03:45:49Z	144	Good branching and merging tutorials for Torto...	<p>Are there any really good tutorials explain...
2	120	83.0	2008-08-01T15:50:08Z	NaN	21	ASP.NET Site Maps	<p>Has anyone got experience creating ...
3	180	2089740.0	2008-08-01T18:42:19Z	NaN	53	Function for creating color wheels	<p>This is something I've pseudo-solved many t...
4	260	91.0	2008-08-01T23:22:08Z	NaN	49	Adding scripting functionality to .NET applica...	<p>I have a little game written in C#. It uses...

IN 2: tags=pd.read_csv(r"Tags.csv") tags.head(5)

OUT[2]:

	Id	Tag
0	80	flex
1	80	actionscript-3
2	80	air
3	90	svn
4	90	tortoisesvn

IN 3:ques.info ()

OUT[3]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1264216 entries, 0 to 1264215
Data columns (total 7 columns):
 #   Column        Non-Null Count    Dtype
---  ------        --------------    -----
 0   Id            1264216 non-null  int64
 1   OwnerUserId   1249762 non-null  float64
 2   CreationDate  1264216 non-null  object
 3   ClosedDate    55959 non-null    object
 4   Score         1264216 non-null  int64
 5   Title         1264216 non-null  object
 6   Body          1264216 non-null  object
dtypes: float64(1), int64(2), object(4)
memory usage: 67.5+ MB
```

IN 4: tags.info ()

OUT [4]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3750994 entries, 0 to 3750993
Data columns (total 2 columns):
 #   Column  Dtype
---  ------  -----
 0   Id      int64
 1   Tag     object
dtypes: int64(1), object(1)
memory usage: 57.2+ MB
```

IN 5: answers=pd.read_csv(r"Answers.csv", encoding="ISO-8859-1")

answers.head()

OUT [5]:

	Id	OwnerUserId	CreationDate	ParentId	Score	Boc
0	92	61.0	2008-08-01T14:45:37Z	90	13	<p><a href="http://svnbook.rec bean.com/">Vers
1	124	26.0	2008-08-01T16:09:47Z	80	12	<p>I wound up using this. It is a kind of ha
2	199	50.0	2008-08-01T19:36:46Z	180	1	<p>I've read somewhere the human ey can't dis
3	269	91.0	2008-08-01T23:49:57Z	260	4	<p>Yes, I thought about that, but I soc figu
4	307	49.0	2008-08-02T01:49:46Z	260	28	<p>< href="http://www.codeproject.com/Article

IN 6: import pandas as pd

Read the CSV files questions_df =

pd.read_csv('questions.csv',encoding="ISO-8859-1") answers_df =

pd.read_csv('answers.csv',encoding="ISO-8859-1") tags_df =

pd.read_csv('tags.csv')

Merge the DataFrames

new_df = pd.merge(questions_df, answers_df, on='Id')

new_df = pd.merge(new_df, tags_df, on='Id') #

Save the merged DataFrame to a new CSV file

new_df.to_csv('new_df.csv', index=False)

DATA PREPROCESSING

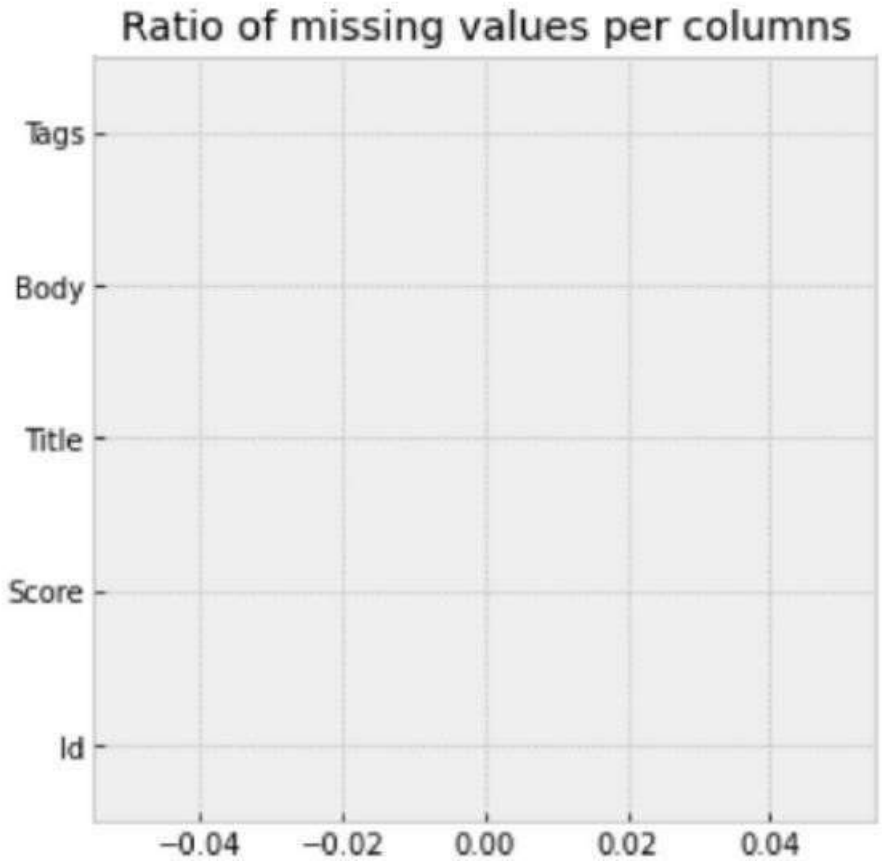
- **HANDLING MISSING VALUES:**

IN 7: import pandas as pd

```
# Read questions.csv questions_df = pd.read_csv('Questions.csv') # Handle missing values in questions.csv
questions_df.fillna('', inplace=True)
# Read answers.csv
answers_df = pd.read_csv('Answers.csv') # Handle missing values in answers.csv
answers_df.fillna('', inplace=True)
# Read tags.csv
tags_df = pd.read_csv('Tags.csv') # Handle missing values in tags.csv
tags_df.fillna('', inplace=True)
# Perform further processing or analysis on the data
# ...
plt.figure(figsize=(5,5)) new_df.isnull().mean(axis=0).plot.barh()
plt.title("Ratio of missing values per column")
```

OUT [7]:

```
Text(0.5, 1.0, 'Ratio of missing values per columns')
```



IN8: print('Dupplicate entries:{}',format(new_df.duplicated().sum()))

new_df.drop_duplicates(inplace=True)

- **CLEANING TAGS COLUMN**

IN 8: import pandas as pd

# Read questions.csv
questions_df = pd.read_csv('Questions.csv') #

Clean the tags column in questions.csv

questions_df['Tags'] = questions_df['Tags'].str.replace('><', ',')

questions_df['Tags'] = questions_df['Tags'].str.replace('', '')

```
# Read answers.csv
answers_df = pd.read_csv('Answers.csv')
# Clean the tags column in answers.csv answers_df['Tags']
= answers_df['Tags'].str.replace('><', ',')
answers_df['Tags'] = answers_df['Tags'].str.replace(", ")
# Read tags.csv tags_df =
pd.read_csv('Tags.csv') # Clean the
Tag column in tags.csv
tags_df['Tag'] = tags_df['Tag'].str.replace('><', ',') tags_df['Tag']
= tags_df['Tag'].str.replace(", ")
# Perform further processing or analysis on the cleaned tags data
# ...
new_df['Tags']=new_df['Tags'].apply(lambda x:x.split()) all_tags=[item
for sublist in new_df['Tags].values for item in sublist']] len(all_tags)
```

OUT 8:

```
224129
```

IN 9: my_set=set(all_tags)

```
unique_tags=list(my_set) len(unique_tags)
```

OUT [9]:

```
14883
```

IN 10: new_df.head(5)

OUT[10]:

	Title	Body	Tags
0	SQLStatement.execute() - multiple queries in o...	<p>I've written a database generation script i...	flex actionscript-3 air
1	Good branching and merging tutorials for Torto...	<p>Are there any really good tutorials explain...	svn tortoisesvn branch branching-and-merging
2	ASP.NET Site Maps	<p>Has anyone got experience creating ...	sql asp.net sitemap
3	Function for creating color wheels	<p>This is something I've pseudo-solved many t...	algorithm language-agnostic colors color-space
4	Adding scripting functionality to .NET applica...	<p>I have a little game written in C#. It uses...	c# .net scripting compiler-construction

IN 11: flat_list=[item for sublist in new_df['Tags'].values for item in sublist]

keywords=nltk.freqDist(flat_list)

keywords=nltk.FreqDist(keywords)

frequencies_words=keywords.most_common(100)

tags_features=[word[0] for word in frequencies_words]

tags_features

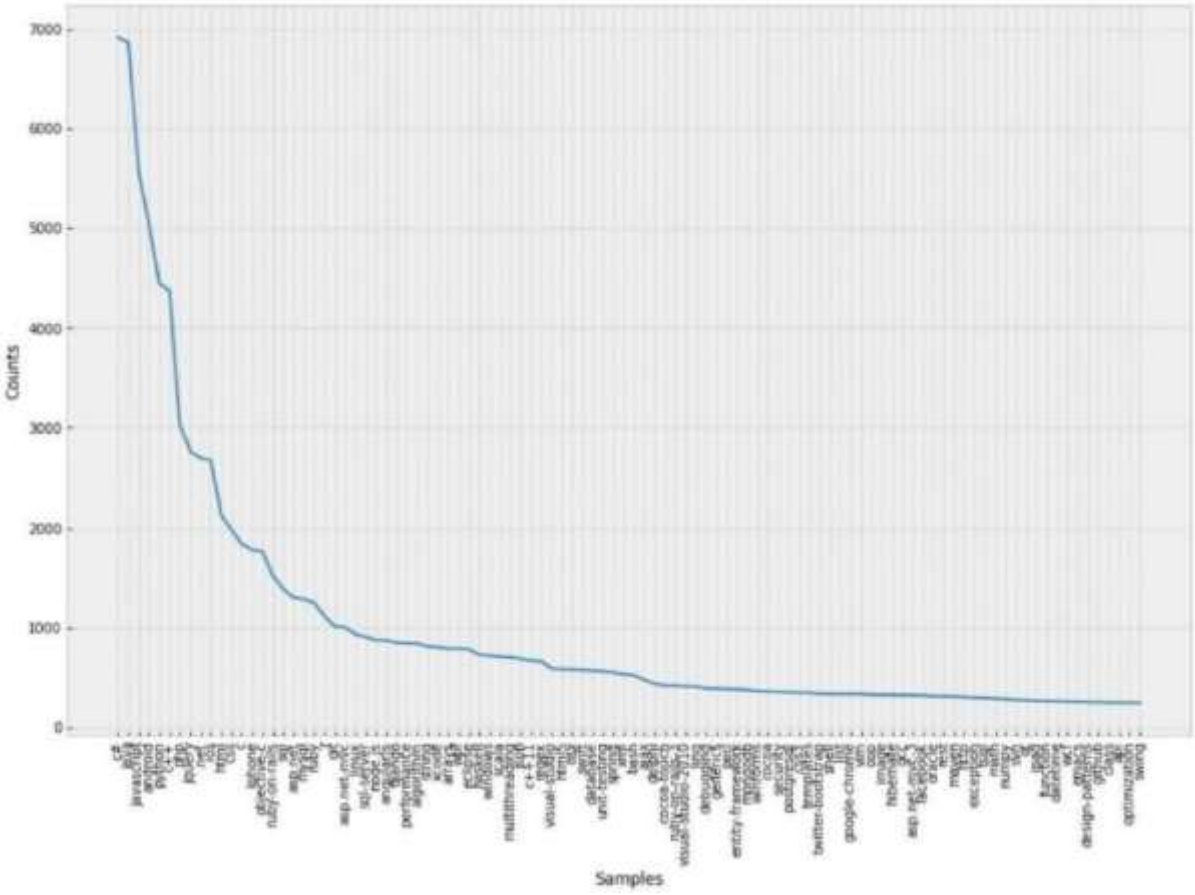
OUT[11]:

```
['c#',
 'java',
 'javascript',
 'android',
 'python',
 'c++',
 'php',
 'jquery',
 '.net',
 'ios',
 'html',
```

IN 12: fig,ax=plt.sunplots(figsize=(15,10))

keywords.plot(100,cumulative=False)

OUT[12]:



IN 13: def most_common(tags):

tags_filtered=[]

for i in range(0,len(tags)):

if tags[i] in tags_features:

tags_filtered.append(tags[i]) return tags_filtered

new_df['Tags']=new_df['Tags'].apply(lambda x: most_coomon(x))

new_df['Tags']=new_df['Tags'].apply(lambda x:x if len(x)0 else None)

new_df.shape

OUT [13]:

(72950, 3)

IN 14:

new_df.dropna(subset=['Tags'],inplace=True)

new_df.heas(5) new_df.shape

OUT [14]:

(63167, 3)

IN 15: new_df['Bodt']=new_df['Body'].apply(lambda x:BeautifulSoup(x).get_text()) def clean_text(text): text=text.lower()

text=re.sub(r"What's","What is",text)

text=re.sub(r"\'s"," ",text) text=re.sub(r"\'ve","have",text)

text=re.sub(r"can't","can not",text)

text=re.sub(r"n't","not",text)

text=re.sub(r"i'm","i am",text)

text=re.sub(r"\re", "are",text)

text=re.sub(r"\d", "would",text)

text=re.sub(r"\'ll","will",text)

text=re.sub(r"\'scuse","exuse",text)

text=re.sub(r"\'\n"," ",text) text=re.sub(r"\'\n"," ",text)

text=re.sub('s+',",text) text=text.strip(") return text

IN 16:

new_df['body']=new_df['body'].apply(lambda x; clean_text(x))

IN 17: token=ToktokTtokrnizer()

IN 18: punct='!#$&\'()*+,./:;<=>?@[\\]^_'{|}~' def strip_list_noempty(my list):

newlist=(item.strip() if hasattr(item, 'strip')else item for item in

mylist) return[item for item in new list if item!=''] def

clean_punch(text):

words=token.tokenize(text) punctuation_filtered=[]

regex=re.compile('[%5]' % re.escape(punct))

remove_punctuation=str.maketrans(",",punct)

for w in words:

if w in tags_features:

punctuation_filtered.append(w) else:

IN 19: punctuation_filtered.append(regrex.sub(''),w))

filtered_list=strip_list_noempty(punctuation_filtered)

return''.join(map(str,filtered_list))

IN 20: new_df['body']=new_df['body'].apply(lambda x: clean_punt(x))

IN 21: new_df['body'][2] **OUT**

[21]:

'has anyone got experience creating sql-based asp.net site-map providers i have got the default xml file website working properly with my menu and site map path controls but i will need a way for the users of my site to create and modify pages dynamically i need to tie page viewing permissions into the standard asp.net membership system as well'. **IN 22:** def lemitizeWords(text):

words=token.tokenize(text)

listlemma=[] for w in words:

x=lemma.lemmatize(w,pos"v")

listlemma.append(x)

return''.join(map(str,listlemma))

def stopWordsRemove(text):

stop_words=set(stopwords.words("english"))

words=token.tokenize(text) filtered=[w for w in words if not w in stop_words] return".join(map(str,filtered))

IN 23: new_df['body']=new_df['body'].apply(lambda x: lemitzewords(x))

new_df['body']=new_df['body'].apply(lambda x: stopwordsremove (x))

IN 24: new_df.head(5)

OUT [24]:

	Title	Body	Tags
1	Good branching and merging tutorials for Torto...	really good tutorials explain branch merge apa...	[svn]
2	ASP.NET Site Maps	anyone get experience create sql-based asp.net...	[sql, asp.net]
3	Function for creating color wheels	something pseudo-solved many time never quite ...	[algorithm]
4	Adding scripting functionality to .NET applica...	little game write c use database back-end trad...	[c#, .net]
5	Should I use nested classes in this case?	work collection class use video playback recor...	[c++, oop, class]

IN 25: new_df['Title']=new_df['Title'].apply(lambda x:str(x))

new_df['Title']=new_df['Title'].apply(lambda x:clean_text(x))

new_df['Title']=new_df['Title'].apply(lambda x:clean_punct(x))

new_df['Title']=new_df['Title'].apply(lambda x:lemitizewords(x))

new_df['Title']=new_df['Title'].apply(lambda x:stopWordsRemove(x))

new_df.head(5)

OUT [25]:

	Title	Body	Tags
1	good branch merge tutorials tortoisesvn	really good tutorials explain branch merge apa...	[svn]
2	asp.net site map	anyone get experience create sql-based asp.net...	[sql, asp.net]
3	function create color wheel	something pseudo-solved many time never quite ...	[algorithm]
4	add script functionality .net applications	little game write c use database back-end trad...	[c#, .net]
5	use nest class case	work collection class use video playback recor...	[c++, oop, class]

EXPLORATORY DATA ANALYSIS:

IN26: import pandas as pd import

matplotlib.pyplot as plt

Display the first few rows of each

dataset print("Questions:")

print(questions_df.head())

print("\nAnswers:")

print(answers_df.head()) print("\nTags:")

print(tags_df.head())

Get the summary statistics of numerical columns in each

table print("Questions statistics:")

print(questions_df.describe())

print("\nAnswers statistics:")

print(answers_df.describe()) print("\nTags

statistics:") print(tags_df.describe())

Check the data types of each column in each table

print("Questions data types:")

print(questions_df.dtypes) print("\nAnswers data

types:") print(answers_df.dtypes) print("\nTags

data types:") print(tags_df.dtypes)

Count the number of missing values in each column in each table

print("Questions missing values:") print(questions_df.isnull().sum())

print("\nAnswers missing values:")

print(answers_df.isnull().sum())

```
  print("\nTags missing values:")
  print(tags_df.isnull().sum())
# Visualize the distribution of scores in the Questions table
  plt.hist(questions_df['Score'], bins=20) plt.xlabel('Score')
plt.ylabel('Frequency')
plt.title('Distribution of Scores in Questions') plt.show()
# Visualize the distribution of scores in the Answers table
plt.hist(answers_df['Score'], bins=20) plt.xlabel('Score')
plt.ylabel('Frequency')
plt.title('Distribution of Scores in Answers')
 plt.show()  #  Explore
the top tags
top_tags = tags_df['Tag'].value_counts().head(10)
plt.bar(top_tags.index, top_tags.values) plt.xlabel('Tags')
plt.ylabel('Count') plt.title('Top
10 Tags')
plt.xticks(rotation=90)
plt.show()
```

OUT [26]:

```
Questions:
    Id  OwnerUserId          CreationDate            ClosedDate  Score
0   80         26.0  2008-08-01T13:57:07Z                   NaN     26  \
1   90         58.0  2008-08-01T14:41:24Z  2012-12-26T03:45:49Z    144
2  120         83.0  2008-08-01T15:50:08Z                   NaN     21
3  180    2089740.0  2008-08-01T18:42:19Z                   NaN     53
4  260         91.0  2008-08-01T23:22:08Z                   NaN     49
```

```
                                                  Title
0  SQLStatement.execute() - multiple queries in o.……..\
1  Good branching and merging tutorials for Torto...
2                                   ASP.NET Site Maps
3                  Function for creating color wheels
4  Adding scripting functionality to .NET applica...


                                                   Body
0  <p>I've written a database generation script i...
1  <p>Are there any really good tutorials explain...
2  <p>Has anyone got experience creating <strong>...
3  <p>This is something I've pseudo-solved many t...
4  <p>I have a little game written in C#. It uses...

Answers:
    Id  OwnerUserId         CreationDate  ParentId  Score
0   92         61.0  2008-08-01T14:45:37Z        90     13
1  124         26.0  2008-08-01T16:09:47Z        80     12
2  199         50.0  2008-08-01T19:36:46Z       180      1
3  269         91.0  2008-08-01T23:49:57Z       260      4
4  307         49.0  2008-08-02T01:49:46Z       260     28

                                                   Body
0  <p><a  href="http://svnbook.red-bean.com/">Vers...
1  <p>I wound up using this. It is a kind of a ha...
2  <p>I've read somewhere the human eye can't dis...
3  <p>Yes, I thought about that, but I soon figur...
4  <p><a  href="http://www.codeproject.com/Article...

Tags:
   Id             Tag
0  80            flex
1  80  actionscript-3
2  80             air
3  90             svn
4  90     tortoisesvn
```

```
Questions statistics:
                 Id    OwnerUserId        Score
count  1.264216e+06  1.249762e+06  1.264216e+06
mean   2.132745e+07  2.155177e+06  1.781537e+00
std    1.151445e+07  1.801265e+06  1.366389e+01
min   8.000000e+01  1.000000e+00  -7.300000e+01
25%    1.142598e+07  6.589110e+05  0.000000e+00
50%    2.172542e+07  1.611830e+06  0.000000e+00
75%    3.154542e+07  3.353792e+06  1.000000e+00
max    4.014338e+07  7.046594e+06  5.190000e+03

Answers statistics:
```

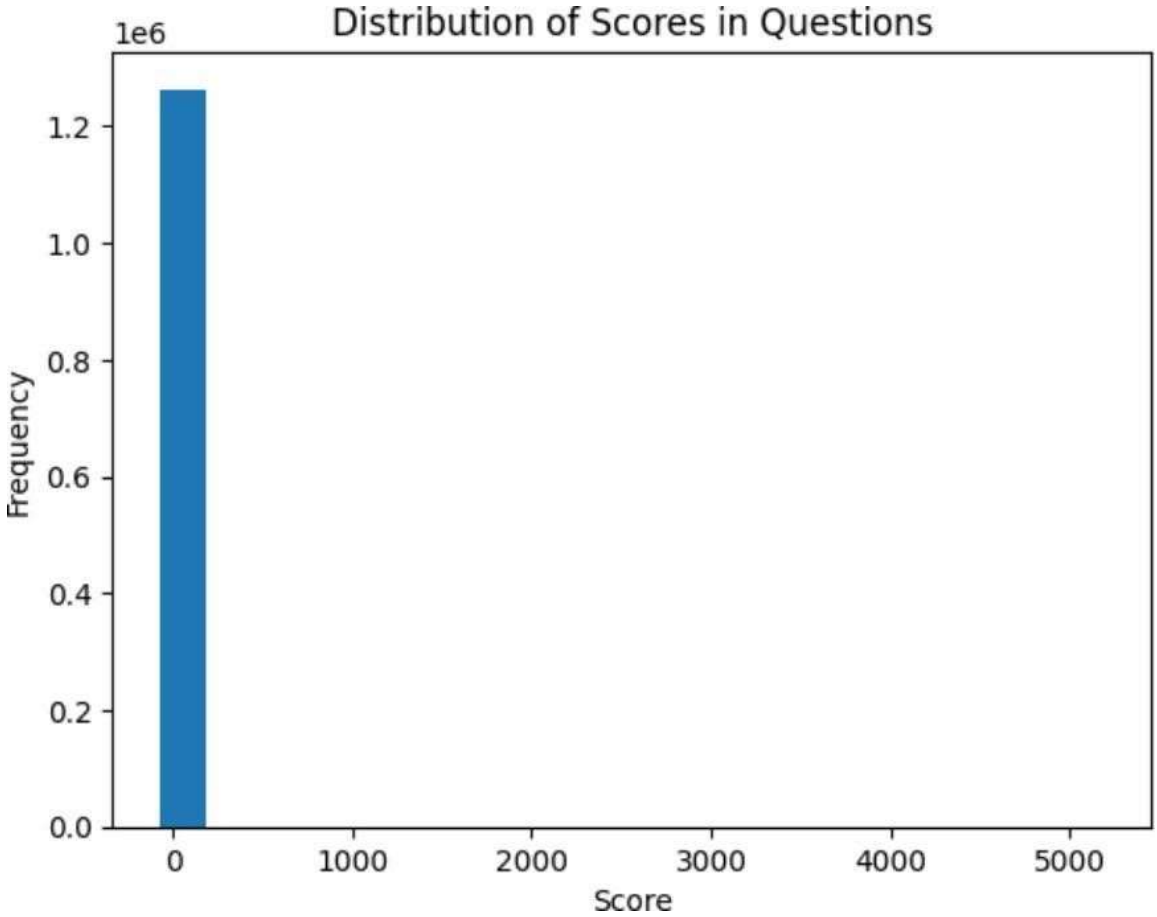
```
                 Id   OwnerUserId      ParentId         Score
count  2.014516e+06  2.001316e+06  2.014516e+06  2.014516e+06
mean   1.915490e+07  1.487613e+06  1.808390e+07  2.480563e+00
std    1.168713e+07  1.549051e+06  1.169152e+07  1.590938e+01
min    9.200000e+01  1.000000e+00  8.000000e+01 -4.200000e+01
25%    8.854490e+06  2.818680e+05  7.692900e+06  0.000000e+00
50%    1.866246e+07  9.546430e+05  1.712404e+07  1.000000e+00
75%    2.929756e+07  2.197072e+06  2.804977e+07  2.000000e+00
max    4.014339e+07  7.045028e+06  4.014319e+07  5.718000e+03
```

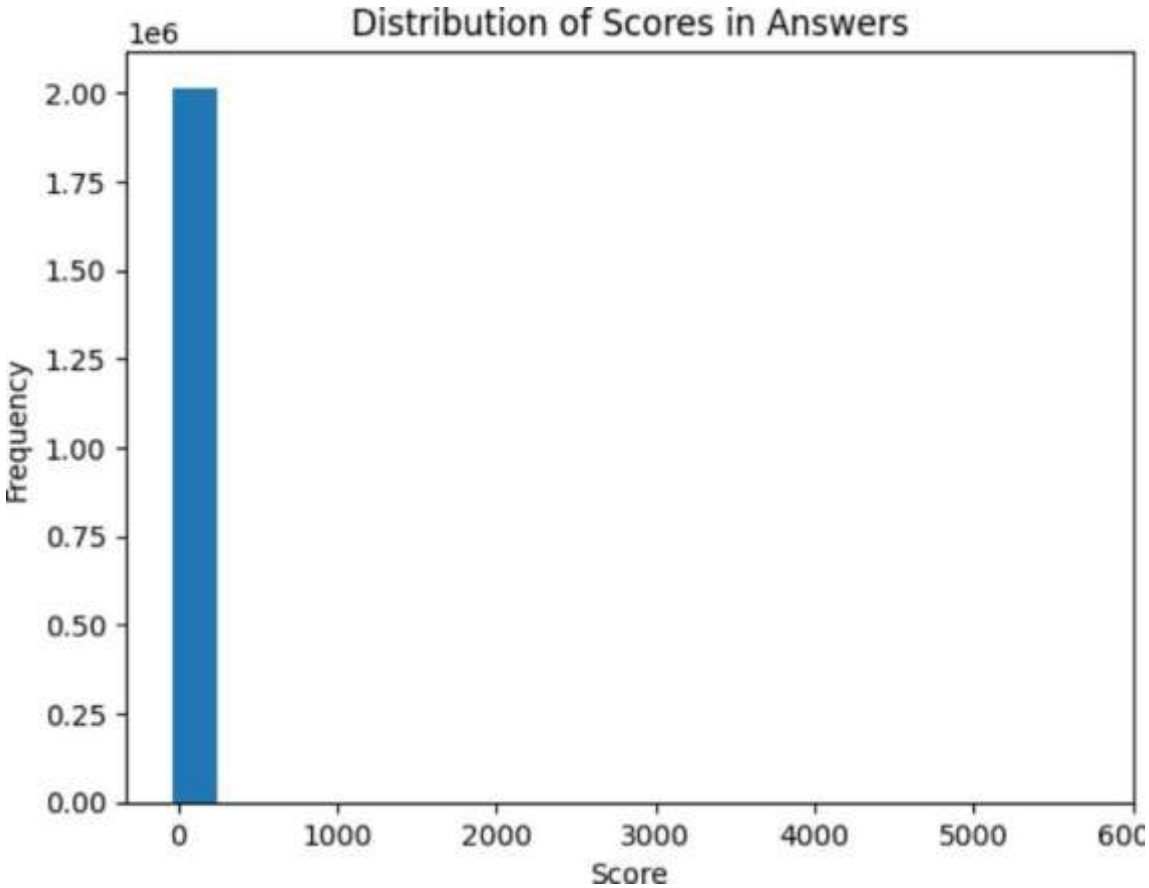
```
Tags statistics:
                 Id
count  3.750994e+06
mean   2.148285e+07
std    1.147246e+07
min    8.000000e+01
25%    1.164430e+07
50%    2.196248e+07
75%    3.164509e+07
max    4.014338e+07
Questions data types:
Id                int64
OwnerUserId     float64
CreationDate     object
ClosedDate       object
Score             int64
Title            object
Body             object
dtype: object

Answers data types:
Id                int64
OwnerUserId     float64
CreationDate     object
ParentId          int64
Score             int64
Body             object
dtype: object

Tags data types:
Id      int64
Tag    object
dtype: object
Questions missing values:
Id                    0
OwnerUserId       14454
CreationDate          0
ClosedDate      1208257
Score                 0
Title                 0
Body                  0
dtype: int64
```

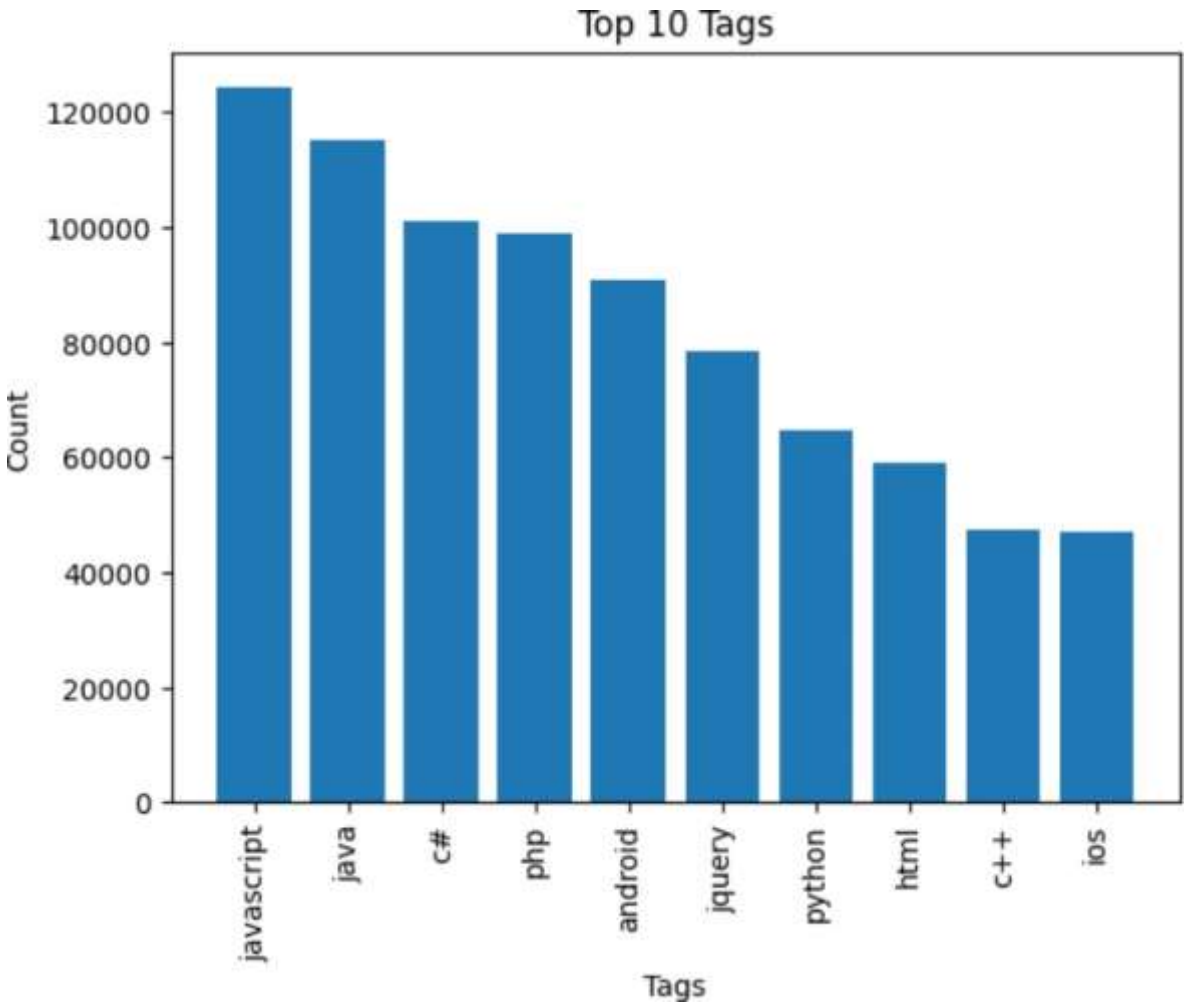
```
Answers missing values:
Id                   0
OwnerUserId      13200
```

[redacted]

```
CreationDate        0
ParentId            0
Score               0
Body                0
dtype: int64

Tags missing values:
Id        0
Tag    1113
dtype: int64
```




Distribution of Scores in Answers
1e6
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Frequency
0
1000
2000
3000
4000
5000
600
Score



IN 27: no_topics=20

text=new_df['Body']

vectorizer_train=TfidfVectorizer(analyzer='word',min_df=0.0,max_df=2.0,strip_accent

TF_IDF_matrix=vectorizer_train.fit_transform(text)

lda=LatentDirichletAllocation(no_topics,max_iter=5,learniing_method='online')

IN 28: def display_topics(model,feature_names,no_top_words);

for topic_idx, topic in enumerate(model,components_);

print(". ..... ") print("Topic %d:"%(topic_idx))

print("",join([feature_names[i] for i in topic,argsort()[:-no_top_words-1:-1]]))

print(". ..... ") no_top_words=10

display_topics(Ida,vectorizer_train_train.get_feature_names(),no_top_words)

IN 29: x1=new_df['Body']

x2=new_df['Title']

y=new_df['Tags']

IN 30: multilabel_binarizer=MultiLabelBinazer() =multilabel binazier.fit_transform(y)

IN 31: vectorizer_x1=TfidfVectorizer(analyzer='word',min_df=0.0,max_df=1.0,strip_accents=N

vectorizer_x2=TfidfVectorizer(analyzer='word',min_df=0.0,max_df=1.0,strip_accents=N

x1_tfidf=vectorizer_x1.fit_transform(x1)

x2_tfidf=vectorizer_x2.fit_transform(x2)

x_tfidf=hstack([x1_tfidf,x2_tfidf])

IN 32: X1_tfidf = vectorizer_X1.fit_transform(X1)

X2_tfidf = vectorizer_X2.fit_transform(X1)

IN 33: sgd = SGDClassifier() lr

= LogisticRegression()

mn = MultinomialNB() svc =

LinearSVC() prec_dict = {}

hamloss_dict = {} for classi in [sgd,

lr, mn, svc]: clf =

OneVsRestClassifier(classi)

clf.fit(X_train, y_train) y_pred =

clf.predict(X_test) ham =

hamming_loss(y_test, y_pred)

prec = precision_score(y_test, y_pred, average='weighted')

clsnam = classi. class. Name prec_dict[clsnam]

= ham

print('Classifier: ', clsnam) print("Hamming

Loss: ", ham) print('Precision: ', prec)

print('Recall: ', recall_score(y_test, y_pred, average='weighted')) **SAVING THE BEST MODEL:**

IN 34: # Exporting Model import

joblib

joblib_file = "tagpredictor.pkl" joblib.dump(clf,

joblib_file)

Load from file tagPredictorModel =

joblib.load('tagPredictor.pkl')

RESULTS AND SCREENSHOTS:

```
finalapp.py - Autonomous-Tagging-of-Stack-Overflow-Questions-Project-main - Visual Studio Code
finalapp.py
Autonomous-Tagging-of-Stack-Overflow-Questions-Project-main > finalapp.py > index
1   from flask import Flask, render_template, jsonify, request
2   from tag_predictor import getTags

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                                   Python Debug Console

PS C:\Users\samue\Desktop\Autonomous-Tagging-of-Stack-Overflow-Questions-Project-main>  & 'D:\Python1\python.exe' 'c:\Users\samue\.vs
code\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '60270' '--' 'C:\Users\samu
e\Desktop\Autonomous-Tagging-of-Stack-Overflow-Questions-Project-main\Autonomous-Tagging-of-Stack-Overflow-Questions-Project-main\fin
alapp.py'
 * Serving Flask app 'finalapp'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

```
---
Clf:  MLPClassifier
Jacard score: 48.421945317623646
Hamming loss: 1.3439924014563875
---
```

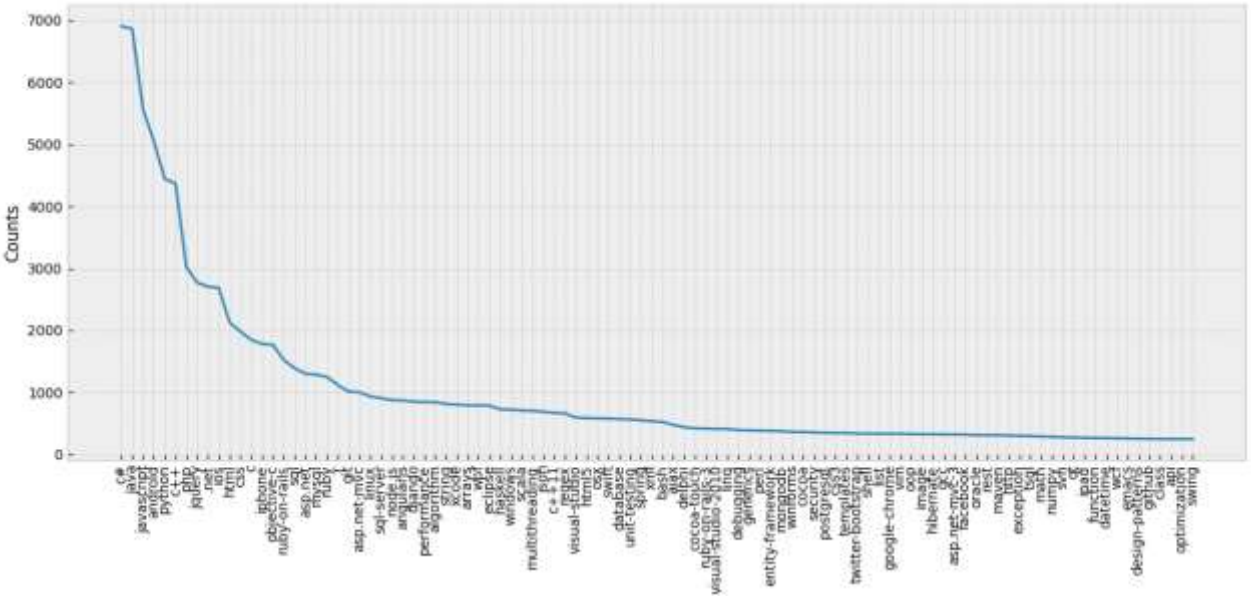
```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                                   Python Debug Console

use would like find know question one code way time
--------------------------------------------
Clf:  DummyClassifier
Jacard score: 0.0
Hamming loss: 1.5717112553427262
---
Clf:  SGDClassifier
Jacard score: 47.71581974565986
Hamming loss: 0.9547253443090074
---
Clf:  LogisticRegression
Jacard score: 48.31842646826824
Hamming loss: 0.971188855469368 4
---
Clf:  MultinomialNB
Jacard score: 40.27806674355669
Hamming loss: 1.1754788665505778
---
Clf:  LinearSVC
Jacard score: 52.31847923592423
Hamming loss: 0.9556751622605667
---
Clf:  Perceptron
Jacard score: 46.875824494749615
Hamming loss: 1.43351274339085
---
Clf:  PassiveAggressiveClassifier
Jacard score: 49.42296674883346
Hamming loss: 1.2666613899002692
---
```

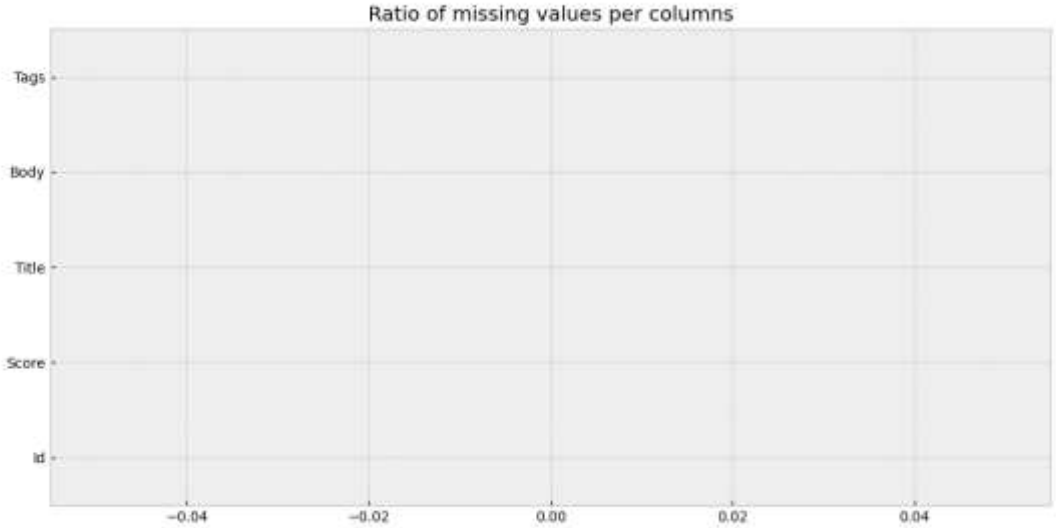
```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                                   Python Debug Console

--------------------------------------------
Topic 15:
ios difference xcode matrix notification config explain computer android please
--------------------------------------------
--------------------------------------------
Topic 16:
view image controller model fragment use frame mvc draw scope
--------------------------------------------
--------------------------------------------
Topic 17:
node rail gem ruby child parent profile 19 attribute schema
--------------------------------------------
--------------------------------------------
Topic 18:
test li unit ul selector class virtual framework none setup
--------------------------------------------
--------------------------------------------
Topic 19:
use would like find know question one code way time
--------------------------------------------
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL                                   Python Debug Console

Topic 7:
request server use error user get url file service app
--------------------------------------------
--------------------------------------------
Topic 8:
int std array function return foo string const value code
--------------------------------------------
--------------------------------------------
Topic 9:
error project file warn compile build -- library version fail
--------------------------------------------
--------------------------------------------
Topic 10:
php hello echo facebook world xs 64 equal equivalent 32
--------------------------------------------
--------------------------------------------
Topic 11:
table query select row id value column data sql database
--------------------------------------------
--------------------------------------------
Topic 12:
iphone enum app apple play mobile touch record ref sign
--------------------------------------------
--------------------------------------------
Topic 13:
div function button html var click page script class jquery
--------------------------------------------
--------------------------------------------
Topic 14:
string character line text file match print like list word
--------------------------------------------
```

```
--------------------------------------------
Topic 0:
android bean layoutwidth layoutheight textview spring item wrapcontent matchparent name
--------------------------------------------
--------------------------------------------
Topic 1:
file run command install use instal studio project directory import
--------------------------------------------
--------------------------------------------
Topic 2:
public class void new string object return method private static
--------------------------------------------
--------------------------------------------
Topic 3:
thread exception task video catch process block device queue run
--------------------------------------------
--------------------------------------------
Topic 4:
git width height branch commit plot image size position div
--------------------------------------------
--------------------------------------------
Topic 5:
activity intent alloc bundle nslog person graph savedinstancestate eandroidruntime android
--------------------------------------------
--------------------------------------------
Topic 6:
124 date 00 12 11 14 15 datetime 10 13
--------------------------------------------
--------------------------------------------
```

```
dtypes: float64(1), int64(2), object(4)
memory usage: 67.5+ MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3750994 entries, 0 to 3750993
Data columns (total 2 columns):
 #   Column  Dtype
---  ------  -----
 0   Id      int64
 1   Tag     object
dtypes: int64(1), object(1)
memory usage: 57.2+ MB
Dupplicate entries: 0
```

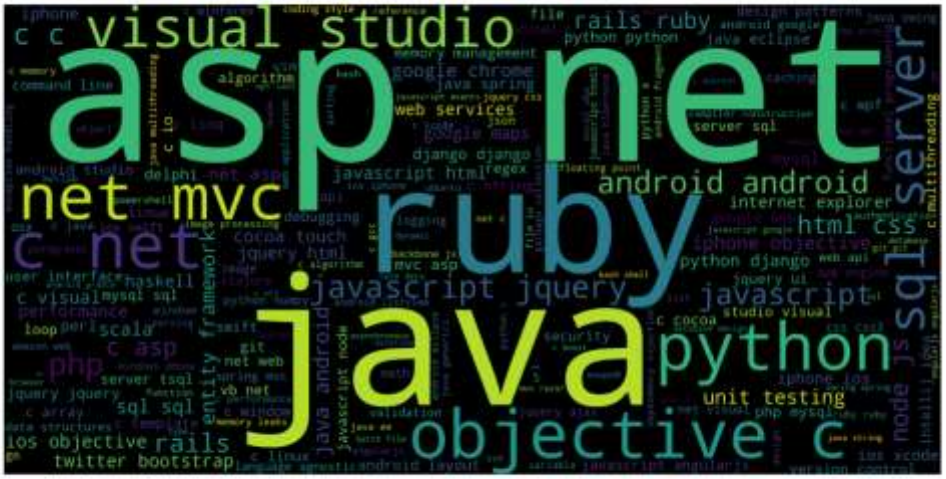


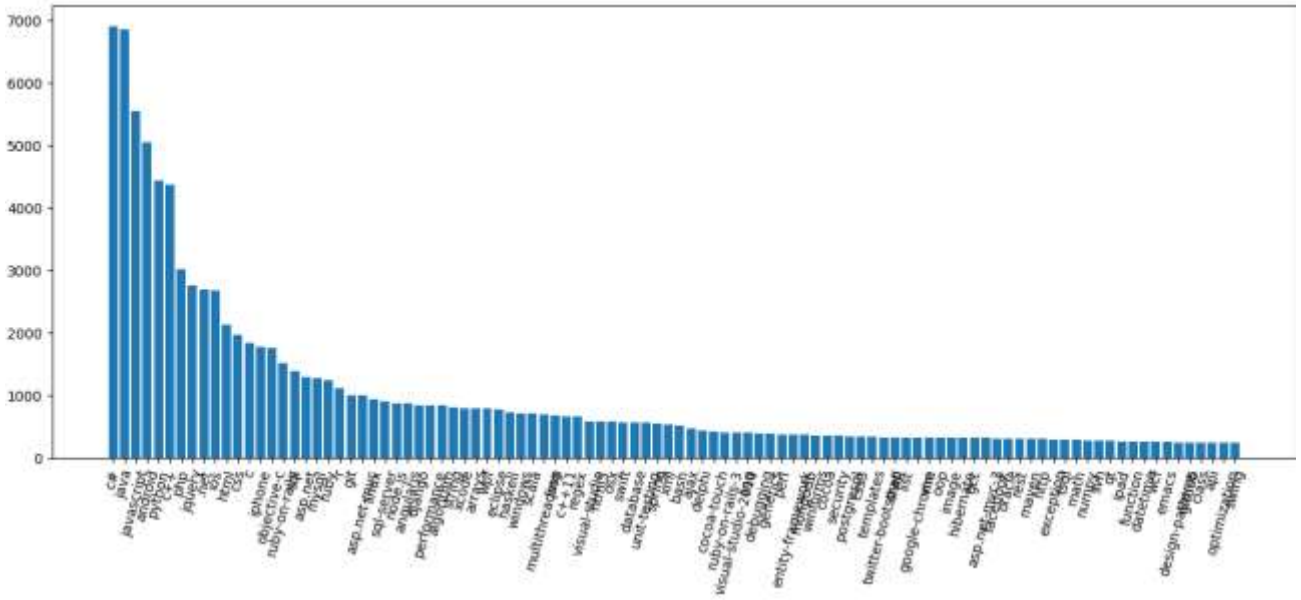



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3750994 entries, 0 to 3750993
Data columns (total 2 columns):
 #   Column  Dtype
---  ------  -----
 0   Id      int64
 1   Tag     object
dtypes: int64(1), object(1)
memory usage: 57.2+ MB
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL                                   Python Debug Console

PS C:\Users\samue\Desktop\Autonomous-Tagging-of-Stack-Overflow-Questions-Project-main\Autonomous-Tagging-of-Stack-Overflow-Questions-Projec
t-main>  & 'D:\Python1\python.exe' 'c:\Users\samue\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter/../
..\debugpy\launcher' '59538' '--' 'C:\Users\samue\Desktop\Autonomous-Tagging-of-Stack-Overflow-Questions-Project-main\Autonomous-Tagging-of
-Stack-Overflow-Questions-Project-main\predicting-tags.py'
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1264216 entries, 0 to 1264215
Data columns (total 7 columns):
 #   Column        Non-Null Count    Dtype
---  ------        --------------    -----
 0   Id            1264216 non-null  int64
 1   OwnerUserId   1249762 non-null  float64
 2   CreationDate  1264216 non-null  object
 3   ClosedDate    55959 non-null    object
 4   Score         1264216 non-null  int64
 5   Title         1264216 non-null  object
 6   Body          1264216 non-null  object
dtypes: float64(1), int64(2), object(4)
memory usage: 67.5+ MB
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL                                   Python Debug Console

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\samue\Downloads\Autonomous-Tagging-of-Stack-Overflow-Questions-Project-main>  & 'D:\Python1\python.exe' 'c:\Users\samue\.vscode
\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '59364' '--' 'C:\Users\samue\Download
s\Autonomous-Tagging-of-Stack-Overflow-Questions-Project-main\model.py'
CLF:  SGDClassifier
Jaccard score: 32.10384676270381
------
CLF:  LogisticRegression
Jaccard score: 34.908580022162425
------
CLF:  LinearSVC
Jaccard score: 46.17210173605614
------
PS C:\Users\samue\Downloads\Autonomous-Tagging-of-Stack-Overflow-Questions-Project-main>
```


asp net
visual studio
ruby
java
python
net mvc
c net
objective c
javascript jquery
javascript
android android
sql server
unit testing
rails
twitter bootstrap
ios objective
scala
c asp
entity framework
google chrome
web services
rails ruby
python python
iphone objective
python django
internet explorer
html css
cocoa touch
jquery html
mysql sql
haskell
node js
iphone ios
studio visual
jquery jquery
sql sql
server tsql
net web
vb net
delphi
regex
loop
performance
swift
git
mvc asp
jquery ui
web api
python python
java eclipse
memory management
command line
algorithm
debugging
image processing
security
validation
memory leaks
data structures
c linux
ios xcode
version control
design patterns
file
reference
coding style
iphone
c io
c++11
visual-studio-2010
ruby-on-rails-3
cocoa-touch
objective-c
asp.net-mvc
sql-server
regex
arrays
windows
perfomance
algorithm
multithreading
database
unit-testing
bash
delphi
security
templates
twitter-bootstrap
google-chrome
image
hibernate
asp.net-mvc-3
mvc
exceptions
math
numpy
function
datetime
emacs
design-patterns
optimization


7000
6000
5000
4000
3000
2000
1000
0
c#
java
javascript
android
python
jquery
php
ios
iphone
objective-c
ruby-on-rails
asp.net
mysql
sql
linux
git
asp.net-mvc
sql-server
regex
arrays
perfomance
algorithm
multithreading
c++11
visual-studio
database
unit-testing
bash
delphi
cocoa-touch
ruby-on-rails-3
visual-studio-2010
entity-framework
twitter-bootstrap
google-chrome
image
hibernate
asp.net-mvc-3
mvc
exceptions
math
numpy
function
datetime
emacs
design-patterns
optimization


Tag Predictor
what is python?
Submit
Tags: python

GITHUB LINK:

smartinternz02/SBSPS-Challenge-9880-Autonomous-Tagging-Of-Stack-Overflow-Questions: Autonomous-Tagging-Of-Stack-Overflow-Questions (github.com) (https://github.com/smartinternz02/SBSPS-Challenge-9880-Autonomous-Tagging-Of-Stack-Overflow-Questions)

GITHUB ID: SBSPS-Challenge-9880-Autonomous-Tagging-Of-Stack-Overflow-Questions

DEMO VIDEO LINK

SMARTINTERNZ Autonomous Tagging Of Stack Overflow Questions IBM HACK CHALLENGE 2023 - Data Science - YouTube

(https://www.youtube.com/watch?v=2G6r6T91jek)