Mining Stack Overflow Data to visualize evolving trends in Technology

Ritika Bhatia
Department of Computer Science,
Carleton University,
K1S5B6,Ottawa,ON,Canada,
Email:ritikabhatia@cmail.carleton.ca

Wajiha Zafar
School of Information Technology,
Ryerson University,
Toronto, ON, Canada
Email:wajiha.zafar@ryerson.ca

Abstract— Stack overflow is a major question and answering community that constantly renders help to the developers and programmers acting as the community members. The questions and answers posted on the site not only describes the relevancy of the technical issues, but also depicts insights about the technical trends which are followed be the developers. This project involves mining and exploiting the question and answering data of Stack overflow website in order to explore the diverse technologies used by the developers and how the technical discussions revolve around the changes in the technology. Topic modeling, an unsupervised technique is applied on the dataset to visualize the distinct topics related to the technologies discussed by the developers. The topic modeling results when linked to the tags, describes the overall picture of which technology or programming language has been discussed.

Keywords—Data Mining, Mining Software Repositories, Natural Language Processing, Topic Modeling

I. INTRODUCTION

Online users require a medium to share or exchange their thoughts on plethora of diverse topics. Stack overflow is one of the vibrant and vigorous platforms that allows software developers to share and enhance their technical skills by constantly challenging themselves in the way they could help others by clearing queries of others. Due to large volume of information and data present, Community Based Question Answering Sites are gaining popularity,[1] and also the participation of the users on such online platforms is also increasing. Since the foundation of Stack overflow in 2008, it is gaining popularity and has become significant platform for developers to solve programming related problems. Besides discussion, it also facilitates users to rate the answer based on the satisfaction and its correctness. This voting system helps the users to earn reputation points and badges which encourages other users on the forum for participation[1]. In year 2016, there were around 4 million registered users, 11 million questions and 17 million answers.

The communities like Stack overflow and its attainment rate of the users is entirely dependent on the participation of the users which provide meaningful insights to various problems[2]. The data analysis in software development helps developer to apply various unique techniques and use the results of analysis for the process's optimization and better production decisions. In this project, we would be studying various implications of software data mining. The software development and engineering involve designing, developing and maintain of software. There are a lot of different types of data available related to the software development such as graphs, texts, records and figures. Data Mining is the process of understanding and analyzing large and complex dataset using machine learning and statistical techniques to generate new and useful information.

The research project will focus on mining and analyzing Stack Overflow Data. The Stack-Overflow is large community-based Question Answering (Q&A) platforms and it is growing tremendously nowadays, and the number of user interactions is also increasing. Stack Overflow is also one of popular computer programming related Q&A platform used by over 5 million developer and programmer across the world. It is an online platform, which helps developer and programmer in exchanging question answers and help in resolving issues related to the code, programming and development problems. The stack-overflow also features the option for user to ask questions and answer existing question and can vote and comment on the answers and questions posted by the other users across the world. The software data mining helps software developer and engineer to get more insight knowledge about the issues and problems faced by other developer during the development process and it also helps them to know what the most trending issue in software community are.

The previous data scientist and researchers focused on various aspects of stack overflow data, the data is analyzed to study the various and trending jobs in computer industry and also the regional difference between programmers along with sentiment analysis of programmer and developers by analyzing the user comments on Stack-Overflow. This research project will study various types of questions asked by the users, what are the recent technologies about which the users are talking, figuring out topics using Latent Dirichlet Allocation.

In this project, the dataset we used is from the Stack-Sample which is 10% sample of Stack-Overflow questions and answers data from 2008 to 2016 and freely available on Kaggle. This analysis will focus only on the Questions posted on stack-overflow and topic modeling to dig deep into the text related to the questions.

The methodology will consist of data loading, data Visualization, data cleaning and data preprocessing. The main objective of this research will be finding out the topics using probability-based approach LDA and topic evolution over the years and visualize the top topics evolved in these years This research will contain a descriptive analysis also under which the developer's questions will be categorized and finding out the most popular and similar topics. With the practical implementation of this project, it will highlight the area which needs attention by just manipulating as well as extraction of the real time-information which is the main key element of this project.

II. RELATED WORK

There have been many techniques devised and researches conducted in order to explore the infinite content of the repository. In research paper[3], the authors aimed to find the interest of the developers on Stack Overflow discussing about different topics from diverse backgrounds, what is the trending technical framework preference. They applied Latent Dirichlet Allocation, a statistical topic model[3], to know various topics discussed and in various domains, how the interest of the developers differs from time, and the changes described in the specific technologies. Other research paper[4], authors aimed to find the deleted questions on the stack overflow platform, at the time of the creation of the question, and find the low quality and the good quality questions. Moreover, they further explored to find out the probability of the deteriorated quality question to be deleted.

Another interesting research paper,[5] aimed to mine the discussion of the mobile development community in order to find out the relevant technical topics and the current frameworks that are dominated in that community. They applied Topic Modeling in order to highlight the different topics, and find their share and also found what are the main and most discussed topics. They also focused on the interrelationship existed between different questions and how one question leads to another question. Another research paper[6] mined the testing questions on stack overflow, in order to find out the client and server applications used for testing purposes. They also put in application of Latent Dirichlet Allocation, unsupervised learning technique in order to find the relevant topics pertaining to testing.

Another similar paper[7], that used Topic Modeling in order to find the topics concerning to the clients who are constantly utilizing the web services and their APIs. They also examined different associations and patterns existing between the discussion points of the web developers. The research paper[8] mined questions posted by web developers to figure out the categories of the topics of the discussion, mapping the most important topics and discussion points present in the discussion forums of web development. How the topics of web developers are inter-linked to the topics of the mobile community and also the difficulties faced by developers in the environment of the development. Another research paper[9], used Topic Modeling to find out the trends and topic impacts over the time changing, the inter relationship of the topics, the variations in the usage of the programming languages. They used graphs in order to visualize the changing topics over time. Another paper[10], mined the discussions of software engineers discussing about the android framework, the issues involved in the mobile development industry, the ease in usage of selective tools in this procedure, what are the challenges faced by the developers. They also used Topic Modeling in order to find out the relevant topics discussed. Another paper[10], mined the questions related to the software energy consumption, to figure out the distinctive characteristics, of the most common questions asked by the developers, the solutions deployed by the developers. Another paper[11], that mined the analogical libraries and topics related to those libraries present in the question and answer discussions. The authors utilized advanced NLP techniques and word embeddings for the association of different libraries, applying POS tagging[11], and constructing skip gram models.

III. THE DATASET

The dataset used in this project is Stack Sample: 10% of Stack overflow Q&A. The dataset includes the total of 10% extracted data including questions and answers from Stack overflow. The data consists of the following files:

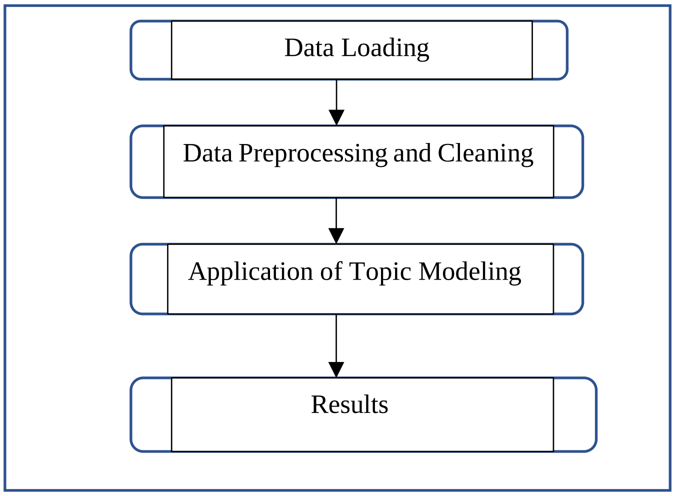
(i) Questions: Contains attributes such as body, title, creation date of the issue, closed date of the issue, owner id (the person who has posted the question) and the score on that particular question.

(ii) Answers: This file consists of the body or the content of the answers, creation date of the answer and the owner id of the answerer. The attribute Parent Id associates answer to the question table.

(iii) Tags: Tags file contains the specific tags of the question which programmer has used while posting or answering the question.

IV. METHODOLOGY

The methodology of this project initiates with the data cleaning and preprocessing in order to make algorithms consume the data. After the preprocessing is done, LDA topic modeling is applied to visualize the results of the diverse topics obtained. Figure 1: describes the methodology of the research followed.

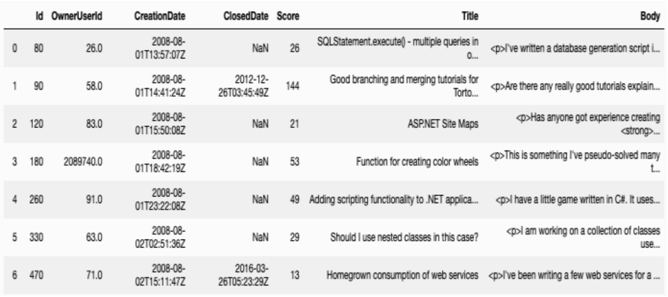

Data Loading
Data Preprocessing and Cleaning
Application of Topic Modeling
Results

A. Data Cleaning and Preprocessing

Dataset is in the form of CSV and it is loaded for manipulations using pandas library. The raw data collected contains various noises and undesired elements that can hinder desired results. In order to adjust and conforms the data according to the set format[2], extensive time is utilized to perform data cleaning. Additionally, when data is collected from multiple sources, it becomes essential to remove outliers and unnecessary information. Data cleaning is a procedure that recognizes incomplete, inaccurate, incorrect or irrelevant data[12], sets the appropriate format so that the data can be consumed by machine learning algorithms. Data cleaning ensures handling missing values, syntactics and grammatical errors fixing, removing unwanted and unnecessary observations[12] because ignoring such factors can deteriorate the quality of data. In this project, several steps were carried in order to carry the procedure for data cleaning. Figure 1describes the brief look of the dataset.

(i) Tokenization

Tokenization is an advanced Natural Language Processing Technique that take words in simplistic format as its main constituents[13]. Linguistics and Lexicographers take into consideration the concepts of words, their collation, their structure, their multi-morpheme[13], for usage in machine translation and Processing of Natural language. Token is the smallest part of the word that cannot be broken or split further. For instance, the sentence “This is a simple program to run” will return following tokens **“This”, “is”, “a”, “simple”, “program”, “to”, “run”.**

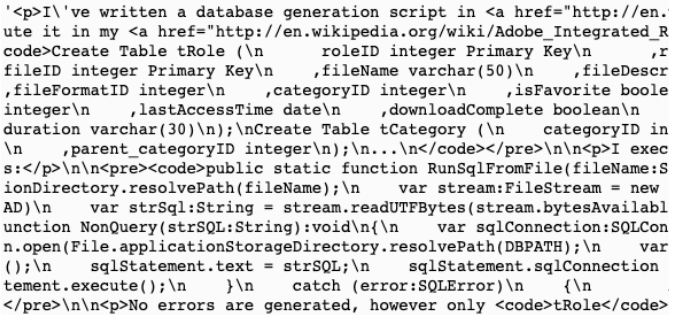


	Id	OwnerUserId	CreationDate	ClosedDate	Score	Title	Body
0	80	26.0	2008-08-01T13:57:07Z	NaN	26	SQLStatement.execute() - multiple queries in o...	<p>I've written a database generation script i...
1	90	58.0	2008-08-01T14:41:24Z	2012-12-26T03:45:49Z	144	Good branching and merging tutorials for Torto...	<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 <strong>...
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...
5	330	63.0	2008-08-02T02:51:36Z	NaN	29	Should I use nested classes in this case?	<p>I am working on a collection of classes use...
6	470	71.0	2008-08-02T15:11:47Z	2016-03-26T05:23:29Z	13	Homegrown consumption of web services	<p>I've been writing a few web services for a ...

Figure 1: Snapshot of the dataset used

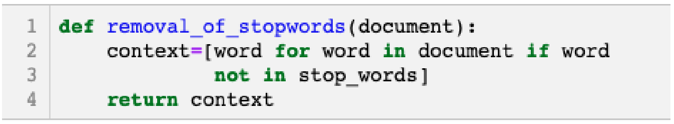
(ii) Removal of Stop words and Punctation Marks

One of the task in Data cleaning, requires removal of stop words. Stop words are words that occur frequently in the sentence to support the structure of the sentence. For instance, the stop words includes a, and, the, with, or, etc. NLTK or the Natural Language Toolkit provides feature for removal of stop words for various languages like English, Spanish, French, etc. For this project, the stop words list in English is utilized. The matching of the words present in the list is done with the data, and those words are filtered out from the textual data. NLTK library also provides features to remove punctuation marks or extra symbols present in the text that does not enhance the importance of overall textual data. Therefore, the punctuation marks are also removed from the text. Figure 3 and Figure 4 shows the function in python for removal of stop words and punctuation marks. Figure 2 illustrates the body of the question that needs to be cleaned.



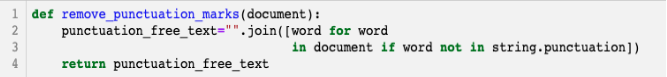
```
'<p>I\'ve written a database generation script in <a href="http://en.
ute it in my <a href="http://en.wikipedia.org/wiki/Adobe_Integrated_R
code>Create Table tRole (\n      roleID integer Primary Key\n      ,r
fileID integer Primary Key\n    ,fileName varchar(50)\n    ,fileDescr
,fileFormatID integer\n    ,categoryID integer\n    ,isFavorite boole
integer\n    ,lastAccessTime date\n    ,downloadComplete boolean\n
duration varchar(30)\n);\nCreate Table tCategory (\n    categoryID in
\n    ,parent_categoryID integer\n);\n...\n</code></pre>\n\n<p>I exec
s:</p>\n\n<pre><code>public static function RunSqlFromFile(fileName:S
ionDirectory.resolvePath(fileName);\n    var stream:FileStream = new
AD)\n    var strSql:String = stream.readUTFBytes(stream.bytesAvailabl
unction NonQuery(strSQL:String):void\n{\n    var sqlConnection:SQLCon
n.open(File.applicationStorageDirectory.resolvePath(DBPATH);\n    var
();\n    sqlStatement.text = strSQL;\n    sqlStatement.sqlConnection
tement.execute();\n    }\n    catch (error:SQLError)\n    {\n
</pre>\n\n<p>No errors are generated, however only <code>tRole</code>
```

Figure 2: uncleaned body of the question



```python
def removal_of_stopwords(document):
    context=[word for word in document if word
               not in stop_words]
    return context
```

Figure 3: illustrating the function to remove stop words

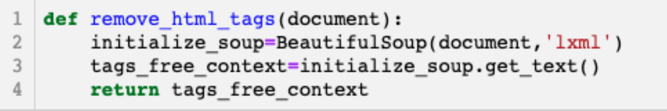


```python
def remove_punctuation_marks(document):
    punctuation_free_text="".join([word for word
                                   in document if word not in string.punctuation])
    return punctuation_free_text
```

Figure 4: illustrating the removal of punctuation marks from the text

(iii) Removal of Email Symbols

There are many unwanted symbols present in the data, including the symbols ‘@’, ‘.com’,’https’,’//’, that needs to be refined for the actual analysis of the data. Therefore, these symbols were removed from the data using regular expressions from Natural Language Processing Toolkit. The sentence is passed into the function which matches the pattern that needs to be removed and returns the cleared sentence.



```python
def remove_html_tags(document):
    initialize_soup=BeautifulSoup(document,'lxml')
    tags_free_context=initialize_soup.get_text()
    return tags_free_context
```

Figure 5: remove html tags from the text

(iv) Stemming

Stemming is a technique that is used to obtain the original or the root word out of the complete word. For instance, the root stem for words collecting, collect, and collected is same as collect. Therefore the procedure of stemming trims the suffixes and prefixes present in the sentence. This is done because, the conversion of words into vector form during preprocessing will consider the same word as different words and this can lead to bias in the dataset. Therefore, stemming needs to be performed.

V. Topic Modeling

LDA considers hidden patterns to associate the topics. The basic ideology behind Topic Modeling is to figure out the subjects that represent the data corpus in general. Topic Modeling is an unsupervised learning technique that discovers the latent topics that occur in the textual data. It constructs the topic per document model, known as Dirichlet distributions. It discovers the topical distributions that are present within the dataset, relates those textual data to the topics, and organize those topics in specific format. It is viewed as a technique to obtain the mixture of topics out of the context.

The hyper parameters of the Topic Modeling includes the alpha and the beeta, which defines the density of the topics per document and words per topic. The results of the Topic Modeling returns the matrix of the probability determining which words belong to which topic. This probability distribution will yield and aides to understand the diverse topics discussed in Stack Overflow. Figure 6 illustrates the Topic Modeling distribution out of the bag of words approach.

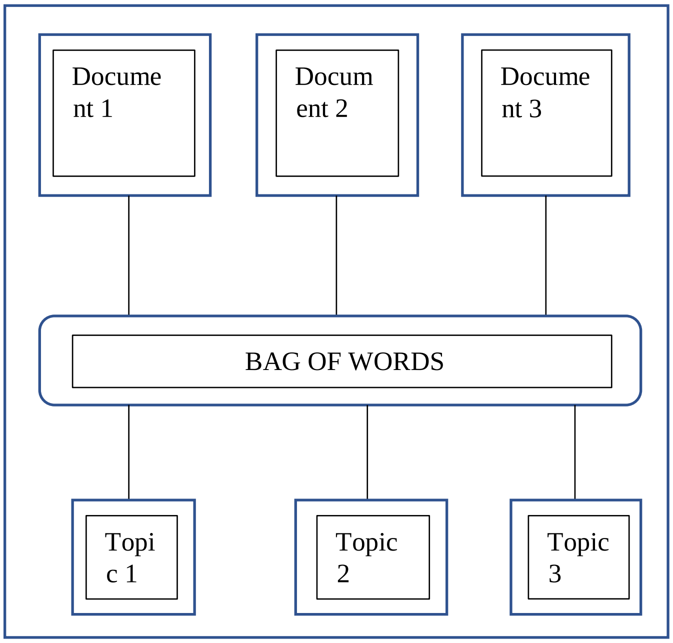


Figure 6 illustrating about the concept of Topic Modeling

Latent: Latent describes the hidden context or the abstract context that we aim to find out. Initially, it is assumed that the topics existing within the documents are unknown, yet the text is believed to be generated from those topics.

Dirichlet: It is assumed to be "Distribution of Distributions"[14]. It refers to the probability distribution formulated within the topics in document and the words per topic.

Allocation: Allocation simply maps the association with the words to topic and topics to the documents.

Consider P(T/D) defines the probability of the topics per document and P(W/T) defines the words present per topic.

The probability is calculated on the basis of the sum of products of all the probabilities existed within the documents and within the topics.

VI. RESULTS

The topic modeling in machine learning is used for discovering the Topics that occur in a collection of documents or a text. It is powerful tools used to define the hidden topic from the large raw text and also used in organizing the large documents. Topic Modeling is unsupervised machine learning algorithm which means that we as a data analyst don't have to decide over time about what topic will be with in the sea of raw texts. We implemented LDA topic modeling algorithm for this purpose. We used python's library genism and NLTK for the LDA and preprocessing of text in Question Body.

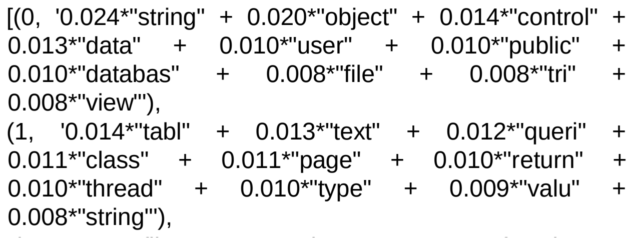
LDA is used to classify the large text in a document and build a topic per document model and words per topic model. LDA probabilistically assign word to a topic based on two things:

1. What topics in documents and How many time words has been assigned to the topic across the document (beta distribution)?
2. Extract what topic people discuss frequently from raw large text.

In our data set analysis, each stack overflow question body is a document. The statistically modeling process finds the topics in the text dataset and which words contribute to the topics and which topics contribute to the document texts. We fit the LDA model using the 12 topics to on each year data set, we build the document-term matrix. The topic modeling helps us in identifying which words contribute the topic across the documents. The LDA gave the top 12 topic keywords with their weights. The keywords are nicely segregated and collectively represent the topic by probability weights.

LDA Results:

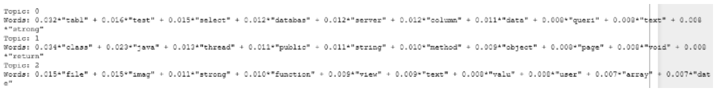
We fit the LDA model using the 12 topics to on each year data set, we build the document-term matrix. The topic modeling helps us in identifying which words contribute the topic across the documents. The LDA gave the top 12 topic keywords with their weights. The keywords are nicely segregated and collectively represent the topic by probability weights.



```
[(0, '0.024*"string" + 0.020*"object" + 0.014*"control" +
0.013*"data" + 0.010*"user" + 0.010*"public" +
0.010*"databas" + 0.008*"file" + 0.008*"tri" +
0.008*"view"'),
(1, '0.014*"tabl" + 0.013*"text" + 0.012*"queri" +
0.011*"class" + 0.011*"page" + 0.010*"return" +
0.010*"thread" + 0.010*"type" + 0.009*"valu" +
0.008*"string"'),
```

The above screen shot is the LDA model result of 2008, the top 3 Topics keywords with their weights. The results show that, keywords "String" and "Object" have highest weight of 0.024 while the same keyword "Code" is repeated in other topic and while in topic 3 keyword "File" have a weight of 0.018 while the same keyword "file" is repeated in another topic as well. These results show that in 2008 the people were more discussing about the issues related to the datatypes or related to the code file.

Topic Modeling result 2009:

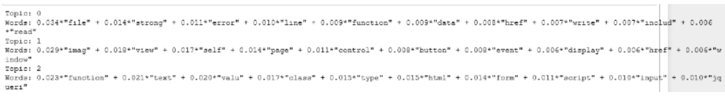


```
Topic: 0
Words: 0.032*"tabl" + 0.016*"test" + 0.015*"select" + 0.012*"databas" + 0.012*"server" + 0.012*"column" + 0.011*"data" + 0.008*"queri" + 0.008*"text" + 0.008
*"strong"
Topic: 1
Words: 0.034*"class" + 0.023*"java" + 0.013*"thread" + 0.011*"public" + 0.011*"string" + 0.010*"method" + 0.009*"object" + 0.008*"page" + 0.008*"void" + 0.008
*"return"
Topic: 2
Words: 0.015*"file" + 0.015*"imag" + 0.011*"strong" + 0.010*"function" + 0.009*"view" + 0.009*"text" + 0.008*"valu" + 0.008*"user" + 0.007*"array" + 0.007*"dat
e"
```

Topic Modeling result 2010:

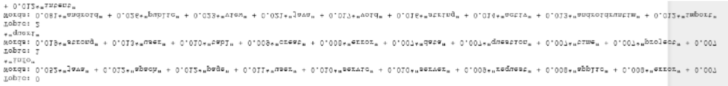
```
[(0, '0.025*"http" + 0.020*"strong" + 0.018*"user" +
0.016*"href" + 0.013*"applic" + 0.010*"page" +
0.010*"nofollow" + 0.008*"question" + 0.008*"know" +
0.007*"develop"'),
(1, '0.038*"java" + 0.016*"error" + 0.015*"server" +
0.011*"servic" + 0.010*"connect" + 0.010*"time" +
0.009*"messag" + 0.008*"client" + 0.008*"thread" +
0.008*"http"'),
```

Topic Modeling result 2011:



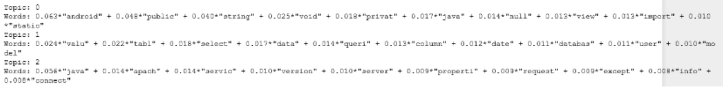
```
Topic: 0
Words: 0.034*"file" + 0.014*"strong" + 0.011*"error" + 0.010*"line" + 0.009*"function" + 0.009*"data" + 0.008*"href" + 0.007*"write" + 0.007*"includ" + 0.006
*"read"
Topic: 1
Words: 0.029*"imag" + 0.018*"view" + 0.017*"self" + 0.014*"page" + 0.011*"control" + 0.008*"button" + 0.008*"event" + 0.006*"display" + 0.006*"href" + 0.006*"w
indow"
Topic: 2
Words: 0.023*"function" + 0.021*"text" + 0.020*"valu" + 0.017*"class" + 0.015*"type" + 0.015*"html" + 0.014*"form" + 0.011*"script" + 0.010*"input" + 0.010*"jq
ueri"
```

Topic modeling results 2012:



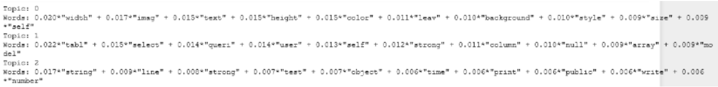
```
Topic: 0
Words: 0.025*"java" + 0.013*"apach" + 0.013*"page" + 0.011*"user" + 0.010*"servic" + 0.010*"server" + 0.008*"request" + 0.008*"applic" + 0.008*"error" + 0.007
*"info"
Topic: 1
Words: 0.019*"string" + 0.013*"user" + 0.010*"tabl" + 0.009*"creat" + 0.008*"error" + 0.007*"data" + 0.007*"question" + 0.007*"time" + 0.007*"project" + 0.007
*"queri"
Topic: 2
Words: 0.081*"android" + 0.039*"public" + 0.033*"view" + 0.021*"java" + 0.017*"void" + 0.018*"privat" + 0.014*"activ" + 0.013*"androidruntim" + 0.013*"import"
+ 0.013*"intent"
```

Topic Modeling results 2013:



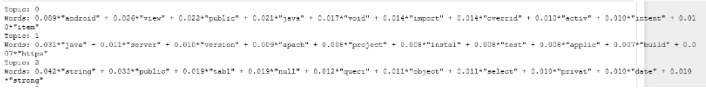
```
Topic: 0
Words: 0.063*"android" + 0.048*"public" + 0.040*"string" + 0.025*"void" + 0.018*"privat" + 0.017*"java" + 0.014*"null" + 0.013*"view" + 0.013*"import" + 0.010
*"static"
Topic: 1
Words: 0.024*"valu" + 0.022*"tabl" + 0.018*"select" + 0.017*"data" + 0.014*"queri" + 0.013*"column" + 0.012*"date" + 0.011*"databas" + 0.011*"user" + 0.010*"mo
del"
Topic: 2
Words: 0.056*"java" + 0.014*"apach" + 0.014*"servic" + 0.010*"version" + 0.010*"server" + 0.009*"properti" + 0.009*"request" + 0.009*"except" + 0.008*"info" +
0.008*"connect"
```

Topic Modeling results 2014:



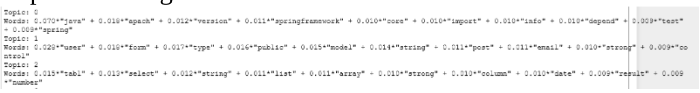
```
Topic: 0
Words: 0.020*"width" + 0.017*"imag" + 0.015*"text" + 0.015*"height" + 0.015*"color" + 0.011*"leav" + 0.010*"background" + 0.010*"style" + 0.009*"size" + 0.009
*"self"
Topic: 1
Words: 0.022*"tabl" + 0.015*"select" + 0.014*"queri" + 0.014*"user" + 0.013*"self" + 0.012*"strong" + 0.011*"column" + 0.010*"null" + 0.009*"array" + 0.009*"mo
del"
Topic: 2
Words: 0.017*"string" + 0.009*"line" + 0.008*"strong" + 0.007*"test" + 0.007*"object" + 0.006*"time" + 0.006*"print" + 0.006*"public" + 0.006*"write" + 0.006
*"number"
```

Topic Modeling results 2015:



```
Topic: 0
Words: 0.089*"android" + 0.026*"view" + 0.022*"public" + 0.021*"java" + 0.017*"void" + 0.014*"import" + 0.014*"overrid" + 0.010*"activ" + 0.010*"intent" + 0.01
0*"item"
Topic: 1
Words: 0.031*"java" + 0.011*"server" + 0.010*"version" + 0.009*"apach" + 0.008*"project" + 0.008*"instal" + 0.008*"test" + 0.008*"applic" + 0.007*"build" + 0.0
07*"https"
Topic: 2
Words: 0.042*"string" + 0.033*"public" + 0.019*"tabl" + 0.019*"null" + 0.012*"queri" + 0.011*"object" + 0.011*"select" + 0.010*"privat" + 0.010*"date" + 0.010
*"strong"
```

Topic Modeling result 2016:



```
Topic: 0
Words: 0.070*"java" + 0.018*"apach" + 0.012*"version" + 0.011*"springframework" + 0.010*"core" + 0.010*"import" + 0.010*"info" + 0.010*"depend" + 0.009*"test"
+ 0.009*"spring"
Topic: 1
Words: 0.028*"user" + 0.018*"form" + 0.017*"type" + 0.016*"public" + 0.015*"model" + 0.014*"string" + 0.011*"post" + 0.011*"email" + 0.010*"strong" + 0.009*"co
ntrol"
Topic: 2
Words: 0.015*"tabl" + 0.013*"select" + 0.012*"string" + 0.011*"list" + 0.011*"array" + 0.010*"strong" + 0.010*"column" + 0.010*"date" + 0.009*"result" + 0.009
*"number"
```

We visualized the LDA results in R using the “ggplot” and “tidy verse” library to see the trends of top terms in each topic. We applied the same LDA techniques to each year data and got following LDA results

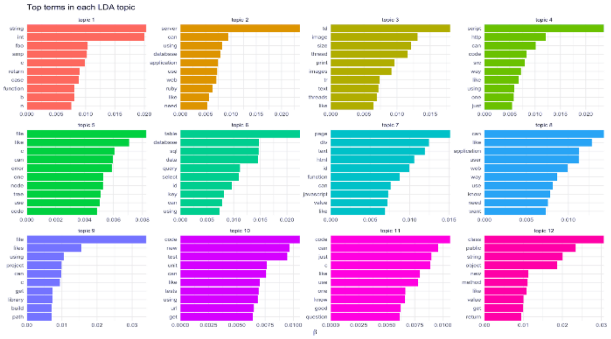


Fig 6: Top Topics of year 2008

The above group of plots of top topic related to the topic shows that the top terms used in the 12 topics, the terms in the topics are in English language not the words from code. The topic These terms show that, most occurring keyword in all the 12 topic is “code”, “file”, “string”, ’C”, which clearly indicate that software developer and programmer were categorically talking and discussing about the object-oriented programming issue and most of the words in these topics look general and applicable to almost all of the technologies. The same type visualization of all the other year’s topic is shown in the appendix of this report.

Connecting Topics to Tags:

The topic model estimates the proportion of words from each document that are generated from topic and also estimate the probability of topics from the document. So, we created the LDA matrix of the Question topic and then trained on the tags to see the connection of tags with topics. The visualization of 2008 tags connected to the topic is below shows that topic one is for C++ programming language, topic 2is for SQL database and server while topic 7 and 8 talk about JavaScript, java and other object oriented programming.

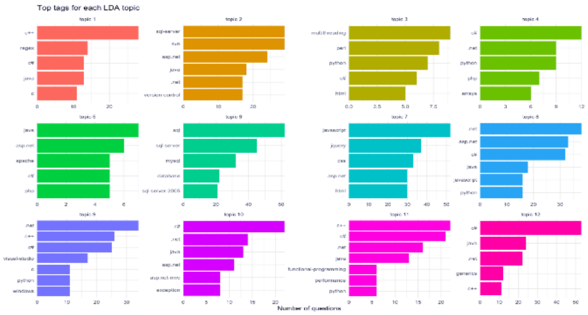


Fig 7: Tags connected to Top Topics of year 2008

The tags revealed the most discussed topics are related to the “Java”, “JavaScript”, “C#” “C++”,” r”, “Sql”, “Python” and “php” are the programming languages discussed most in these tags. Each year the tags related to the topics changed, in later year 2013 to 2016 the most frequent and most talk about is “Andriod” while “java” repeats in every topic from 2010 to 2016.

TF-IDF of Topics for each year:

After applying the LDA topic modeling to the question body and then connecting with the corresponding tags, we calculated the term frequency – Inverse Document frequency for the top words in stack over flow topics. The TF-IDF calculated the importance of words related to the document and occurrence of words in the documents. We did TF-IDF calculation R using the tidy verse library for each year dataset.

On the basis of LDA topic modeling results and tags connected to these topics, we filters the most important topic keywords/terms from the results and then calculated the term frequency in accordance with the documents.

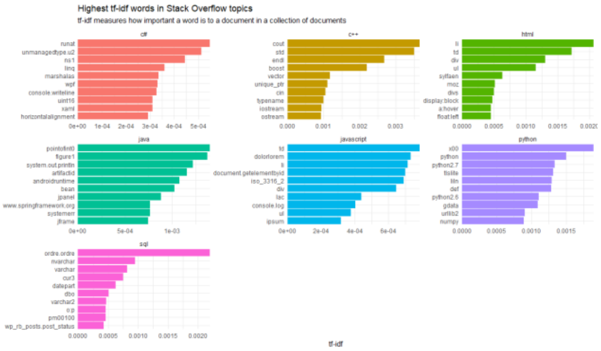


Fig 8: TF-IDS of year 2008 Topics

The plot shows that, terms related to the ”C#”, “JavaScript” and “java” were discussed a lot and frequently as compared to the “Python”, “Sql” and “html” We applied the same Term frequency calculation for each year dataset and then visualize the results using R’s GGPLOT package. The results are included in the appendix of this report.

Comparison of the Topics in each year:

We build word cloud of each year LDA topic results. The world cloud is not considering an optimal for the scientific purposed, but they can provide very good and quick visual overview of the set of terms.

We used R’s word cloud package and also build the same word cloud using the python’s NLTK package. Wordcloud extracts only the most important terms and the one with the high probability across the topics.

line
https
npm
following
date
file
using
http
string
new
var
std
n
s
w
amp
can
one
int
b
c
d
data
use
x
get
code
like
char
type
error
name
database

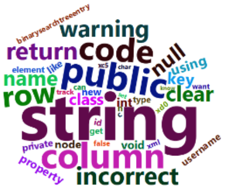
WordCloud-2008

solution
example
service
object
thanks
template
define
unit
public
using
class
type
int
can
set
list
base
web
c
edit
new
class
need
test
use
x
way
name
value
like
net
know
python
want
classes
num
empty
return
add
framework

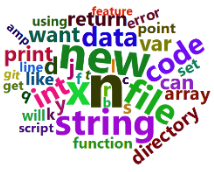
WordCloud-2009

calayer
first
text
select
want
status
true
user
will
name
object
new
set
file
error
string
get
input
val
way
test
id
use
data
one
code
like
can
type
frame
layer
return
selected
json
using
page
value
view
question

WordCloud-2010



WordCloud-2011



WordCloud-2012

type
following
npm
https
line
using
data
error
char
k
std
one
use
code
y
get
var
file
d
c
b
datewt
int
new
amp
n
string
like
http
can
database
name

WordCloud-2013

androidmethod
return
findviewbyid
button
item
override
true
activity
function
data
view
false
can
class
code
public
id
null
var
new
list
void
int
intent
http
import
error
string
android:id
private
listview
textview

WordCloud-2014

question
first
true
selected
status
frame
input
code
using
object
can
one
return
new
test
val
text
data
get
user
file
set
error
way
id
like
function
use
will
view
want
string
type
name
page
value
layer
select
calayer
json

WordCloud-2015

savedinstancestate
android:name
private
method
object
answers
return
static
new
final
catch
file
id
true
void
intent
int
http
e
uri
class
try
view
null
extends
false
code
activity
public
boolean
protected
data
string
list
context
get
question_id
arraylist
override

WordCloud-2016

The above word cloud reveals that the term, “String”, “Public”, “Code” and “Data” are the most frequent and important terms used in the topics and these terms seems to appear in each year topics. This can be inferred that, these terms are related to the most important and top tags and also related as we saw in LDA results that SQL database and different programming languages and keywords related to them have high weights in topics than other.

The word clouds of top terms in topics from year 2008 to 2016 reveals that term “String” is used in each year and we can also infer from that, the topic labels change over the year and user were talking mostly about the object oriented programming or code issues related to the data analytics, web development, data base and conversion or how to deal with string datatype in data analysis.

We also created the small comparison table based on the important tags connected to the topics from the results of Topic modeling to classify the most occurring and changing topic across the year.

The comparison table below shows that, most of the time software developer and programmer discussed about issue related to the programming languages, database and development. The most related topics in all years are about the programming language as “Java”, “JavaScript”, “SQL” and “Php”. We can also see that, from year 2008 to 2016 the discussion topic about the technology and operating system changed from “iPhone” to the “Android”.

This seems like most of development work and most of the trending topic in software industry is about the “Android” application along with other programming languages like “Python” specially most recent year from 2014 to 2016.

Top Topics by relation to their tags	Year
C++, C#, Sql-server, .net, Iphone	2008
C#, Php, Sql, Iphone	2009
Java, JavaScript, C#, Sql	2010
JavaScript, Jquery, Java, C#, Php	2011
Java, Jquery, C#, Php, Andriod	2012
Javascript, Andriod, C++, Php, Sql	2013
Php, C#, Java, Andriod	2014
Java, Python, Javascript, Andriod	2015
Php, Javascripts, HTML, C#, Android, C++	2016

VII. CONCLUSION AND FUTURE WORK

In the nutshell, different topics were discovered out of the textual data in order to find the most relevant technologies discussed. From the overall LDA results and by over all computational time of the program in Python and R increase from year 203 to 2016, which indicates that, number of Questions and answers as well as participants increased with the following years. Also, the results indicate that, the most dominant topics remained within the nine years include technologies related to C#, Java, PHP, Android, JavaScript.
The further research and work can be done related to the classification of the more dominant topics for each year according to the highest-ranking scores. Most of the topic over the year get inactive in stack overflow, we can also look deep into the cause and how long the topic stays active on the stack overflow.

VIII. REFERENCES

[1] Ahasanuzzaman, M., Asaduzzaman, M., Roy, C. K., & Schneider, K. A. (2016, May). Mining duplicate questions in stack overflow. In *Proceedings of the 13th International Conference on Mining Software Repositories* (pp. 402-412). ACM.

[2] Movshovitz-Attias, Dana, et al. "Analysis of the reputation system and user contributions on a question answering website: Stackoverflow." *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, 2013.

[3] Barua, Anton, Stephen W. Thomas, and Ahmed E. Hassan. "What are developers talking about? an analysis of topics and trends in stack overflow." *Empirical Software Engineering* 19.3 (2014): 619-654.

[4] Correa, Denzil, and Ashish Sureka. "Chaff from the wheat: Characterization and modeling of deleted questions on stack overflow." *Proceedings of the 23rd international conference on World wide web*. ACM, 2014.

[5] Rosen, Christoffer, and Emad Shihab. "What are mobile developers asking about? a large scale study using stack overflow." *Empirical Software Engineering* 21.3 (2016): 1192-1223.

[6] Kochhar, Pavneet Singh. "Mining testing questions on stack overflow." *Proceedings of the 5th International Workshop on Software Mining*. ACM, 2016.

[7] Venkatesh, Pradeep K., et al. "What do client developers concern when using web apis? an empirical study on developer forums and stack overflow." *2016 IEEE International Conference on Web Services (ICWS)*. IEEE, 2016.

[8] Bajaj, Kartik, Karthik Pattabiraman, and Ali Mesbah. "Mining questions asked by web developers." *Proceedings of the 11th Working Conference on Mining Software Repositories*. ACM, 2014.

[9] Movshovitz-Attias, Dana, et al. "Analysis of the reputation system and user contributions on a question answering website: Stackoverflow." *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, 2013.

[10] Villanes, Isabel K., et al. "What are software engineers asking about android testing on stack overflow?." *Proceedings of the 31st Brazilian Symposium on Software Engineering*. ACM, 2017.

[11] Pinto, Gustavo, Fernando Castor, and Yu David Liu. "Mining questions about software energy consumption." *Proceedings of the 11th Working Conference on Mining Software Repositories*. ACM, 2014.

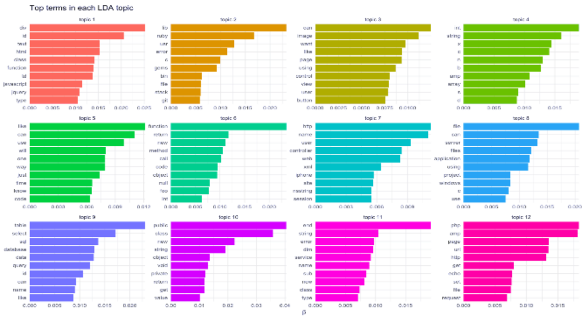
[12] https://sunscrapers.com/blog/why-is-clean-data-so-important-for-analytics-and-business-intelligence/

[13] Webster, J. J., & Kit, C. (1992). Tokenization as the initial phase in NLP. In *COLING 1992 Volume 4: The 15th International Conference on Computational Linguistics*.

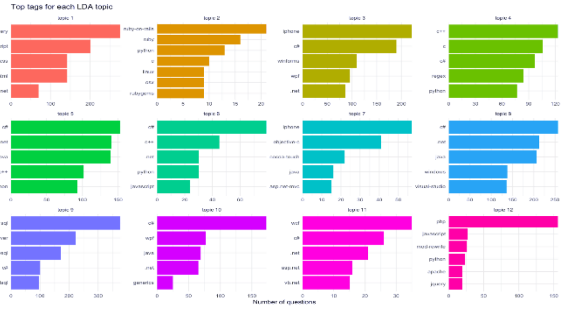
[14] https://medium.com/analytics-vidhya/topic-modeling-using-lda-and-gibbs-sampling-explained-49d49b3d1045

Appendix A: Topic Visualization:

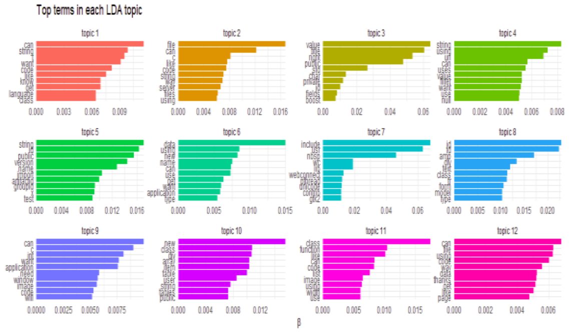
Results 2009 – Top Topics:



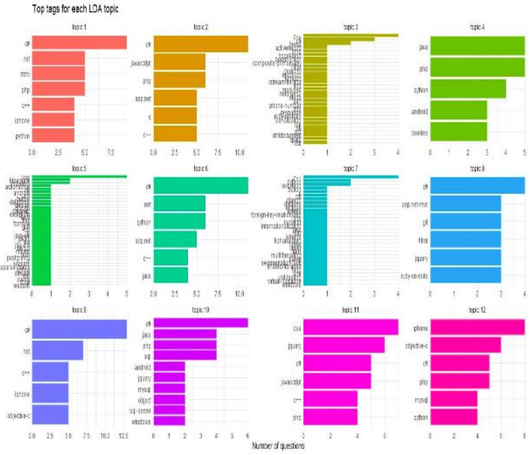
Tags connected to Topic – 2009:



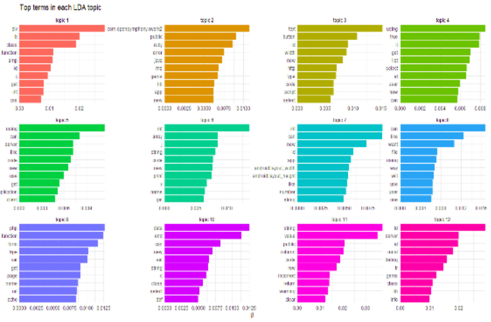
Results year-2010 – Top Topics:



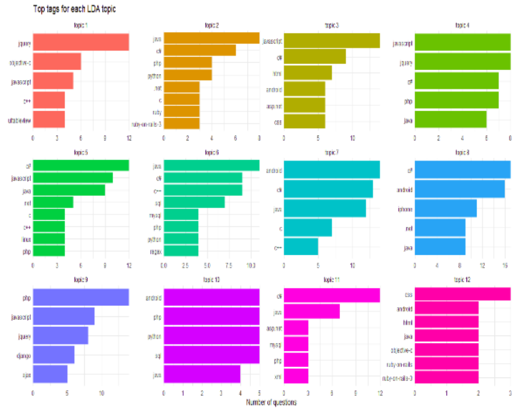
Tags Connected to the Topics – 2010:



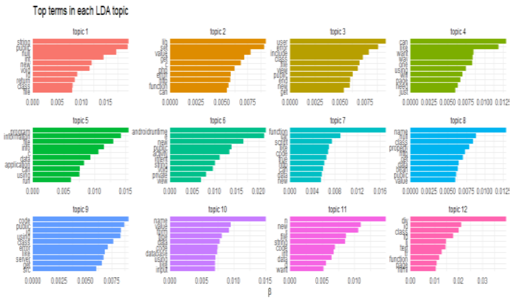
Results year-2011 – Top Topics:



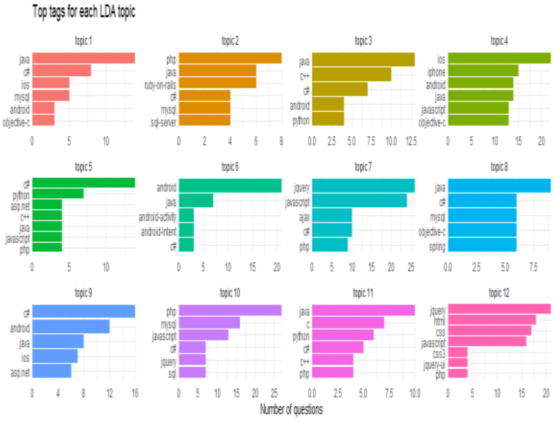
Tags connected to Topic – 2011:



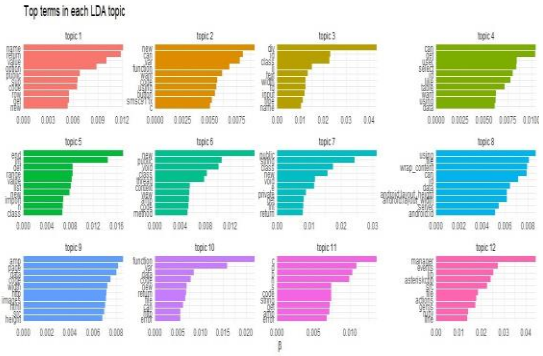
Results year – 2012 – Top Topic:



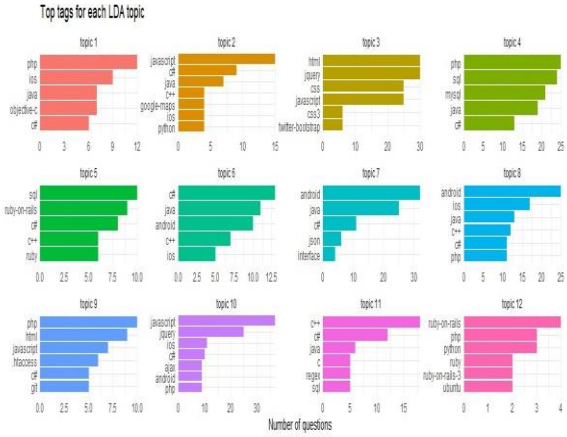
Tags connected to Top Topics – 2012:



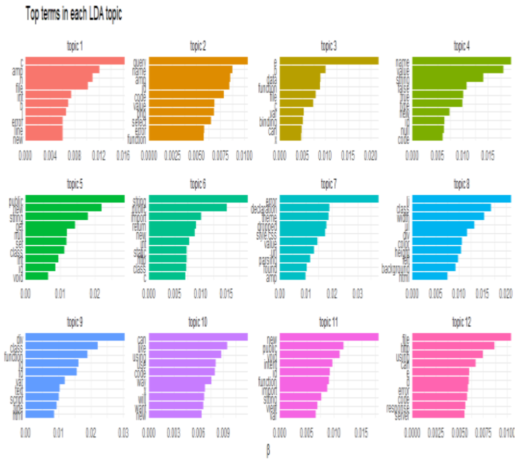
Results year – 2013 – Top Topic:



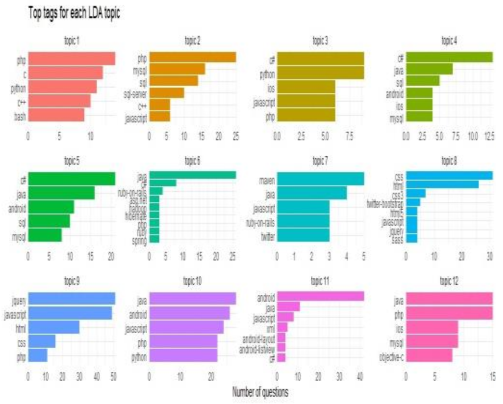
Tags connected to Top Topics – 2013:



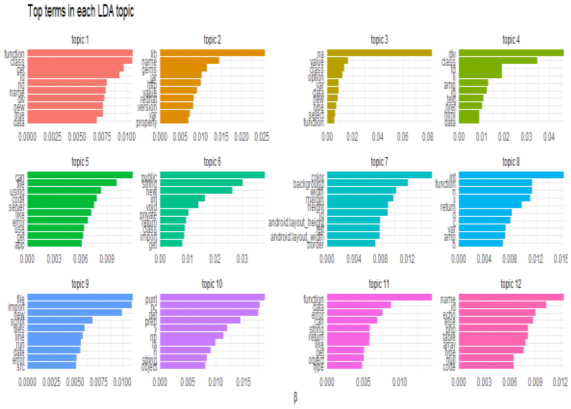
Results year – 2014:



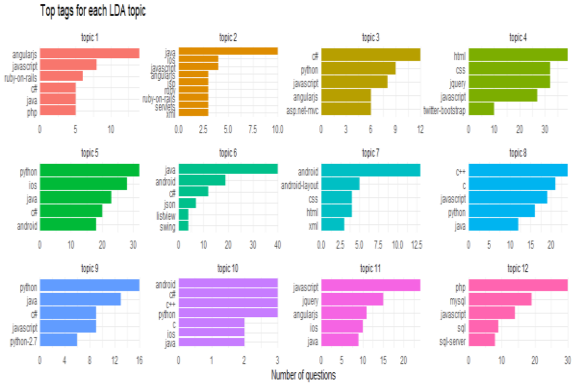
Tags connected to Top Topics – 2014:



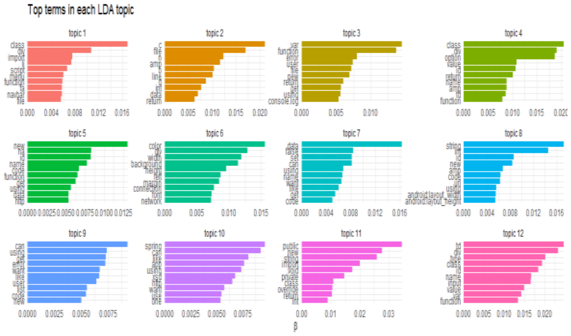
Results year – 2015- Top Topics:



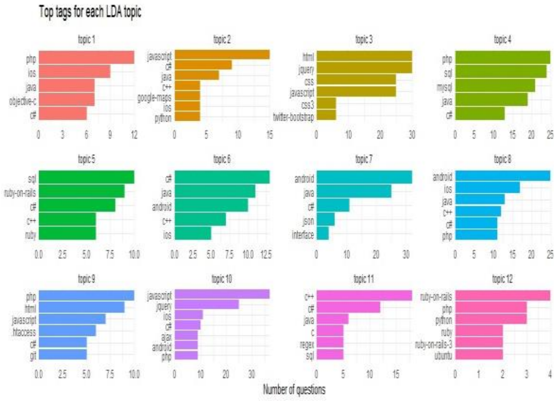
Tags connected to the Top Topics – 2015:



Result year – 2016 – Top Topics:

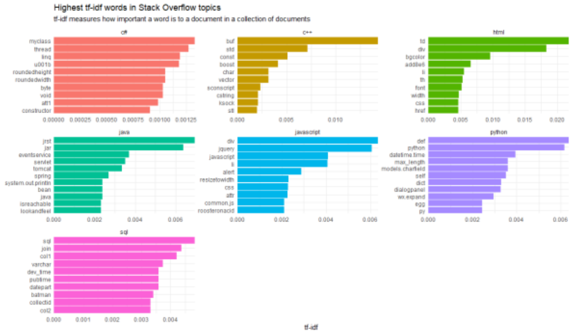


Tags connected to the Top Topics – 2016:

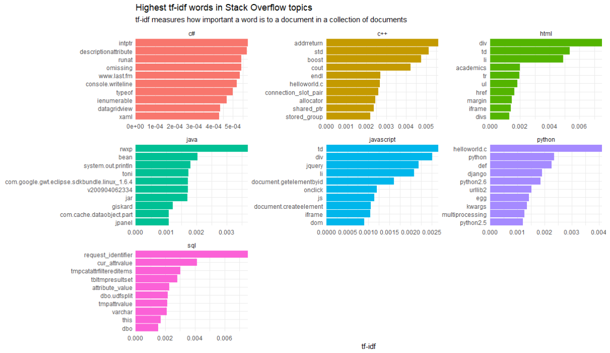


Appendix B: TF-IDF Visualization:

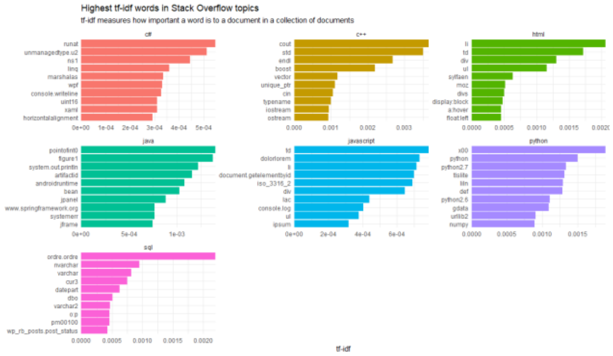
TF-IDF plot – 2008:



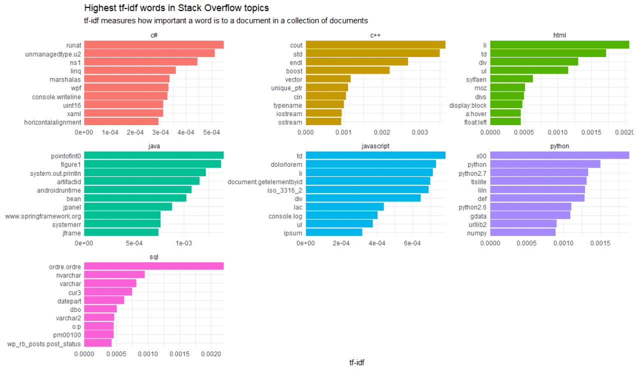
TF-IDF Plot – 2009:



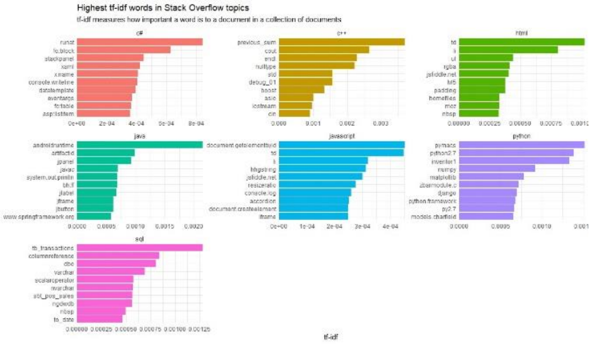
TF-IDF Plot – 2010:



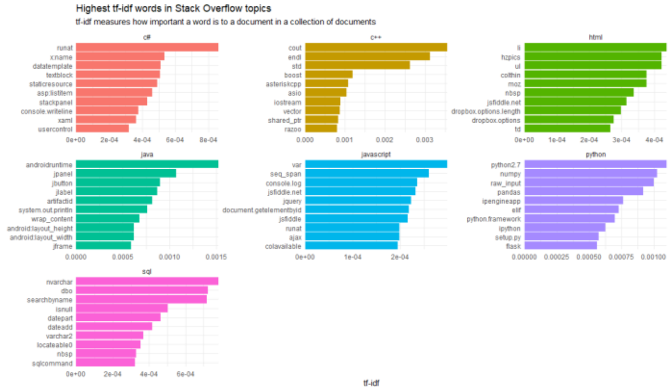
TF-IDF Plot – 2011:



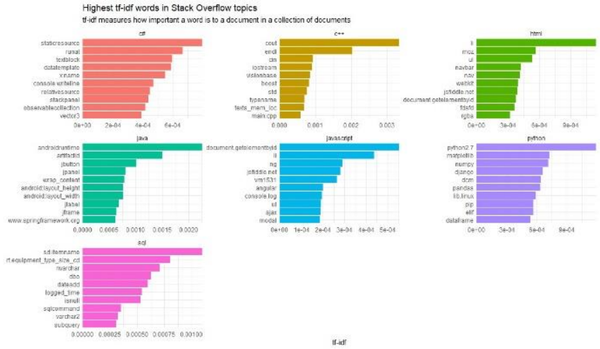
TF-IDF Plot – 2012:



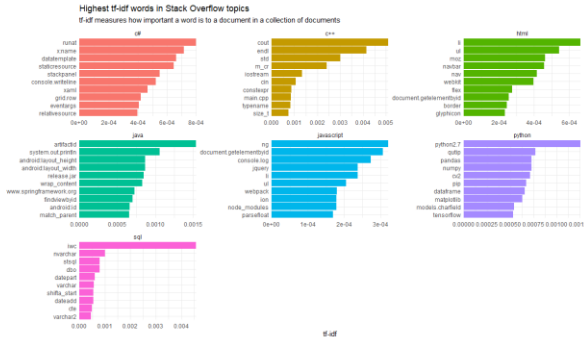
TF-IDF Plot – 2013:



TF-IDF Plot – 2014:



TF-IDF Plot – 2016:


Highest tf-idf words in Stack Overflow topics
tf-idf measures how important a word is to a document in a collection of documents
c#
runat
x:name
datatemplate
staticresource
stackpanel
console.writeline
xaml
grid.row
eventargs
relativesource
0e+00 2e-04 4e-04 6e-04 8e-04
c++
cout
endl
std
m_cr
iostream
cin
constexpr
main.cpp
typename
size_t
html
li
ul
moz
navbar
nav
webkit
flex
document.getelementbyid
border
glyphicon
0e+00 2e-04 4e-04 6e-04
java
artifactid
system.out.println
android:layout_height
android:layout_width
release.jar
wrap_content
www.springframework.org
findviewbyid
android:id
match_parent
javascript
ng
document.getelementbyid
console.log
jquery
li
ul
webpack
ion
node_modules
parsefloat
0e+00 1e-04 2e-04 3e-04
python
python2.7
qutip
pandas
numpy
cv2
pip
dataframe
matplotlib
models.charfield
tensorflow
sql
iwc
nvarchar
stsql
dbo
datepart
varchar
shifta_start
dateadd
cte
varchar2
tf-idf