**Prediction of the quality rating on stack overflow questions**

**Major Project Report**

Submitted in partial fulfillment of the requirements

for the degree of

**Bachelor of Engineering (Computer Engineering)**

by:

|  |  |
| --- | --- |
| Ajinkya Thakur | TU3F1819123 |
| Mohammad Amaan | TU3F1819118 |
| Anisha Gupta | TU3F1819122 |
| Saiganesh Jeyapandi | TU3F1819120 |

Under the Guidance of

Dr. Shaveta Malik



**Department of Computer Engineering**

**TERNA ENGINEERING COLLEGE**

Nerul (W), Navi Mumbai 400706

(University of Mumbai)

(2021-2022)

****

**TERNA ENGINEERING COLLEGE, NERUL**

**Department of Computer Engineering**

Academic Year 2021-22

**CERTIFICATE**

This is to certify that the project entitled **“Prediction of the quality rating on stack overflow questions”** is a bonafide work of

|  |  |
| --- | --- |
| Ajinkya Thakur | TU3F1819123 |
| Mohammad Amaan | TU3F1819118 |
| Anisha Gupta | TU3F1819122 |
| Saiganesh Jeyapandi | TU3F1819120 |

submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the Bachelor of Engineering (Computer Engineering).

**Guide Project Convener Head of Department Principal**

Project Report Approval

This Project Report – A entitled “**Prediction of the quality rating on stack overflow questions**” by following students is approved for the degree of ***B.E. in "Computer Engineering"***.

***Submitted by:***

|  |  |
| --- | --- |
| Ajinkya Thakur | TU3F1819123 |
| Mohammad Amaan | TU3F1819118 |
| Anisha Gupta | TU3F1819122 |
| Saiganesh Jeyapandi | TU3F1819120 |

Examiners Name & Signature:

1.---------------------------------------------------------

2.----------------------------------------------------------

Date:

Place:

**Declaration**

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will cause disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

|  |  |
| --- | --- |
| Ajinkya Thakur | TU3F1819123 |
| Mohammad Amaan | TU3F1819118 |
| Anisha Gupta | TU3F1819122 |
| Saiganesh Jeyapandi | TU3F1819120 |

Date:

Place:

**Acknowledgment**

We would like to express our sincere gratitude towards my guide **Dr. Shaveta Malik**, for the help, guidance, and encouragement they provided during the Progress seminar. This work would have not been possible without their valuable time, patience and motivation. We thank them for making my stint thoroughly pleasant and enriching. It was a great learning and an honor being their student.

We are deeply thankful to **Prof. Archana Mire** (H.O.D Computer Department), and the entire team in the Computer Department. They supported us with scientific guidance, advice, and encouragement, they were always helpful and enthusiastic and this inspired us in our work.

We take the privilege to express our sincere thanks to **Dr. L. K. Ragha**, our principal**,** for providing encouragement and much support throughout our work.

.

|  |  |
| --- | --- |
| Ajinkya Thakur | TU3F1819123 |
| Amaan Mohammad | TU3F1819118 |
| Anisha Gupta | TU3F1819122 |
| Saiganesh Jeyapandi | TU3F1819120 |

**Abstract**

The mass local area of developers use entries like stackoverflow to tackle their inquiries. It tends to be considered as a gathering point of experienced computer engineers and the individuals who are searching for an answer for a particular issue.

To observe an answer we want a decent quality rating question. As in excess of 8,000 posts show up on Stack Overflow consistently, the viable programmed sifting of them is fundamental.

To save the professionality and nature of Q&A discussions, members should adhere to the guidelines laid out by the local area. One such principle decision is that the inquiries posted on the gateway ought to be applicable, unambiguous, and conceivable. This implies that each question being submitted onto Stack Overflow must be connected either to explicit advancement issues or strategies like utilizing a specific API, algorithmic issues, apparatuses or system. Questions that can't be addressed properly and directly, e.g., open-finished, off-subject, or loquacious inquiries, which might decrease the nature of the entryway, ought not to be upheld. The primary reason for Stack Overflow is to address distinct specialized issues, subsequently, abstract posts or questions raising ceaseless, assessment related conversations ought to likewise be kept away from. Likewise, questions should be planned in a basic and unequivocal manner with the goal that their motivation is clear to the expected respondents. Questions that don't meet the above measures will be shut down or, sometimes, even erased from the site by mediators or experienced individuals with recognized honors. Simultaneously, great inquiries and answers are compensated by the local area by upvoting their score, which, subsequently, expands the client notoriety of the inquiry banner or the respondent. In the end, at a specific standing level, selective freedoms will be conceded to these effectively contributing clients.

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Sr.no** | **Figures** | **Page no.** |
| 4.1 | Flowchart diagram | 7 |
| 4.2 | Input of question | 8 |
| 4.3 | Quality of question | 8 |
| 7.1 | Dataset | 20 |
| 7.2 | Visualized Dataset | 21 |
| 8.1 | Gantt Chart | 22 |
| 9.1 | Confusion Matrix ( Test Case-1 ) | 23 |
| 9.2 | Confusion Matrix ( Test Case-2 ) | 24 |

**List of abbreviation**

|  |  |
| --- | --- |
| SVM | Support Vector Machine |
| TF-IDF | Term frequency - Inverse document frequency |
| tfi,j | Number of occurrence of i in j |
| dfi | Number of documents containing i |
| n | Total number of documents |

**Table of Contents**

|  |  |  |
| --- | --- | --- |
|  | Abstract  List of figures  List of abbreviation |  |
| Chapter 1 | Introduction | 1 |
|  | 1.1 Aim and Objectives of Project | 2 |
|  | 1.2 Motivation | 2 |
|  | 1.3 Organization Of The Report | 3 |
| Chapter 2 | Literature Survey | 4 |
|  | 2.1 Existing System | 4 |
|  | 2.2 Problem statement | 5 |
|  | 2.3 Scope of project | 5 |
| Chapter 3 | Proposed System | 6 |
|  | 3.1 Software and hardware requirements | 6 |
|  | 3.2 Software components | 6 |
| Chapter 4 | Design | 7 |
|  | 4.1 Flowchart diagram | 7 |
| Chapter 5 | Methodology | 9 |
|  | 5.1 Dataset | 8 |
|  | 5.2 Data preprocessing | 9 |
|  | 5.3 Feasibility and risk analysis | 12 |
| Chapter 6 | Implementation | 14 |
|  | 6.1 Working of the system | 14 |
| Chapter 7 | Performance evaluation | 20 |
|  | 7.1 Description of data | 21 |
|  | 7.2 Experimental setup | 22 |
|  | 7.3 Experiment result | 23 |
| Chapter 8 | Timeline | 24 |
|  | 8.1 Gantt chart | 24 |
| Chapter 9 | Result | 25 |
|  | Conclusion | 27 |
|  | References | 27 |

**Chapter 1**

**Introduction**

Stack Overflow is a [question and answer website](https://en.wikipedia.org/wiki/Question-and-answer_site) for professional and enthusiast programmers. It is the flagship site of the [Stack Exchange Network](https://en.wikipedia.org/wiki/Stack_Exchange).It was created to be a more open alternative to earlier question and answer websites such as [Experts-Exchange](https://en.wikipedia.org/wiki/Experts-Exchange). The website serves as a platform for users to ask and answer questions, and, through membership and active participation, to vote questions and answers up or down similar to [Reddit](https://en.wikipedia.org/wiki/Reddit) and edit questions and answers in a fashion similar to a [wiki](https://en.wikipedia.org/wiki/Wiki). Users of Stack Overflow can earn [reputation points](https://en.wikipedia.org/wiki/Reputation_system) and "badges"; for example, a person is awarded 10 reputation points for receiving an "up" vote on a question or an answer to a question, and can receive badges for their valued contributions, which represents a [gamification](https://en.wikipedia.org/wiki/Gamification) of the traditional Q&A website. Users unlock new privileges with an increase in reputation like the ability to vote, comment, and even edit other people's posts**.**

Throughout the course of recent years, online inquiry responding to (Q&A) discussions, for example, Stack Overflow have turned into the fundamental archives of information for programmers. Since the data accumulated on the entry is attached to proficient inquiries begun from specialists or specialist software engineers, these posts give a viable and furthermore essentially pertinent help to both amateur and experienced experts. The primary motivation behind Stack Overflow is to tackle obvious specialized issues, consequently, emotional posts or questions raising endless, assessment related conversations ought to likewise be stayed away from. Likewise, questions should be figured out in a straightforward and express manner with the goal that their motivation is clear to the possible respondents. Questions that don't meet the above standards will be shut or, now and again, even erased from the site by mediators or experienced individuals with recognized honors.

The focal point of our current review is the quality-based order of inquiries transferred to Stack Overflow in light of their semantic attribute.

**1.1 Aim and Objectives of Project :**

Individuals have real issues that they can't sort out all alone. In the wake of hanging tight for a broad time frame period they lose trust and become mixed up in the actual documentation. To keep away from this, clients ought to rehash questions and convert low quality inquiries into a decent quality inquiry that helps the client as well as different clients that have a go at looking through a similar inquiry at some point later from here on out. However, there is certifiably not a measurement on Stack Overflow that proposes whether your inquiries are low quality or superior grade.

**1.2 Motivation:**

As the Covid pandemic is going on, the number of individuals learning all alone has expanded because of shutting schools and colleges. Hence a decent Q&A gathering is a must, to accomplish that great quality inquiries ought to be posted as individuals noting the questions on the Q&A discussion are restricted.

As everybody doesn't rehash their inquiries prior to presenting it on gathering on improving the nature of the inquiry as it will make sense of the uncertainty very well, But it isn't the genuine situation not every person put a lot of exertion on outlining a decent quality question, hence it is important to make a framework to confirm the nature of the inquiry to upgrade the general situation of the discussion itself. which will draw in additional individuals to contribute their insight and in the end settle the inquiries or inquiries of different clients.

So, we center around creating a framework that can foresee the nature of the given inquiry before posting it on the discussion which in the long run prompts a superior arrangement of inquiries getting posted on the forum. Hence having less burden on the response and the actual gathering.

**1.3 Organization of report**

**Chapter 1:**  Brief overview about the introduction and aims for developing this project.

**Chapter 2:**  The report includes the literature survey on the existing system.

**Chapter 3:** Shows the analysis of the project and software requirement system.

**Chapter 4:**  Diagrammatical design of our project

**Chapter 5:**  Shows the methodology, the problem definition tells us about the expected outcome

**Chapter 6:**  Working of the system

**Chapter 7:**  Performance evaluation of the system

**Chapter 8:**  Timeline of the project compilation

**Chapter 9:**  The conclusion and references of this project

**Chapter 2**

**Literature Survey**

**2.1 Existing system :**

It presents a novel approach for classifying questions based exclusively on their linguistic and semantic features using deep learning methods. Their binary classiﬁer relying on the textual properties of posts can predict whether the question is to be closed with an accuracy of 74% similar to the results of previous metrics-based models. In accordance with our ﬁndings they have concluded that by combining deep learning and natural language processing methods, the maintenance of quality at Q&A forums could be supported using only the raw text of posts. [1]

The objective of the research is to ease the tagging of questions on Stack Overflow. This work proposes TagStack system, a machine learning and feedback-based framework for predicting tags on Stack Overflow. It perform experiment on real world and publicly available dataset, and results shows that TagStack system is effective in predicting tags on Stack Overflow. [2]

Community Question Answering websites (CQA) have growing popularity as a way of providing and searching for information. CQA attracts users as they provide a direct and rapid way to find the desired information. As recognizing good questions can improve the CQA services and the user’s experience, the current study focuses on question quality instead. t. The influence of the question tags, length of the question title and body, presence of a code snippet, the user reputation and terms used to formulate the question is tested. For each set of dependent variables, Ridge regression models are estimated. The results indicate that the inclusion of terms in the models improves their predictive power Additionally, they investigate which lexical terms determine high and low quality questions. The terms with the highest and lowest coefficients are semantically analyzed. The analysis shows that terms predicting high quality are terms expressing, among others, excitement, negative experience or terms regarding exceptions. Terms predicting low quality questions are terms containing spelling errors or indicating off-topic questions and interjections. [3]

**2.2 Problem Statement**

Individuals have genuine inquiries that they can find out for themselves. After a huge delay they lose trust and can't track down any arrangement in the given documentation. To stay away from this, clients should re-read the inquiries and transform low-level questions into a decent inquiry that helps the client as well as different clients who are attempting to look for a similar inquiry later on. Notwithstanding, there are no measurements in Stack Overflow proposing that your inquiries are of low quality or top caliber.

**2.3 Scope of Project :**

By applying this technique to order the inquiries, stack overflow will actually want to approve the nature of the inquiry prior to posting it on the gathering and to see whether it ought to be altered or shut. A robotized method for dissecting the separated terms would be an improvement and a decent idea for future exploration.

**Chapter 3**

**Proposed system**

**3.1 Software and Hardware requirement**

For programming we have involved python language as it supports language to assemble control and the board, testing and numerous other required offices.To carry out programs we utilized jupyter notebook to make and share reports that contain live code, conditions, perceptions, and text.

For equipment the least prerequisite is 8gb ram and i5 processor to easily run the program.

**3.2 Software components**

* Jupyter notebook
* Python libraries
* Support vector machine
* Term frequency-inverse document frequency

**Chapter 4**

**Design**

**4.1 Flowchart diagram**

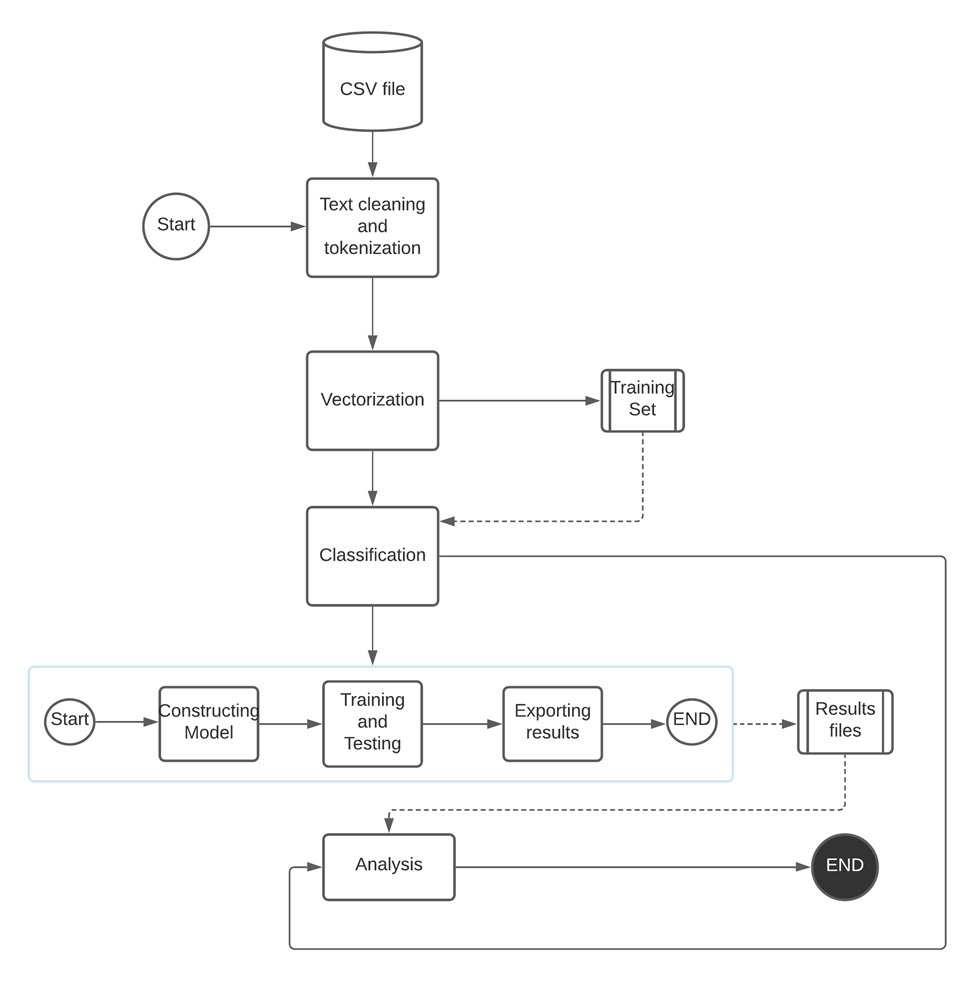


Figure 4.1 Flowchart

**4.2 User Interface of the model**

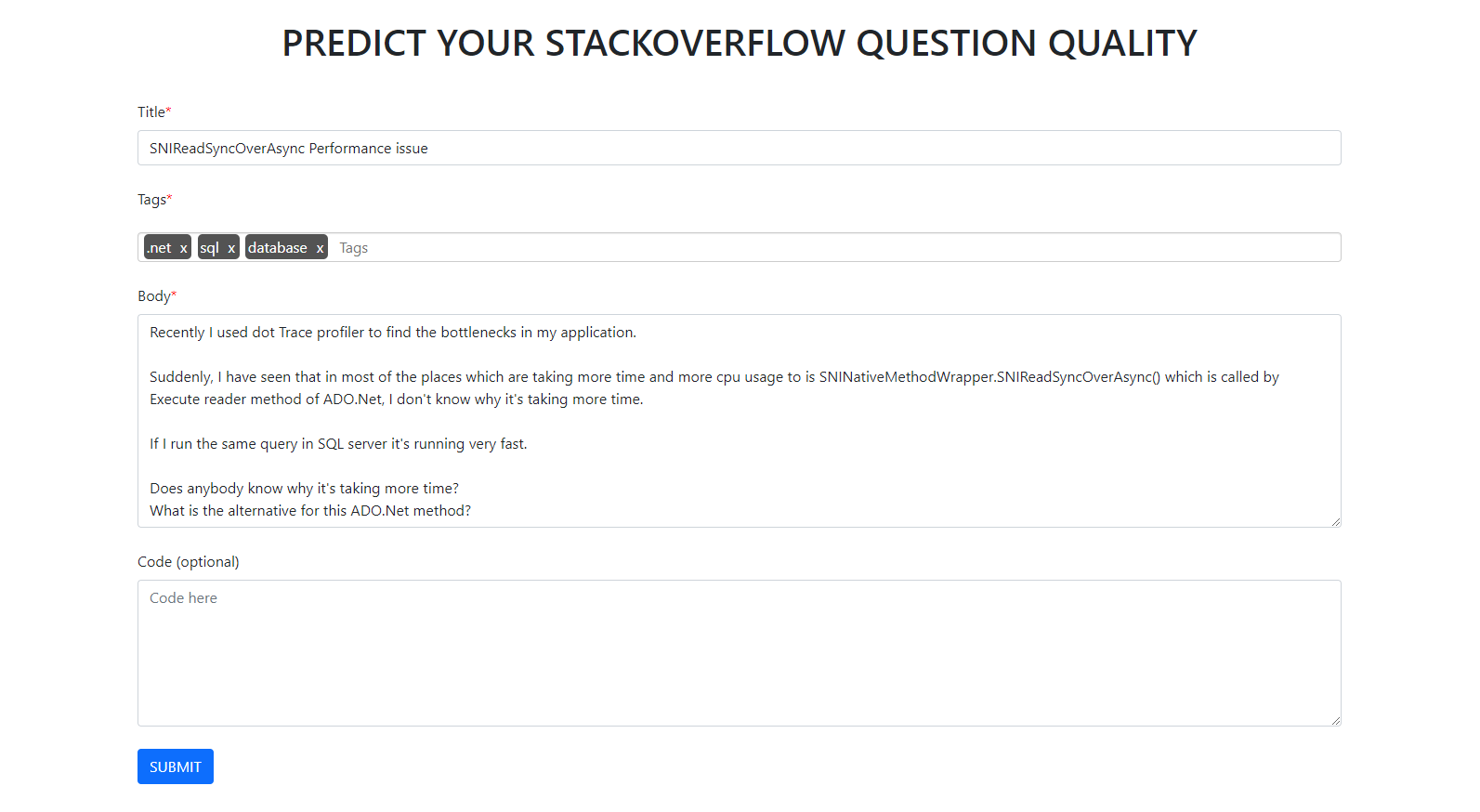


Figure 4.2 Input of Question

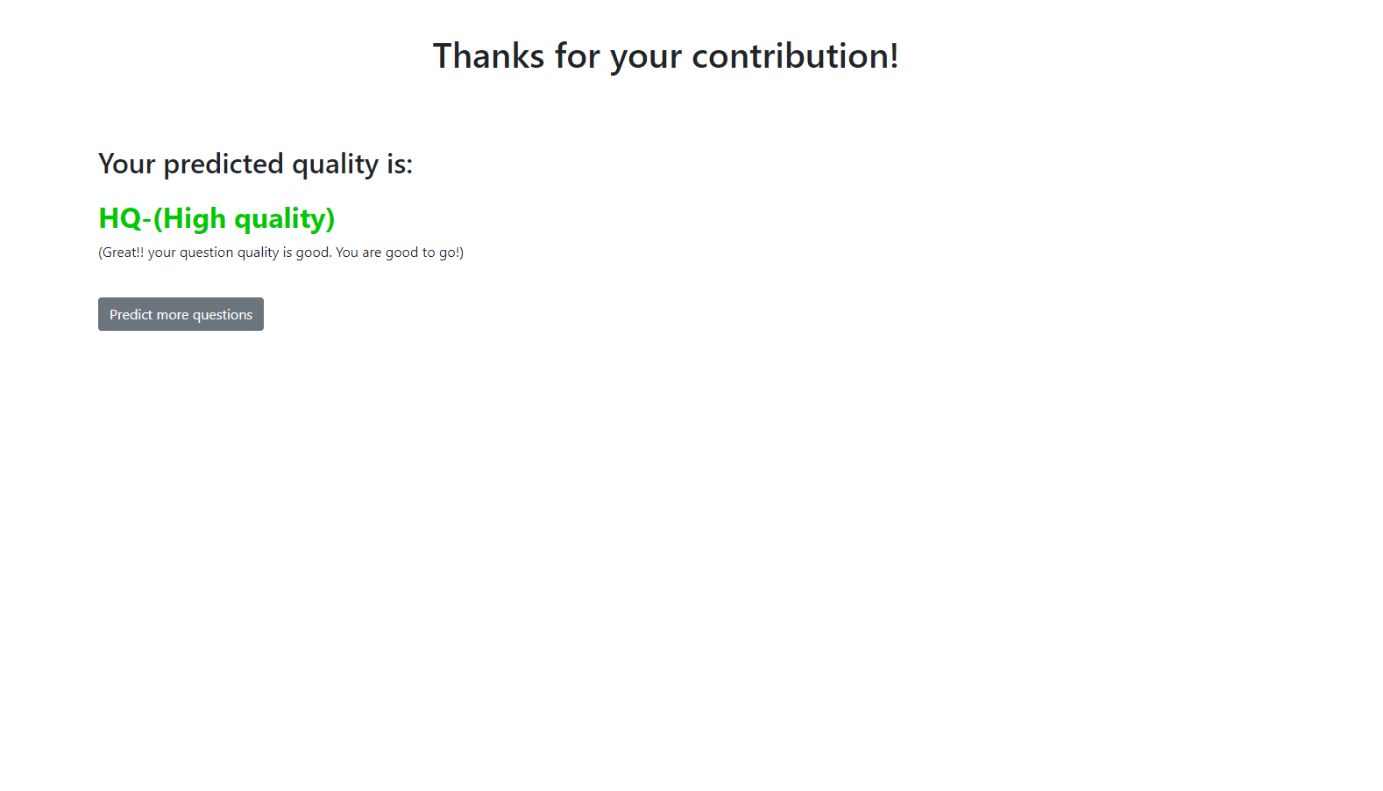


Figure 4.3 Quality of question

**Chapter 5**

**Methodology**

**5.1 Dataset**

Our dataset consists of 60,000 rows i.e unique questions. Ranging from year 2016-2021.Different type of information about the questions represented in the form of various columns which are:

* **Question id**: Every question has its unique id to identify the serial number of the question. Questions are according to the question id
* **Title**: It is a Multiview approach to suggest moderation action in community question answering sites. Basically it is the title of question
* **Body**: Body is a paragraph format answer to the question
* **Tags**: A tag is a word or expression that portrays the subject of the inquiry. It is a method for associating specialists with questions they will actually want to respond to by arranging inquiries into explicit, obvious classifications. Tags can likewise be utilized to assist you with distinguishing questions that are fascinating or pertinent to you.
* **Creation date**: In this column it will show the date and time when the question was published. All dates are in UTC format and with an accurate time.
* **Question quality**: Its shows whether the question:

HQ: High-quality posts without a single edit.

LQ\_EDIT: Low-quality posts with a negative score, and multiple community edits. However, they still remain open after those changes.

LQ\_CLOSE: Low-quality posts that were closed by the community without a single edit.

**5.2 Data preprocessing**

* **Data cleaning**

Data cleaning is the most common way of fixing or eliminating inaccurate, adulterated, erroneously organized, copy, or deficient information inside a dataset. While consolidating different information sources, there are numerous amazing open doors for information to be copied or mislabeled.

Assuming information is mistaken, results and calculations are untrustworthy, despite the fact that they might look right. There is no outright method for recommending the specific strides in data process on the grounds that the cycles will fluctuate from dataset to dataset. However, it is significant to lay out a format for your Data cleaning process, so that it corrects the way without fail.

* **Remove punctuation**

Disposing of pointless pieces of the information, or noise, by switching all characters over completely to lowercase, eliminating punctuation marks, and grammatical mistakes.

Eliminating noise proves to be useful when you need to do message investigation on bits of information like remarks or tweets. The code in the accompanying segments will be useful to dispose of the text that disrupts text analysis.

* **Tokenize**

Tokenizing resembles parting an entire sentence into words. You can think about a basic separator for this reason. Be that as it may, a separator will neglect to divide the contractions isolated by "." or exceptional characters, as U.A.R.T., For instance. Challenges increase when more dialects are incorporated. What about managing compound words in dialects like German or French?

The greater part of these issues can be settled by utilizing the nltk library. The word\_tokenize module breaks the words into tokens and these words go about as a contribution for the standardization and cleaning process. It can additionally be utilized to change over a string (text) into numeric information so that AI models can process it.

* **Remove stopwords**

"Stop words" are an accessible overflow in any human language. By eliminating these words, we eliminate the low-level data from our text to give more clarity of mind to the significant data. All together words, we can say that the expulsion of such words shows no adverse results on the model we train for our undertaking.

Expulsion of stop words certainly decreases the dataset size and consequently lessens the preparation time because of the less number of tokens engaged with the preparation

* **Stemming**

Stemming is most certainly the less complex of the two methodologies. With stemming, words are diminished to their promise stems. A word stem need not be a similar root as a word reference based morphological root, it simply is an equivalent to or more modest type of the word.

Stemming algorithms are ordinarily rule-based. You can see them as heuristic interaction that kind of cuts off the closures of words. A word is checked out and goes through a progression of conditionals that decide how to chop it down.

For instance, we might have a postfix decide that, in view of a rundown of known additions, removes them. In the English language, we have postfixes like "- ed" and "- ing" which might be helpful to slice off to plan the words "cook," "cooking," and "cooked" all to a similar stem of "cook.”

* **Support vector machine**

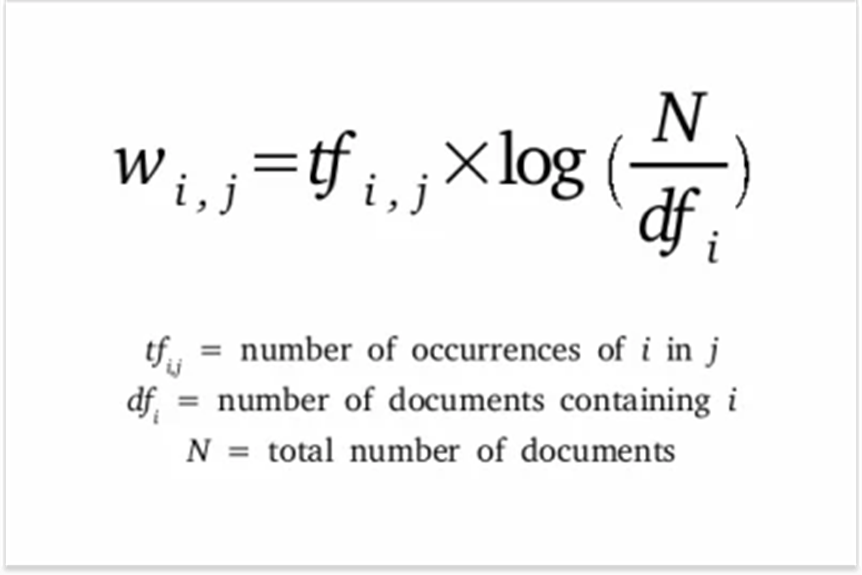
Support Vector Machine or SVM is one of the most famous Supervised Learning algorithms, which is utilized for Classification as well as Regression issues. In any case, principally, it is utilized for Classification issues in Machine Learning.The objective of the SVM algorithm is to make the best line or choice limit that can isolate n-dimensional space into classes so we can without much of a stretch put the new data of interest in the right classification later on. This best choice limit is known as a hyperplane.SVM picks the outrageous focuses/vectors that assist in making the hyperplane. These outrageous cases are called support vectors, and subsequently calculation is named as Support Vector Machine.

Support vector machine is another straightforward calculation that each machine learning master ought to have in his/her armory. Support vector machine is exceptionally liked by a larger number of people as it produces critical exactness with less calculation power. Support Vector Machine, truncated as SVM can be utilized for both relapse and characterization assignments. Yet, it is broadly utilized in order targets.

* **Tfidf**

TF-IDF means "Term Frequency — Inverse Document Frequency". This is a strategy to evaluate words in a bunch of archives. We for the most part process a score for each word to connote its significance in the record and corpus. This strategy is a broadly involved procedure in Information Retrieval and Text Mining. t is simpler for any programming language to figure out printed information as mathematical worth. Along these lines, consequently, we want to vectorize all of the text so it is better addressed. By vectorizing the reports we can additionally play out numerous undertakings like tracking down the important archives, positioning, bunching, and so forth.

The formula that is used to compute the tf-idf for a term t of a document d in a document set is tf-idf(t, d) = tf(t, d) \* idf(t), and the idf is computed as idf(t) = log [ n / df(t) ] + 1, where n is the total number of documents in the document set and df(t) is the document frequency of t; the document frequency is the number of documents in the document set that contain the term t.



**5.3 Feasibility and Risk analysis:**

* It includes exceptionally extended and muddled method of computations and analysis
* The complexity and size of the software is expanding step by step and regularly another programming language or apparatus is coming, Due to which it is important for programming specialists to refresh and broaden their ranges of abilities as indicated by the current programming improvement pattern.
* Since linear regression assumes a linear relationship between the input and output factors, it neglects to fit complex datasets appropriately
* If we consider a large dataset, time complexity will be higher while performing stemming during data cleaning process and memory limit is exceeded while vectorization.
* As we will be using the SVM algorithm, which won't be able to perform well with large datasets as the required training time is higher.
* Data might take some time to execute and to show accurate prediction
* As the support vector classifier works by putting data points above and below the classifying hyperplane there is no probabilistic explanation for the classification.

**Chapter 6**

**Implementation**

**6.1 Working of the system**

**Data cleaning**

As the data is in raw(HTML) format not cleaned as it is extracted from the stackoverflow site. Hence removing the html tags and cleaning the data is a must.

Here, function cleanhtml cleans the data by removing html tags with help of regex library.

|  |
| --- |
| def cleanHtml(row):  row=row.lower()  clean = re.compile('<.\*?>')  cleantext = re.sub(clean, '', row)  return cleantext |

**Remove punctuation**

It helps to get rid of unhelpful parts of the data, or noise, by converting all characters to lowercase, removing punctuations marks.

|  |
| --- |
| def remove\_punctuation(row):   no\_punc\_txt=[]  for ch in row:  if ch in string.punctuation:  no\_punc\_txt.append(' ')  else:  no\_punc\_txt.append(ch)    no\_punc\_txt="".join(no\_punc\_txt)    return no\_punc\_txt |

**Tokenize**

Tokenization is essentially splitting a phrase, sentence, paragraph, or an entire text document into smaller units, such as individual words or terms. Each of these smaller units are called tokens.

|  |
| --- |
| def tokenize(row):  row=row.split()  r=[]  for x in row:  try:  int(x)  except:  if len(x)<=10:  r.append(x)  return r |

**Remove stopwords**

Stop words are available in abundance in any human language. By removing these words, we remove the low-level information from our text in order to give more focus to the important information. In order words, we can say that the removal of such words does not show any negative consequences on the model we train for our task.

Removal of stop words definitely reduces the dataset size and thus reduces the training time due to the fewer number of tokens involved in the training.

|  |
| --- |
| stopwords=stopwords.words('english') def remove\_stopwords(row):  tokens=[]  for word in row:  if word not in stopwords:  tokens.append(word)    return tokens[:80] |

**Stemming**

Stemming is a natural language processing technique that lowers inflection in words to their root forms, hence aiding in the preprocessing of text, words, and documents for text normalization.

According to Wikipedia, inflection is the process through which a word is modified to communicate many grammatical categories, including tense, case, voice, aspect, person, number, gender, and mood. Thus, although a word may exist in several inflected forms, having multiple inflected forms inside the same text adds redundancy to the NLP process.

As a result, we employ stemming to reduce words to their basic form or stem, which may or may not be a legitimate word in the language.

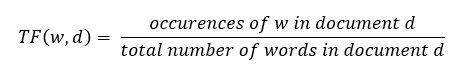
For instance, the stem of these three words, connections, connected, connects, is “connect”. On the other hand, the root of trouble, troubled, and troubles is “trouble,” which is not a recognized word.

|  |
| --- |
| p=PorterStemmer() def stemming(row):  stem\_lst=[p.stem(word) for word in row]  return stem\_lst |

**Tf-idf value calculation**

* **Term Frequency (TF)**

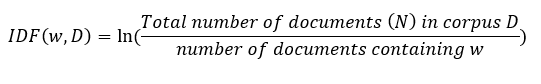
It is a measure of the frequency of a word (w) in a document (d). TF is defined as the ratio of a word’s occurrence in a document to the total number of words in a document. The denominator term in the formula is to normalize since all the corpus documents are of different lengths.

****

## Inverse Document Frequency (IDF)

It is the measure of the importance of a word. Term frequency (TF) does not consider the importance of words. Some words such as’ of’, ‘and’, etc. can be most frequently present but are of little significance. IDF provides weightage to each word based on its frequency in the corpus D.

IDF of a word (w) is defined as

****

**TF-IDF**: tf-idf(t, d) = tf(t, d) \* log(N/(df + 1))

|  |
| --- |
| def calculate\_tf\_idf(ds,alpha):  title=[]  body=[]  for t in ds['title']:  for word in t:  title.append(word)  title=set(title)  title=list(title)  for b in ds['body']:  for word in b:  body.append(word)   body=set(body)  body=list(body)    d={}  N=len(ds)  for i in range(N):  tokens=ds['title'][i]  for word in tokens:  try:  d[word].add(i)  except:  d[word]={i}  tokens=ds['body'][i]  for word in tokens:  try:  d[word].add(i)  except:  d[word]={i}   for i in d:  d[i]=len(d[i])  df= dict(sorted(d.items(), key = itemgetter(1), reverse = True)[:10000])  vocab=[i for i in df]    def freq(token):  c=0  if token in df:  return df[token]  return c   tf\_idf = {}  for i in range(N):   tokens = ds['body'][i]   counter = Counter(tokens + ds['title'][i])  words\_count = len(tokens + ds['title'][i])   for token in np.unique(tokens):   tf = counter[token]/words\_count  Df = freq(token)  idf = np.log(N/(Df+1))  tf\_idf[i, token] = tf\*idf    tf\_idf\_title = {}  for i in range(N):   tokens = ds['title'][i]   counter = Counter(tokens + ds['body'][i])  words\_count = len(tokens + ds['body'][i])   for token in np.unique(tokens):   tf = counter[token]/words\_count  Df = freq(token)  idf = np.log(N/(Df+1))  tf\_idf\_title[i, token] = tf\*idf   for i in tf\_idf:  tf\_idf[i]\*=alpha  for i in tf\_idf\_title:  if i in tf\_idf:  tf\_idf[i]=tf\_idf\_title[i]  else:  tf\_idf[i]=tf\_idf\_title[i]\*(1-alpha)  return tf\_idf,vocab |

* For encoding the output variable to reduce the processing time and to eventually increases the machine learning algorithm efficiency

|  |
| --- |
| encoder = LabelEncoder() y\_train = encoder.fit\_transform(train['y']) y\_train |

|  |
| --- |
| array([0, 2, 1, ..., 1, 2, 2]) |

**Vectorize**

|  |
| --- |
| def vectorize(df,tf\_idf):  x\_train=pd.DataFrame()  x\_train['document']=np.array(df['body'])   for doc in df.index:   lst=np.zeros(len(vocab))  for i in range(len(lst)):  if vocab[i] in x\_train['document'][doc]:  lst[i]=tf\_idf[doc,vocab[i]]  x\_train['document'][doc]=lst.copy()  return x\_train |

**Chapter 7**

**Performance evaluation**

**7.1 Description of data**

This is a dataset containing 60,000 Stack Overflow questions from 2016-2021. Questions are classified into three categories:

1. HQ: High-quality posts without a single edit.
2. LQ\_EDIT: Low-quality posts with a negative score, and multiple community edits. However, they still remain open after those changes.
3. LQ\_CLOSE: Low-quality posts that were closed by the community without a single edit.

This data comprises of:

1. Id
2. Title
3. Body
4. Tags
5. Creation Date
6. Type of question

**7.2 Experimental setup**

|  |
| --- |
| import pandas as pd import numpy as np import nltk import re import string from nltk.corpus import stopwords from nltk.stem import PorterStemmer from num2words import num2words import matplotlib.pyplot as plt %matplotlib inline from operator import itemgetter from collections import Counter from sklearn.metrics import accuracy\_score from sklearn.metrics import confusion\_matrix from sklearn.svm import LinearSVC from sklearn.preprocessing import LabelEncoder |

**Data visualization**



Figure 7.1 Dataset

**Data representation**

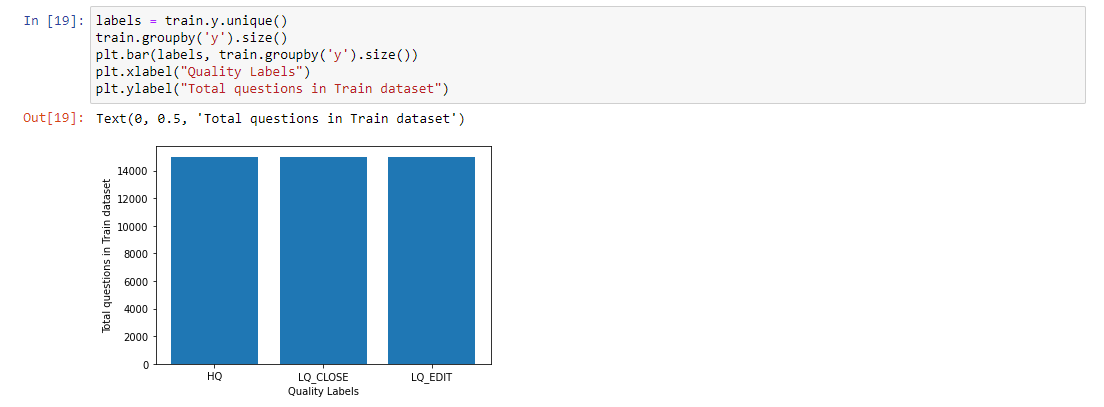
****

Figure 7.2 Visualized Dataset

**7.3 Experiment result**

Various methods applied regarding this project which are mainly:

* Deciding the optimal Vector size which satisfies given condition:
  + Minimum Vector size
  + Not too large as it will result in overfitting
  + Using most frequent words for vector
* Using different weight assigning technique for title and body which is Deciding how much importance title and body works carry.For body let us say we assigned the weight alpha then title would carry weight (1-alpha).This methods helps to give different importance to title and body words eventually creating only one single vector which include both title and body words.
* Another method for tf-idf vectorization is the one including only the words in the body.Which considers the vector mainly emphasizes on Body of the questions

In our setup the TF-IDF vectorization which considers only body of the question performed well as compared to TF-IDF vectorization which considers both title and body of the question

**Chapter 8**

**Problem timeline**

**8.1 Gantt chart**

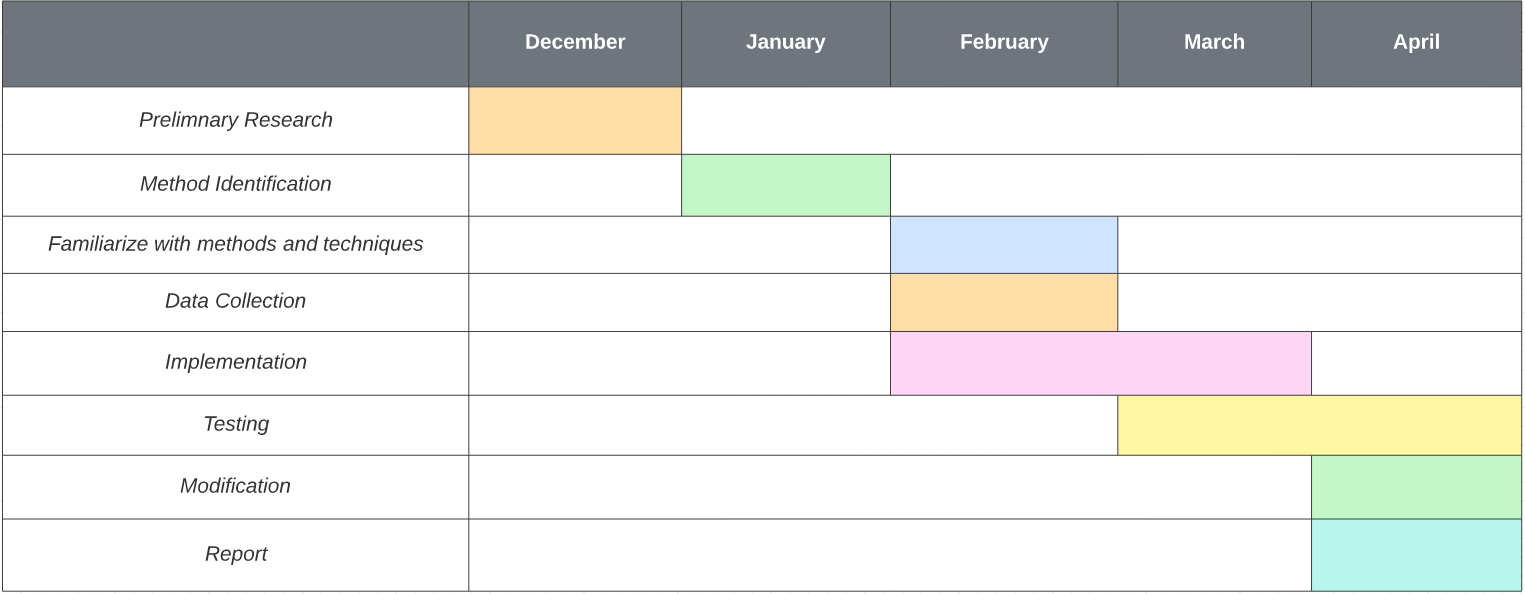
****

Figure 8.1 Gantt chart

**Chapter 9**

**Result**

* **Test case 1**

Training Data consist of 45,000 Unique Questions

Testing Dataset consist of 15,000 Unique Questions

Total Dataset size is 60,000 Unique Questions

Vector size-10,000

Vectorization method used ,considers both title and body of the given questions and vector is made of most frequent 10,000 words in both title and body.Here we have assigned different weights to the title and body words.

From the fig. 9.1,total 10,312 questions quality is predicted right.

Accuracy-69.02%

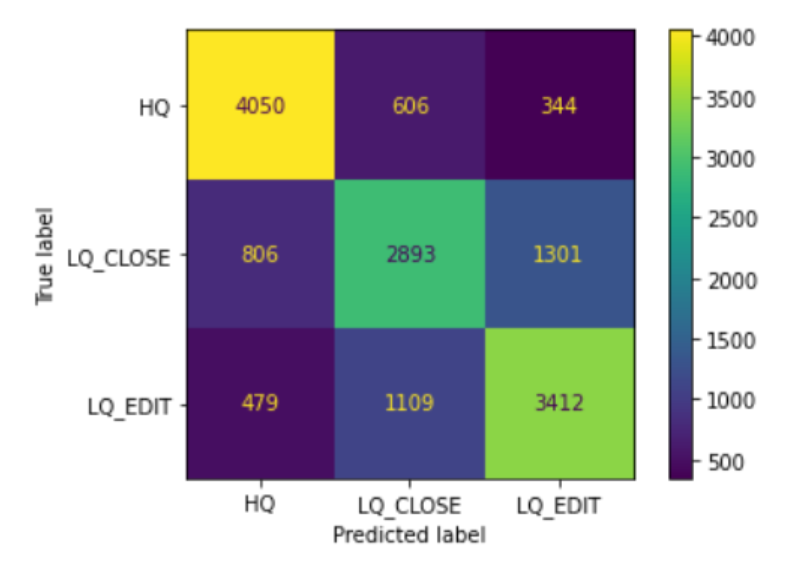


Figure 9.1 Confusion Matrix( Test case-1 )

* **Test case 2**

Training Data consist of 45,000 Unique Questions

Testing Dataset consist of 15,000 Unique Questions

Total Dataset size is 60,000 Unique Questions

Vector size-5,000

Vectorization method used ,considers both title and body of the given questions and creating a document which is consist of the both title and words.Vector is made of most frequent 5,000 words in the both title and body.Here we have calculated TF-IDF values of the word present in the body,if the word is only present in the title the that word won't be considered while Calculating Tf-Idf values From the fig. 9.1,total 10,371 questions quality is predicted right.

Accuracy-69.26%

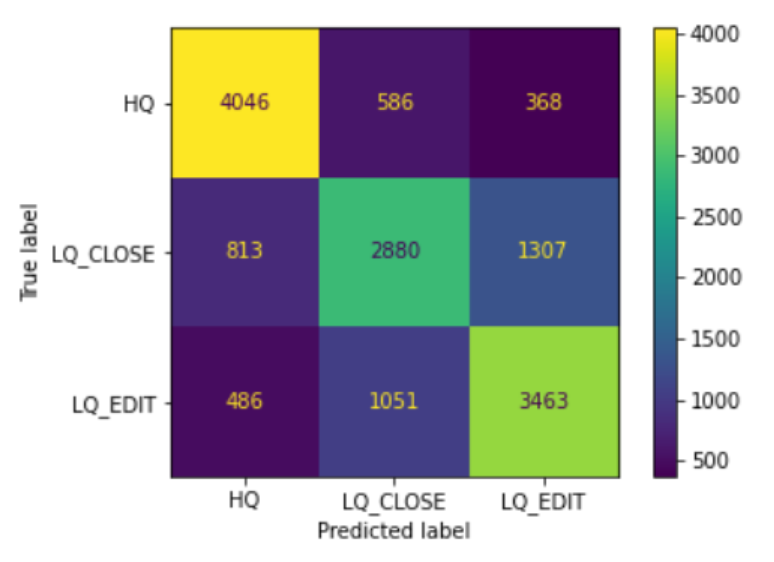


Figure 9.2 Confusion Matrix( Test case-2 )

**Conclusion**

The classification of the questions based on their quality into three different categories such as HQ,LQ\_EDIT and .LQ\_CLOSE will be efficient if the test case 2 is considered in which only the words which are only present in the title are not considered while the words which are present in both title and body are considered can give a slight uplift in the efficiency of the Machine learning module. Hence more emphasis should be put on the questions body to achieve a good accuracy.Further improvement can be done using Deep learning techniques.

**References**

1. <https://www.researchgate.net/publication/333574762_Towards_an_Accurate_Prediction_of_the_Question_Quality_on_Stack_Overflow_Using_a_Deep-Learning-Based_NLP_Approach>
2. <https://ieeexplore.ieee.org/document/8938342>
3. <http://aclanthology.lst.uni-saarland.de/R15-1005.pdf>
4. <https://www.kaggle.com/datasets/imoore/60k-stack-overflow-questions-with-quality-rate>
5. https://towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089