In [1]:

```
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:100% !important; }</style>"))
```

In [2]:

```
import sklearn
print('The scikit-learn version is {}.'.format(sklearn.__version__))
```

```
The scikit-learn version is 0.20.1.
```

TASK <\h1> :-

- **Use bag of words upto 4 grams and compute the micro f1 score with Logistic regression(OvR)**
- **Perform hyperparam tuning on alpha (or lambda) for Logistic regression to improve the performance using GridSearch**
- **Try OneVsRestClassifier with Linear-SVM (SGDClassifier with loss-hinge)**

STACK OVERFLOW TAG PREDICTION ASSIGNMENT

In [3]:

```
!ls
```

```
20_02_2019_final_(2).ipynb  Train.csv  Train.zip
```

In [4]:

```
!pip install --user scikit-multilearn
```

```
Requirement already satisfied: scikit-multilearn in /home/saikrishna6680/.lo
cal/lib/python3.7/site-packages (0.2.0)
```

In [5]:

```
!pip3 install wordcloud
```

```
Requirement already satisfied: wordcloud in /home/saikrishna6680/.local/lib/
python3.5/site-packages
Requirement already satisfied: pillow in /usr/local/lib/python3.5/dist-packa
ges (from wordcloud)
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.5/site
-packages (from wordcloud)
Requirement already satisfied: icc-rt in /usr/local/lib/python3.5/dist-packa
ges (from numpy>=1.6.1->wordcloud)
Requirement already satisfied: mkl in /usr/local/lib/python3.5/dist-packages
(from numpy>=1.6.1->wordcloud)
Requirement already satisfied: mkl-random in /usr/local/lib/python3.5/dist-p
ackages (from numpy>=1.6.1->wordcloud)
Requirement already satisfied: mkl-fft in /usr/local/lib/python3.5/dist-pack
ages (from numpy>=1.6.1->wordcloud)
Requirement already satisfied: tbb4py in /usr/local/lib/python3.5/dist-packa
ges (from numpy>=1.6.1->wordcloud)
Requirement already satisfied: intel-openmp in /usr/local/lib/python3.5/dist
-packages (from icc-rt->numpy>=1.6.1->wordcloud)
Requirement already satisfied: intel-numpy in /usr/local/lib/python3.5/dist-
packages (from mkl-random->numpy>=1.6.1->wordcloud)
Requirement already satisfied: tbb==2019.* in /usr/local/lib/python3.5/dist-
packages (from tbb4py->numpy>=1.6.1->wordcloud)
```

In [6]:

```
#importing libraries
import warnings
warnings.filterwarnings("ignore")
import pickle
import pandas as pd
import sqlite3
import csv
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from wordcloud import WordCloud
import re
import os
from sqlalchemy import create_engine # database connection
import datetime as dt
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem.snowball import SnowballStemmer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.multiclass import OneVsRestClassifier
from sklearn.linear_model import SGDClassifier
from sklearn import metrics
from sklearn.metrics import f1_score,precision_score,recall_score
from sklearn import svm
from sklearn.linear_model import LogisticRegression
from skmultilearn.adapt import mlknn
from skmultilearn.problem_transform import ClassifierChain
from skmultilearn.problem_transform import BinaryRelevance
from skmultilearn.problem_transform import LabelPowerset
from sklearn.naive_bayes import GaussianNB
from datetime import datetime
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import GridSearchCV
```

Stack Overflow: Tag Prediction

1. Business Problem

1.1 Description

Description

Stack Overflow is the largest, most trusted online community for developers to learn, share their programming knowledge, and build their careers.

Stack Overflow is something which every programmer use one way or another. Each month, over 50 million developers come to Stack Overflow to learn, share their knowledge, and build their careers. It features questions and answers on a wide range of topics in computer programming. 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 and edit questions and answers in a fashion similar to a wiki or Digg. As of April 2014 Stack Overflow has over 4,000,000 registered users, and it exceeded 10,000,000 questions in late August 2015. Based on the type of tags assigned to questions, the top eight most discussed topics on the site are: Java, JavaScript, C#, PHP, Android, jQuery, Python and HTML.

Problem Statemtent

Suggest the tags based on the content that was there in the question posted on Stackoverflow.

Source: https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction/

1.2 Source / useful links

Data Source : https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction/data (https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction/data)
Youtube : https://youtu.be/nNDqbUhtIRg (https://youtu.be/nNDqbUhtIRg)
Research paper : https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tagging-1.pdf (https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tagging-1.pdf)
Research paper : https://dl.acm.org/citation.cfm?id=2660970&dl=ACM&coll=DL (https://dl.acm.org/citation.cfm?id=2660970&dl=ACM&coll=DL)

1.3 Real World / Business Objectives and Constraints

1. Predict as many tags as possible with high precision and recall.
2. Incorrect tags could impact customer experience on StackOverflow.
3. No strict latency constraints.

2. Machine Learning problem

2.1 Data

2.1.1 Data Overview

Refer: https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction/data (https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction/data)
All of the data is in 2 files: Train and Test.

Train.csv contains 4 columns: Id,Title,Body,Tags.

Test.csv contains the same columns but without the Tags, which you are to predict.

Size of Train.csv - 6.75GB

Size of Test.csv - 2GB

Number of rows in Train.csv = 6034195

The questions are randomized and contains a mix of verbose text sites as well as sites related to math and programming. The number of questions from each site may vary, and no filtering has been performed on the questions (such as closed questions).

Data Field Explaination

Dataset contains 6,034,195 rows. The columns in the table are:

Id - Unique identifier for each question

Title - The question's title

Body - The body of the question

Tags - The tags associated with the question in a space-seperated format (all lowe
rcase, should not contain tabs '\t' or ampersands '&')

2.1.2 Example Data point

Title: Implementing Boundary Value Analysis of Software Testing in a C++ program?
Body :

```
#include<
iostream>\n
#include<
stdlib.h>\n\n
using namespace std;\n\n
int main()\n
{\n
        int n,a[n],x,c,u[n],m[n],e[n][4];\n
        cout<<"Enter the number of variables";\n         cin>>n;\n
\n
        cout<<"Enter the Lower, and Upper Limits of the variable
s";\n
        for(int y=1; y<n+1; y++)\n
        {\n
           cin>>m[y];\n
           cin>>u[y];\n
        }\n
        for(x=1; x<n+1; x++)\n
        {\n
           a[x] = (m[x] + u[x])/2;\n
        }\n
        c=(n*4)-4;\n
        for(int a1=1; a1<n+1; a1++)\n
        {\n\n
           e[a1][0] = m[a1];\n
           e[a1][1] = m[a1]+1;\n
           e[a1][2] = u[a1]-1;\n
           e[a1][3] = u[a1];\n
        }\n
        for(int i=1; i<n+1; i++)\n
        {\n
           for(int l=1; l<=i; l++)\n
           {\n
               if(l!=1)\n
               {\n
                   cout<<a[l]<<"\\t";\n
               }\n
           }\n
           for(int j=0; j<4; j++)\n
           {\n
               cout<<e[i][j];\n
               for(int k=0; k<n-(i+1); k++)\n
               {\n
                   cout<<a[k]<<"\\t";\n
               }\n
               cout<<"\\n";\n
           }\n
        }    \n\n
        system("PAUSE");\n
        return 0;    \n
```

```
            }\n


\n\n

        <p>The answer should come in the form of a table like</p>\n\n
        <pre><code>
        1            50             50\n
        2            50             50\n
        99           50             50\n
        100          50             50\n
        50           1              50\n
        50           2              50\n
        50           99             50\n
        50           100            50\n
        50           50             1\n
        50           50             2\n
        50           50             99\n
        50           50             100\n
        </code></pre>\n\n
        <p>if the no of inputs is 3 and their ranges are\n
        1,100\n
        1,100\n
        1,100\n
        (could be varied too)</p>\n\n
        <p>The output is not coming,can anyone correct the code or tell me what
    \'s wrong?</p>\n'

Tags : 'c++ c'
```

2.2 Mapping the real-world problem to a Machine Learning Problem

2.2.1 Type of Machine Learning Problem

It is a multi-label classification problem
Multi-label Classification: Multilabel classification assigns to each sample a set of target labels. This can be thought as predicting properties of a data-point that are not mutually exclusive, such as topics that are relevant for a document. A question on Stackoverflow might be about any of C, Pointers, FileIO and/or memory-management at the same time or none of these.
__Credit__: http://scikit-learn.org/stable/modules/multiclass.html

2.2.2 Performance metric

Micro-Averaged F1-Score (Mean F Score) : The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal. The formula for the F1 score is:

*F1 = 2 * (precision * recall) / (precision + recall)*

In the multi-class and multi-label case, this is the weighted average of the F1 score of each class.

'Micro f1 score':
Calculate metrics globally by counting the total true positives, false negatives and false positives. This is a better metric when we have class imbalance.

'Macro f1 score':
Calculate metrics for each label, and find their unweighted mean. This does not take label imbalance into account.

https://www.kaggle.com/wiki/MeanFScore (https://www.kaggle.com/wiki/MeanFScore)
http://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html (http://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html)

Hamming loss : The Hamming loss is the fraction of labels that are incorrectly predicted.
https://www.kaggle.com/wiki/HammingLoss (https://www.kaggle.com/wiki/HammingLoss)

3. Exploratory Data Analysis

3.1 Data Loading and Cleaning

3.1.1 Using Pandas with SQLite to Load the data

In [7]:

```
#checking file is present or not
if not os.path.isfile('train.db'):
    start = datetime.now()
    #creating database file
    disk_engine = create_engine('sqlite:///train.db')
    start = dt.datetime.now()
    #instead of loading all data i am breaking data into chunks to load
    chunksize = 18000
    j = 0
    index_start = 1
    #Reading csv file
    for df in pd.read_csv('Train.csv', names=['Id', 'Title', 'Body', 'Tags'], chunksize=chu
        df.index += index_start
        j+=1
        print('{} rows'.format(j*chunksize))
        df.to_sql('data', disk_engine, if_exists='append')
        index_start = df.index[-1] + 1
    print("Time taken to run this cell :", datetime.now() - start)
```

```
18000 rows
36000 rows
54000 rows
72000 rows
90000 rows
108000 rows
126000 rows
144000 rows
162000 rows
180000 rows
198000 rows
216000 rows
234000 rows
252000 rows
270000 rows
288000 rows
306000 rows
324000 rows
342000 rows
```

3.1.2 Counting the number of rows

In [8]:

```
#checking file is present or not
if os.path.isfile('train.db'):
    start = datetime.now()
    #creating a database file
    con = sqlite3.connect('train.db')
    #Reading sql file
    num_rows = pd.read_sql_query("""SELECT count(*) FROM data""", con)
    #Always remember to close the database
    print("Number of rows in the database :","\n",num_rows['count(*)'].values[0])
    con.close()
    print("Time taken to count the number of rows :", datetime.now() - start)
else:
    print("Please download the train.db file from drive or run the above cell to genarate t
```

```
Number of rows in the database :
 6034196
Time taken to count the number of rows : 0:00:00.082396
```

3.1.3 Checking for duplicates

In [9]:

```
#Learn SQL: https://www.w3schools.com/sql/default.asp
#Checking file is present or not
if os.path.isfile('train.db'):
    start = datetime.now()
    #connecting database file
    con = sqlite3.connect('train.db')
    #Reading file
    df_no_dup = pd.read_sql_query('SELECT Title, Body, Tags, COUNT(*) as cnt_dup FROM data
    #dont forget to close file always close file
    con.close()
    print("Time taken to run this cell :", datetime.now() - start)
else:
    print("Please download the train.db file from drive or run the first to genarate train.
```

```
Time taken to run this cell : 0:01:19.920156
```

In [10]:

```
#printing first five rows
df_no_dup.head()
# we can observe that there are duplicates
```

Out[10]:

	Title	Body	Tags	cnt_dup
0	Implementing Boundary Value Analysis of S...	<pre> <code>#include<iostream>\n#include&...	c++ c	1
1	Dynamic Datagrid Binding in Silverlight?	I should do binding for datagrid dynamicall...	c# silverlight data-binding	1
2	Dynamic Datagrid Binding in Silverlight?	I should do binding for datagrid dynamicall...	c# silverlight data-binding columns	1
3	java.lang.NoClassDefFoundError: javax/serv...	I followed the guide in <a href="http://sta...	jsp jstl	1
4	java.sql.SQLException:[Microsoft] [ODBC Dri...	I use the following code \n\n<pre> <code>...	java jdbc	2

observation:-

see 2nd and 3rd row It had duplicate questions i have to drop duplicates questions

In [11]:

```
print("number of duplicate questions :", num_rows['count(*)'].values[0]- df_no_dup.shape[0]
```

```
number of duplicate questions : 1827881 ( 30.292038906260256 % )
```

In [12]:

```
# number of times each question appeared in our database
df_no_dup.cnt_dup.value_counts()
```

Out[12]:

```
1    2656284
2    1272336
3     277575
4         90
5         25
6          5
Name: cnt_dup, dtype: int64
```

In [13]:

```
#Checking null values are present
df_no_dup["Tags"].isnull().sum()
```

Out[13]:

7

In [14]:

```
df_no_dup[df_no_dup["Tags"].isnull()]
```

Out[14]:

	Title	Body	Tags	cnt_dup
777547	Do we really need NULL?	<blockquote>\n Possible Duplicate:...	None	1
962680	Find all values that are not null and not in a...	I am running into a problem which results i...	None	1
1126558	Handle NullObjects	I have done quite a bit of research on best...	None	1
1256102	How do Germans call null	In german null means 0, so how do they call...	None	1
2430668	Page cannot be null. Please ensure that this o...	I get this error when i remove dynamically ...	None	1
3329908	What is the difference between NULL and "0"?	What is the difference from NULL and "0"?</...	None	1
3551595	a bit of difference between null and space	I was just reading this quote \n\n<block...	None	2

In [15]:

```
#it drops the missing values
df_no_dup = df_no_dup.dropna(axis = 0)
```

In [16]:

```
df_no_dup = df_no_dup.drop_duplicates(subset={'Title', 'Body', 'Tags'})
```

In [17]:

```
start = datetime.now()
#Splitting text in tags
df_no_dup["tag_count"] = df_no_dup["Tags"].apply(lambda text: len(text.split(" ")) if text!
# adding a new feature number of tags per question
print("Time taken to run this cell :", datetime.now() - start)
df_no_dup.head()
```

```
Time taken to run this cell : 0:00:02.905762
```

Out[17]:

	Title	Body	Tags	cnt_dup	t
0	Implementing Boundary Value Analysis of S...	<pre> <code>#include<iostream>\n#include&...	c++ c	1	
1	Dynamic Datagrid Binding in Silverlight?	I should do binding for datagrid dynamicall...	c# silverlight data-binding	1	
2	Dynamic Datagrid Binding in Silverlight?	I should do binding for datagrid dynamicall...	c# silverlight data-binding columns	1	
3	java.lang.NoClassDefFoundError: javax/serv...	I followed the guide in <a href="http://sta...	jsp jstl	1	
4	java.sql.SQLException:[Microsoft] [ODBC Dri...	I use the following code \n\n<pre> <code>...	java jdbc	2	

In [18]:

```
#knowing length and converts the specified value into a string.
df_no_dup['Titlelen'] = df_no_dup['Title'].str.len()
```

In [19]:

```
#splitting the row
df_no_dup['Title_words'] = df_no_dup['Title'].apply(lambda row: len(row.split(" ")))
```

In [20]:

```
#splitting the row
df_no_dup['Body_words'] = df_no_dup['Body'].apply(lambda row: len(row.split(" ")))
```

In [21]:

```
def normalized_word_Common(row):
        #converting characters into lower case and returns a copy of the string with both l
    w1 = set(map(lambda word: word.lower().strip(), row['Title'].split(" ")))
    w2 = set(map(lambda word: word.lower().strip(), row['Body'].split(" ")))
    return 1.0 * len(w1 & w2)
df_no_dup['word_Common_Title and body'] = df_no_dup.apply(normalized_word_Common, axis=1)


#building function
def normalized_word_Total(row):
    w1 = set(map(lambda word: word.lower().strip(), row['Title'].split(" ")))
    w2 = set(map(lambda word: word.lower().strip(), row['Body'].split(" ")))
    return 1.0 * (len(w1) + len(w2))
df_no_dup['word_Total'] = df_no_dup.apply(normalized_word_Total, axis=1)
#building function
def normalized_word_share(row):
    w1 = set(map(lambda word: word.lower().strip(), row['Title'].split(" ")))
    w2 = set(map(lambda word: word.lower().strip(), row['Body'].split(" ")))
    return 1.0 * len(w1 & w2)/(len(w1) + len(w2))
df_no_dup['word_share'] = df_no_dup.apply(normalized_word_share, axis=1)
#combining two variables
df_no_dup['freq_Title'] = df_no_dup.groupby('Title')['Title'].transform('count')
df_no_dup['freq_Body'] = df_no_dup.groupby('Body')['Body'].transform('count')

df_no_dup['freq_q1+q2'] = df_no_dup['freq_Title']+df_no_dup['freq_Body']
df_no_dup['freq_q1-q2'] = abs(df_no_dup['freq_Title']-df_no_dup['freq_Body'])
#creating csv file
df_no_dup.to_csv("df_no_dup_without_preprocessing_train.csv", index=False)
#printing first five rows
df_no_dup.head()
```

Out[21]:

	Title	Body	Tags	cnt_dup	t
0	Implementing Boundary Value Analysis of S...	<pre> <code>#include<iostream>\n#include&...	c++ c	1	
1	Dynamic Datagrid Binding in Silverlight?	I should do binding for datagrid dynamicall...	c# silverlight data-binding	1	
2	Dynamic Datagrid Binding in Silverlight?	I should do binding for datagrid dynamicall...	c# silverlight data-binding columns	1	
3	java.lang.NoClassDefFoundError: javax/serv...	I followed the guide in <a href="http://sta...	jsp jstl	1	
4	java.sql.SQLException:[Microsoft] [ODBC Dri...	I use the following code \n\n<pre> <code>...	java jdbc	2	

3.3.1 Analysis of some of the extracted features

- Here are some questions have only one single words.

In [22]:

```
#printing minimum length of questions in title
print ("Minimum length of the Text in Title : " , min(df_no_dup['Title_words']))
#printing minimum length of questions in body
print ("Minimum length of the Text in Body : " , min(df_no_dup['Body_words']))
```

```
Minimum length of the Text in Title :  1
Minimum length of the Text in Body :  1
```

In [23]:

```
#Removing question without any tags¶

df_no_dup = df_no_dup[df_no_dup['tag_count']!=0]
```

In [24]:

```
# distribution of number of tags per question
df_no_dup.tag_count.value_counts()
```

Out[24]:

```
3    1206157
2    1111706
4     814996
1     568291
5     505158
Name: tag_count, dtype: int64
```

In [25]:

```
#Creating a new database with no duplicates
#checking file is present or not
if not os.path.isfile('train_no_dup.db'):
    #creating database file
    disk_dup = create_engine("sqlite:///train_no_dup.db")
    #creating dataframe and placing values
    no_dup = pd.DataFrame(df_no_dup, columns=['Title', 'Body', 'Tags','cnt_dup','tag_count'
    no_dup.to_sql('no_dup_train',disk_dup)
```

In [26]:

```
#This method seems more appropriate to work with this much data.
#creating the connection with database file.
#Checking if file is present or not
if os.path.isfile('train_no_dup.db'):
    start = datetime.now()
    #connecting sql fime
    con = sqlite3.connect('train_no_dup.db')
    #Reaving csv file
    tag_data = pd.read_sql_query("""SELECT Tags FROM no_dup_train""", con)
    #Always remember to close the database
    con.close()

    # Let's now drop unwanted column.
    tag_data.drop(tag_data.index[0], inplace=True)
    #Printing first 5 columns from our data frame
    tag_data.head()
    print("Time taken to run this cell :", datetime.now() - start)
else:
    print("Please download the train.db file from drive or run the above cells to genarate
```

```
Time taken to run this cell : 0:00:06.961244
```

Observations

There were almost 30% questions which were duplicates. So the first thing we did, is remove the duplicate questions from the actual dataset and save it in a new dataset.

2656284 questions have occured only 1 time. 1272336 occurs 2 times. 277575 questions occurs 3 times and so on.

There are 1206157 questions which have 3 tags, 1111706 have 2 tags, 814996 questions have 4 tags, 568298 questions have one tag & 505158 questions have 5 tags.

3.2 Analysis of Tags

3.2.1 Total number of unique tags

In [27]:

```
# removing 1st row its extra
df_no_dup=df_no_dup.drop(df_no_dup.index[0])
```

In [28]:

```
#by default 'split()' will tokenize each tag using space.
vectorizer = CountVectorizer(tokenizer = lambda x: x.split())
# fit_transform() does two functions: First, it fits the model
# and learns the vocabulary; second, it transforms our training data
# into feature vectors. The input to fit_transform should be a list of strings.
tag_dtm = vectorizer.fit_transform(tag_data['Tags'])
```

In [29]:

```
print("Number of data points :", tag_dtm.shape[0])
print("Number of unique tags :", tag_dtm.shape[1])
```

```
Number of data points : 4206307
Number of unique tags : 42048
```

In [30]:

```
#'get_feature_name()' gives us the vocabulary.
tags = vectorizer.get_feature_names()
#Lets look at the tags we have.
print("Some of the tags we have :", tags[:10])
```

```
Some of the tags we have : ['.a', '.app', '.asp.net-mvc', '.aspxauth', '.bas
h-profile', '.class-file', '.cs-file', '.doc', '.drv', '.ds-store']
```

3.2.3 Number of times a tag appeared

In [31]:

```
# https://stackoverflow.com/questions/15115765/how-to-access-sparse-matrix-elements
#Lets now store the document term matrix in a dictionary.
freqs = tag_dtm.sum(axis=0).A1
result = dict(zip(tags, freqs))
```

In [32]:

```
#Saving this dictionary to csv files.
if not os.path.isfile('tag_counts_dict_dtm.csv'):
    with open('tag_counts_dict_dtm.csv', 'w') as csv_file:
        writer = csv.writer(csv_file)
        for key, value in result.items():
            writer.writerow([key, value])
tag_df = pd.read_csv("tag_counts_dict_dtm.csv", names=['Tags', 'Counts'])
tag_df.head()
```

Out[32]:

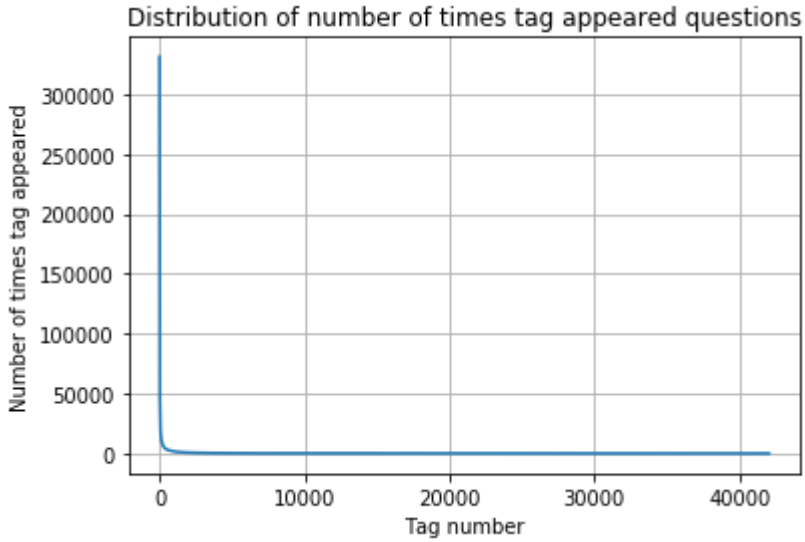
	Tags	Counts
0	.a	18
1	.app	37
2	.asp.net-mvc	1
3	.aspxauth	21
4	.bash-profile	138

In [33]:

```
tag_df_sorted = tag_df.sort_values(['Counts'], ascending=False)
tag_counts = tag_df_sorted['Counts'].values
```

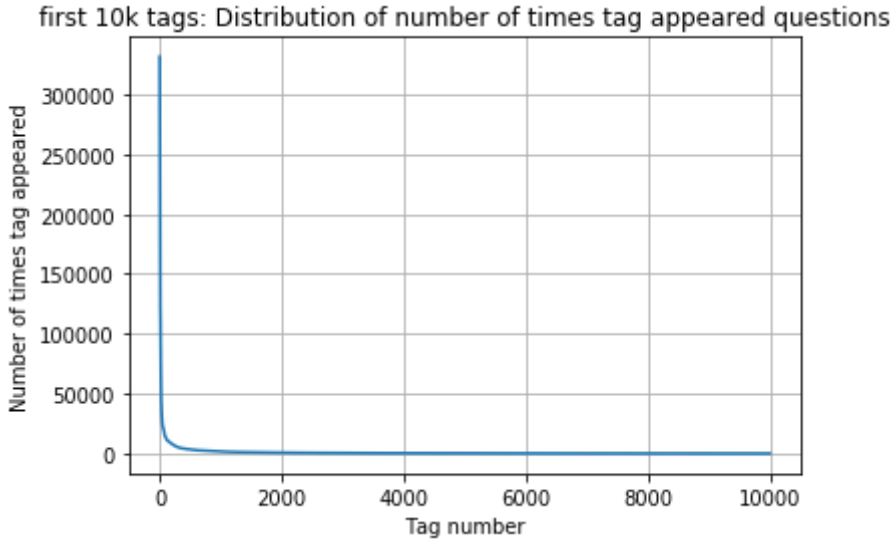
In [34]:

```
plt.plot(tag_counts)
plt.title("Distribution of number of times tag appeared questions")
plt.grid()
plt.xlabel("Tag number")
plt.ylabel("Number of times tag appeared")
plt.show()
```



In [35]:

```
plt.plot(tag_counts[0:10000])
plt.title('first 10k tags: Distribution of number of times tag appeared questions')
plt.grid()
plt.xlabel("Tag number")
plt.ylabel("Number of times tag appeared")
plt.show()
print(len(tag_counts[0:10000:25]), tag_counts[0:10000:25])
```

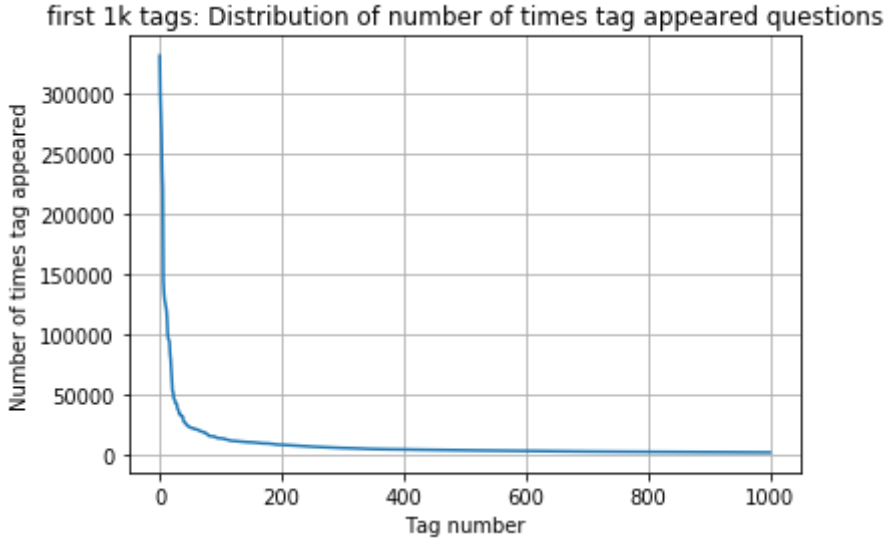



```
400 [331505  44829  22429  17728  13364  11162  10029   9148   8054   7151
   6466   5865   5370   4983   4526   4281   4144   3929   3750   3593
   3453   3299   3123   2986   2891   2738   2647   2527   2431   2331
   2259   2186   2097   2020   1959   1900   1828   1770   1723   1673
   1631   1574   1532   1479   1448   1406   1365   1328   1300   1266
   1245   1222   1197   1181   1158   1139   1121   1101   1076   1056
   1038   1023   1006    983    966    952    938    926    911    891
    882    869    856    841    830    816    804    789    779    770
    752    743    733    725    712    702    688    678    671    658
    650    643    634    627    616    607    598    589    583    577
    568    559    552    545    540    533    526    518    512    506
    500    495    490    485    480    477    469    465    457    450
    447    442    437    432    426    422    418    413    408    403
    398    393    388    385    381    378    374    370    367    365
    361    357    354    350    347    344    342    339    336    332
    330    326    323    319    315    312    309    307    304    301
    299    296    293    291    289    286    284    281    278    276
    275    272    270    268    265    262    260    258    256    254
    252    250    249    247    245    243    241    239    238    236
    234    233    232    230    228    226    224    222    220    219
    217    215    214    212    210    209    207    205    204    203
    201    200    199    198    196    194    193    192    191    189
    188    186    185    183    182    181    180    179    178    177
    175    174    172    171    170    169    168    167    166    165
    164    162    161    160    159    158    157    156    156    155
    154    153    152    151    150    149    149    148    147    146
    145    144    143    142    142    141    140    139    138    137
    137    136    135    134    134    133    132    131    130    130
    129    128    128    127    126    126    125    124    124    123
    123    122    122    121    120    120    119    118    118    117
    117    116    116    115    115    114    113    113    112    111
    111    110    109    109    108    108    107    106    106    106
    105    105    104    104    103    103    102    102    101    101
    100    100     99     99     98     98     97     97     96     96
```

```
      95     95     94     94     93     93     93     92     92     91
      91     90     90     89     89     88     88     87     87     86
      86     86     85     85     84     84     83     83     83     82
      82     82     81     81     80     80     80     79     79     78
      78     78     78     77     77     76     76     76     75     75
      75     74     74     74     73     73     73     73     72     72]
```

In [36]:

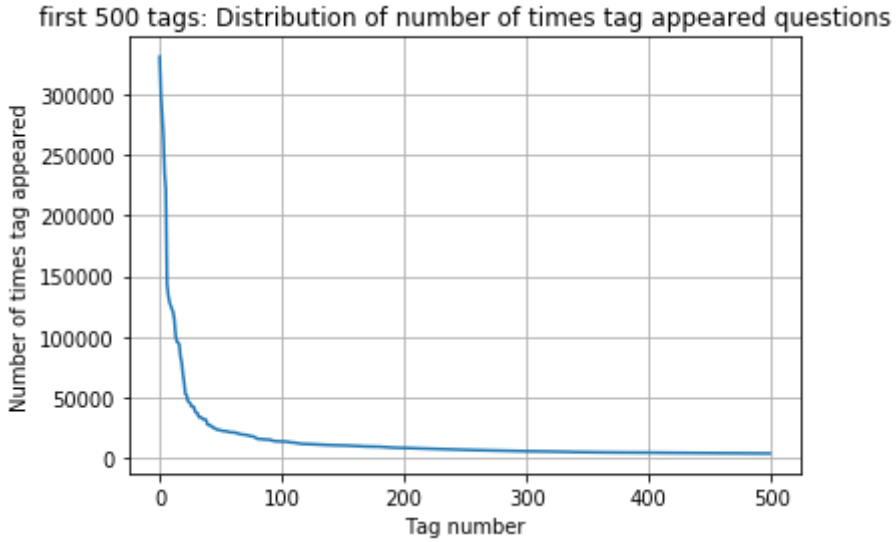
```
plt.plot(tag_counts[0:1000])
plt.title('first 1k tags: Distribution of number of times tag appeared questions')
plt.grid()
plt.xlabel("Tag number")
plt.ylabel("Number of times tag appeared")
plt.show()
print(len(tag_counts[0:1000:5]), tag_counts[0:1000:5])
```




```
200 [331505 221533 122769  95160  62023  44829  37170  31897  26925  24537
  22429  21820  20957  19758  18905  17728  15533  15097  14884  13703
  13364  13157  12407  11658  11228  11162  10863  10600  10350  10224
  10029   9884   9719   9411   9252   9148   9040   8617   8361   8163
   8054   7867   7702   7564   7274   7151   7052   6847   6656   6553
   6466   6291   6183   6093   5971   5865   5760   5577   5490   5411
   5370   5283   5207   5107   5066   4983   4891   4785   4658   4549
   4526   4487   4429   4335   4310   4281   4239   4228   4195   4159
   4144   4088   4050   4002   3957   3929   3874   3849   3818   3797
   3750   3703   3685   3658   3615   3593   3564   3521   3505   3483
   3453   3427   3396   3363   3326   3299   3272   3232   3196   3168
   3123   3094   3073   3050   3012   2986   2983   2953   2934   2903
   2891   2844   2819   2784   2754   2738   2726   2708   2681   2669
   2647   2621   2604   2594   2556   2527   2510   2482   2460   2444
   2431   2409   2395   2380   2363   2331   2312   2297   2290   2281
   2259   2246   2222   2211   2198   2186   2162   2142   2132   2107
   2097   2078   2057   2045   2036   2020   2011   1994   1971   1965
   1959   1952   1940   1932   1912   1900   1879   1865   1855   1841
   1828   1821   1813   1801   1782   1770   1760   1747   1741   1734
   1723   1707   1697   1688   1683   1673   1665   1656   1646   1639]
```

In [37]:

```
plt.plot(tag_counts[0:500])
plt.title('first 500 tags: Distribution of number of times tag appeared questions')
plt.grid()
plt.xlabel("Tag number")
plt.ylabel("Number of times tag appeared")
plt.show()
print(len(tag_counts[0:500:5]), tag_counts[0:500:5])
```



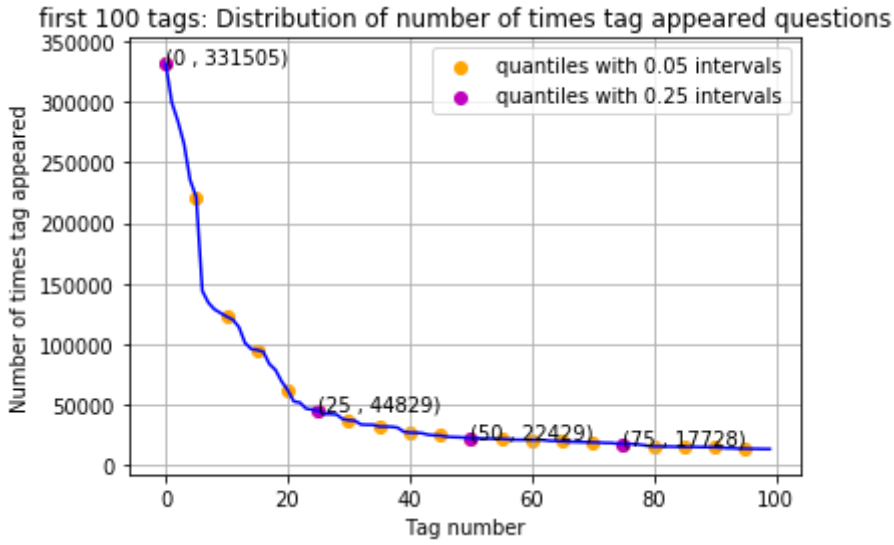

```
100 [331505 221533 122769  95160  62023  44829  37170  31897  26925  24537
  22429  21820  20957  19758  18905  17728  15533  15097  14884  13703
  13364  13157  12407  11658  11228  11162  10863  10600  10350  10224
  10029   9884   9719   9411   9252   9148   9040   8617   8361   8163
   8054   7867   7702   7564   7274   7151   7052   6847   6656   6553
   6466   6291   6183   6093   5971   5865   5760   5577   5490   5411
   5370   5283   5207   5107   5066   4983   4891   4785   4658   4549
   4526   4487   4429   4335   4310   4281   4239   4228   4195   4159
   4144   4088   4050   4002   3957   3929   3874   3849   3818   3797
   3750   3703   3685   3658   3615   3593   3564   3521   3505   3483]
```

In [38]:

```
plt.plot(tag_counts[0:100], c='b')
plt.scatter(x=list(range(0,100,5)), y=tag_counts[0:100:5], c='orange', label="quantiles wit
# quantiles with 0.25 difference
plt.scatter(x=list(range(0,100,25)), y=tag_counts[0:100:25], c='m', label = "quantiles with

for x,y in zip(list(range(0,100,25)), tag_counts[0:100:25]):
    plt.annotate(s="({} , {})".format(x,y), xy=(x,y), xytext=(x-0.05, y+500))

plt.title('first 100 tags: Distribution of number of times tag appeared questions')
plt.grid()
plt.xlabel("Tag number")
plt.ylabel("Number of times tag appeared")
plt.legend()
plt.show()
print(len(tag_counts[0:100:5]), tag_counts[0:100:5])
```




```
20 [331505 221533 122769  95160  62023  44829  37170  31897  26925  24537
  22429  21820  20957  19758  18905  17728  15533  15097  14884  13703]
```

In [39]:

```
# Store tags greater than 10K in one list
lst_tags_gt_10k = tag_df[tag_df.Counts>10000].Tags
#Print the length of the list
print ('{} Tags are used more than 10000 times'.format(len(lst_tags_gt_10k)))
# Store tags greater than 100K in one list
lst_tags_gt_100k = tag_df[tag_df.Counts>100000].Tags
#Print the length of the list.
print ('{} Tags are used more than 100000 times'.format(len(lst_tags_gt_100k)))
```

```
153 Tags are used more than 10000 times
14 Tags are used more than 100000 times
```

Observations:

1. There are total 153 tags which are used more than 10000 times.
2. 14 tags are used more than 100000 times.
3. Most frequent tag (i.e. c#) is used 331505 times.
4. Since some tags occur much more frequenctly than others, Micro-averaged F1-score is the appropriate metric for this probelm.

3.2.4 Tags Per Question

In [40]:

```
#Storing the count of tag in each question in list 'tag_count'
tag_quest_count = tag_dtm.sum(axis=1).tolist()
#Converting each value in the 'tag_quest_count' to integer.
tag_quest_count=[int(j) for i in tag_quest_count for j in i]
print ('We have total {} datapoints.'.format(len(tag_quest_count)))

print(tag_quest_count[:5])
```

```
We have total 4206307 datapoints.
[3, 4, 2, 2, 3]
```

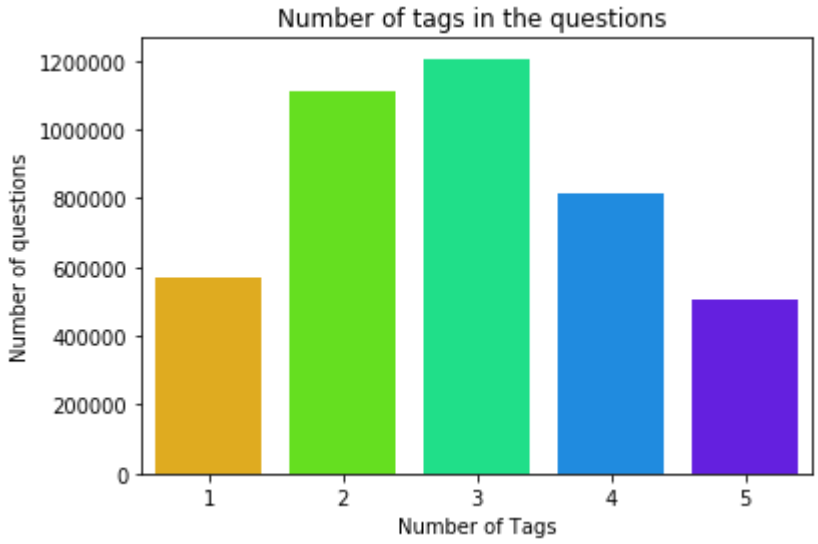
In [41]:

```
print( "Maximum number of tags per question: %d"%max(tag_quest_count))
print( "Minimum number of tags per question: %d"%min(tag_quest_count))
print( "Avg. number of tags per question: %f"% ((sum(tag_quest_count)*1.0)/len(tag_quest_cc
```

```
Maximum number of tags per question: 5
Minimum number of tags per question: 1
Avg. number of tags per question: 2.899443
```

In [42]:

```
sns.countplot(tag_quest_count, palette='gist_rainbow')
plt.title("Number of tags in the questions ")
plt.xlabel("Number of Tags")
plt.ylabel("Number of questions")
plt.show()
```



Observations:

1. Maximum number of tags per question: 5
2. Minimum number of tags per question: 1
3. Avg. number of tags per question: 2.899
4. Most of the questions are having 2 or 3 tags

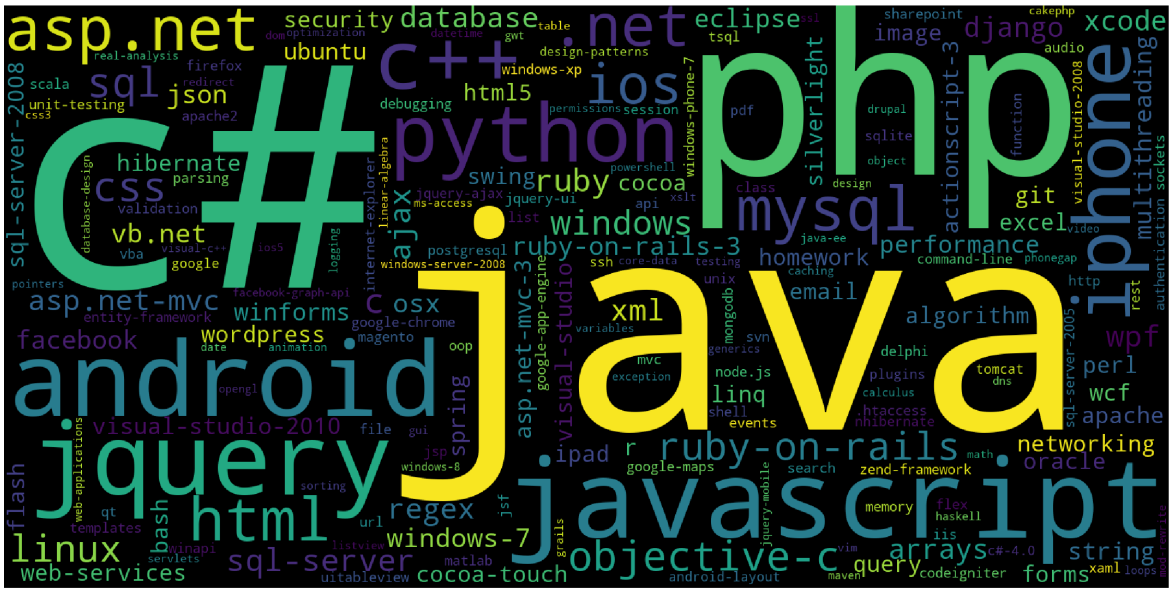
3.2.5 Most Frequent Tags

In [43]:

```
# Ploting word cloud
start = datetime.now()

# Lets first convert the 'result' dictionary to 'list of tuples'
tup = dict(result.items())
#Initializing WordCloud using frequencies of tags.
wordcloud = WordCloud(    background_color='black',
                          width=1600,
                          height=800,
                    ).generate_from_frequencies(tup)

fig = plt.figure(figsize=(30,20))
plt.imshow(wordcloud)
plt.axis('off')
plt.tight_layout(pad=0)
fig.savefig("tag.png")
plt.show()
print("Time taken to run this cell :", datetime.now() - start)
```



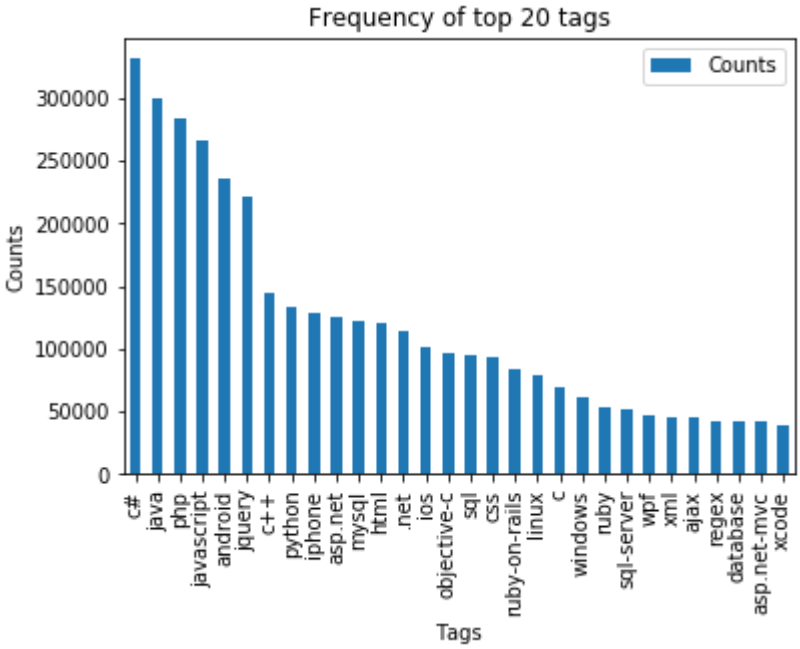
Time taken to run this cell : 0:00:03.542524

Observations:
A look at the word cloud shows that "c#", "java", "php", "asp.net", "javascript", "c++" are some of the most frequent tags.

3.2.6 The top 20 tags

In [44]:

```
i=np.arange(30)
tag_df_sorted.head(30).plot(kind='bar')
plt.title('Frequency of top 20 tags')
plt.xticks(i, tag_df_sorted['Tags'])
plt.xlabel('Tags')
plt.ylabel('Counts')
plt.show()
```



Observations:

1. Majority of the most frequent tags are programming language.
2. C# is the top most frequent programming language.
3. Android, IOS, Linux and windows are among the top most frequent operating systems.

4. Cleaning and preprocessing of Questions

4.1 Preprocessing of questions

1. Separate Code from Body
2. Sampling 0.5million datapoints
3. Remove Spcial characters from Question title and description (not in code)
4. **Give more weightage to title : Add title three times to the question**
5. Remove stop words (Except 'C')
6. Remove HTML Tags
7. Convert all the characters into small letters
8. Use SnowballStemmer to stem the words

In [45]:

```
import nltk
nltk.download('punkt')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     /home/saikrishna6680/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to
[nltk_data]     /home/saikrishna6680/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

Out[45]:

```
True
```

In [46]:

```
def striphtml(data):
    cleanr = re.compile('<.*?>')
    cleantext = re.sub(cleanr, ' ', str(data))
    return cleantext
stop_words = set(stopwords.words('english'))
stemmer = SnowballStemmer("english")
```

In [47]:

```
def create_connection(db_file):
    """ create a database connection to the SQLite database
        specified by db_file
    :param db_file: database file
    :return: Connection object or None
    """
    try:
        conn = sqlite3.connect(db_file)
        return conn
    except Error as e:
        print(e)

    return None

def create_table(conn, create_table_sql):
    """ create a table from the create_table_sql statement
    :param conn: Connection object
    :param create_table_sql: a CREATE TABLE statement
    :return:
    """
    try:
        c = conn.cursor()
        c.execute(create_table_sql)
    except Error as e:
        print(e)

def checkTableExists(dbcon):
    cursr = dbcon.cursor()
    str = "select name from sqlite_master where type='table'"
    table_names = cursr.execute(str)
    print("Tables in the databse:")
    tables =table_names.fetchall()
    print(tables[0][0])
    return(len(tables))

def create_database_table(database, query):
    conn = create_connection(database)
    if conn is not None:
        create_table(conn, query)
        checkTableExists(conn)
    else:
        print("Error! cannot create the database connection.")
    conn.close()

sql_create_table = """CREATE TABLE IF NOT EXISTS QuestionsProcessed (question text NOT NULL
create_database_table("3times_weighted_Title.db", sql_create_table)
```

```
Tables in the databse:
QuestionsProcessed
```

In [50]:

```
7969025/1MUAVbg0jinwAGi9zwLJDo1K1wdRXKCFB?e=download&nonce=4jh2chh5dleum&user=04563076751847
```

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Curr
ent
                                 Dload  Upload   Total   Spent    Left  Spee
d
100  984M    0  984M    0     0   169M      0 --:--:--  0:00:05 --:--:--  19
9M
```

In [51]:

```
# http://www.sqlitetutorial.net/sqlite-delete/
# https://stackoverflow.com/questions/2279706/select-random-row-from-a-sqlite-table
start = datetime.now()
read_db = 'train_no_dup.db'
write_db = 'Processed.db'
if os.path.isfile(read_db):
    conn_r = create_connection(read_db)
    if conn_r is not None:
        reader =conn_r.cursor()
        reader.execute("SELECT Title, Body,Tags From no_dup_train LIMIT 500000;")

if os.path.isfile(write_db):
    conn_w = create_connection(write_db)
    if conn_w is not None:
        tables = checkTableExists(conn_w)
        writer =conn_w.cursor()
        if tables != 0:
            writer.execute("DELETE FROM QuestionsProcessed WHERE 1")
            print("Cleared All the rows")
print("Time taken to run this cell :", datetime.now() - start)
```

```
Tables in the databse:
QuestionsProcessed
Cleared All the rows
Time taken to run this cell : 0:00:11.981732
```

In [52]:

```
#http://www.bernzilla.com/2008/05/13/selecting-a-random-row-from-an-sqlite-table/
from __future__ import division

import csv
import pip
start = datetime.now()
preprocessed_data_list=[]
reader.fetchone()
questions_with_code=0
len_pre=0
len_post=0
questions_proccesed = 0
for row in reader:

    title, question, tags, = row[0], row[1], row[2]

    if '<code>' in question:
        questions_with_code+=1
        is_code = 1
    x = len(question)+len(title)
    len_pre+=x

    code = str(re.findall(r'<code>(.*?)</code>', question, flags=re.DOTALL))

    question=re.sub('<code>(.*?)</code>', '', question, flags=re.MULTILINE|re.DOTALL)
    question=striphtml(question.encode('utf-8'))

    title=title.encode('utf-8')

    question=str(title)+" "+str(question)
    question=re.sub(r'[^A-Za-z]+',' ',question)
    words=word_tokenize(str(question.lower()))

    #Removing all single letter and and stopwords from question except for the letter 'c'
    question=' '.join(str(stemmer.stem(j)) for j in words if j not in stop_words and (len(j

    len_post+=len(question)
    tup = (question,code,tags,x,len(question),is_code)
    questions_proccesed += 1
    writer.execute("insert into QuestionsProcessed(question,code,tags,words_pre,words_post,
    if (questions_proccesed%100000==0):
        print("number of questions completed=",questions_proccesed)

no_dup_avg_len_pre=(len_pre*1.0)/questions_proccesed
no_dup_avg_len_post=(len_post*1.0)/questions_proccesed

print( "Avg. length of questions(Title+Body) before processing: %d"%no_dup_avg_len_pre)
print( "Avg. length of questions(Title+Body) after processing: %d"%no_dup_avg_len_post)
print ("Percent of questions containing code: %d"%((questions_with_code*100.0)/questions_pr

print("Time taken to run this cell :", datetime.now() - start)
```

```
number of questions completed= 100000
number of questions completed= 200000
number of questions completed= 300000
number of questions completed= 400000
Avg. length of questions(Title+Body) before processing: 1239
Avg. length of questions(Title+Body) after processing: 341
```

```
Percent of questions containing code: 57
Time taken to run this cell : 0:11:06.873162
```

In [53]:

```
# never forget to close the conections or else we will end up with database locks
conn_r.commit()
conn_w.commit()
conn_r.close()
conn_w.close()
```

__ Sample quesitons after preprocessing of data __

In [54]:

```
if os.path.isfile(write_db):
    conn_r = create_connection(write_db)
    if conn_r is not None:
        reader =conn_r.cursor()
        reader.execute("SELECT question From QuestionsProcessed LIMIT 10")
        print("Questions after preprocessed")
        print('='*100)
        reader.fetchone()
        for row in reader:
            print(row)
            print('-'*100)
conn_r.commit()
conn_r.close()
```

```
Questions after preprocessed
====================================================================================
========================
('dynam datagrid bind silverlight bind datagrid dynam code wrote code debug
code block seem bind correct grid come column form come grid column although
necessari bind nthank repli advanc',)
------------------------------------------------------------------------------------
------------------------
('java lang noclassdeffounderror javax servlet jsp tagext taglibraryvalid fo
llow guid link instal jstl got follow error tri launch jsp page java lang no
classdeffounderror javax servlet jsp tagext taglibraryvalid taglib declar in
stal jstl tomcat webapp tri project work also tri version jstl still messag
caus solv',)
------------------------------------------------------------------------------------
------------------------
('java sql sqlexcept microsoft odbc driver manag invalid descriptor index us
e follow code display caus solv',)
------------------------------------------------------------------------------------
------------------------
('better way updat feed fb php sdk novic facebook api read mani tutori still
confus find post feed api method like correct second way use curl someth lik
e way better',)
------------------------------------------------------------------------------------
------------------------
('btnadd click event open two window record ad open window search aspx use c
ode hav add button search aspx nwhen insert record btnadd click event open a
noth window nafter insert record close window',)
------------------------------------------------------------------------------------
------------------------
('sql inject issu prevent correct form submiss php check everyth think make
sure input field safe type sql inject good news safe bad news one tag mess f
orm submiss place even touch life figur exact html use templat file forgiv o
kay entir php script get execut see data post none forum field post problem
use someth titl field none data get post current use print post see submit n
oth work flawless statement though also mention script work flawless local m
achin use host come across problem state list input test mess',)
------------------------------------------------------------------------------------
------------------------
('countabl subaddit lebesgu measur let lbrace rbrace sequenc set sigma algeb
ra mathcal want show left bigcup right leq sum left right countabl addit mea
sur defin set sigma algebra mathcal think use monoton properti somewher proo
f start appreci littl help nthank ad han answer make follow addit construct
given han answer clear bigcup bigcup cap emptyset neq left bigcup right left
bigcup right sum left right also construct subset monoton left right leq lef
t right final would sum leq sum result follow',)
```

```
------------------------------------------------------------------------
------------------------
('hql equival sql queri hql queri replac name class properti name error occu
r hql error',)
------------------------------------------------------------------------
------------------------
('undefin symbol architectur objc class skpsmtpmessag referenc error import
framework send email applic background import framework skpsmtpmessag somebo
di suggest get error collect ld return exit status import framework correct
sorc taken framework follow mfmailcomposeviewcontrol question lock field upd
at answer drag drop folder project click copi nthat',)
------------------------------------------------------------------------
------------------------
```

__ Saving Preprocessed data to a Database __

In [55]:

```
#Taking 0.5 Million entries to a dataframe.
write_db = 'Processed.db'
if os.path.isfile(write_db):
    conn_r = create_connection(write_db)
    if conn_r is not None:
        preprocessed_data = pd.read_sql_query("""SELECT question, Tags FROM QuestionsProces
conn_r.commit()
conn_r.close()
```

In [56]:

```
preprocessed_data.head()
```

Out[56]:

	question	tags
0	dynam datagrid bind silverlight bind datagrid ...	c# silverlight data-binding
1	dynam datagrid bind silverlight bind datagrid ...	c# silverlight data-binding columns
2	java lang noclassdeffounderror javax servlet j...	jsp jstl
3	java sql sqlexcept microsoft odbc driver manag...	java jdbc
4	better way updat feed fb php sdk novic faceboo...	facebook api facebook-php-sdk

In [57]:

```
print("number of data points in sample :", preprocessed_data.shape[0])
print("number of dimensions :", preprocessed_data.shape[1])
```

```
number of data points in sample : 499999
number of dimensions : 2
```

In [58]:

```
preprocessed_data = preprocessed_data.drop_duplicates(subset={'question'})
```

In [59]:

```
start = datetime.now()
#Splitting text in tags
preprocessed_data["tag_count"] = preprocessed_data["tags"].apply(lambda text: len(text.spli
# adding a new feature number of tags per question
print("Time taken to run this cell :", datetime.now() - start)
preprocessed_data.head()
```

```
Time taken to run this cell : 0:00:00.442458
```

Out[59]:

	question	tags	tag_count
0	dynam datagrid bind silverlight bind datagrid ...	c# silverlight data-binding	3
2	java lang noclassdeffounderror javax servlet j...	jsp jstl	2
3	java sql sqlexcept microsoft odbc driver manag...	java jdbc	2
4	better way updat feed fb php sdk novic faceboo...	facebook api facebook-php-sdk	3
5	btnadd click event open two window record ad o...	javascript asp.net web	3

In [60]:

```
#Removing question without any tags¶
preprocessed_data = preprocessed_data[preprocessed_data['tag_count']!=0]
```

In [61]:

```
#knowing length and converts the specified value into a string.
preprocessed_data['questionlen'] = preprocessed_data['question'].str.len()
```

In [62]:

```
#splitting the row
preprocessed_data['question_words'] = preprocessed_data['question'].apply(lambda row: len(r
```

In [63]:

```
#printing minimum length of questions in question-1
print ("Minimum length of the Text in Title : " , min(preprocessed_data['question_words']))
```

```
Minimum length of the Text in Title :  1
```

In [64]:

```
preprocessed_data
```

Out[64]:

	question	tags	tag_count	questionlen	question_words
0	dynam datagrid bind silverlight bind datagrid ...	c# silverlight data-binding	3	185	29
2	java lang noclassdeffounderror javax servlet j...	jsp jstl	2	311	45
3	java sql sqlexcept microsoft odbc driver manag...	java jdbc	2	105	16
4	better way updat feed fb php sdk novic faceboo...	facebook api facebook-php-sdk	3	163	30
5	btnadd click event open two window record ad o...	javascript asp.net web	3	195	33
6	sql inject issu prevent correct form submiss p...	php forms	2	515	90
7	countabl subaddit lebesgu measur let lbrace rb...	real-analysis measure-theory	2	500	81
8	hql equival sql queri hql queri replac name cl...	hibernate hql	2	85	15
9	undefin symbol architectur objc class skpsmtpm...	iphone email-integration	2	353	46
10	java lang nosuchmethoderror javax servlet serv...	java servlets jboss	3	892	121
11	obtain updat locat use gps servic app two butt...	android android-widget android-service	3	303	52
12	specifi initi vector iv match block size algor...	c# .net rijndaelmanaged cryptostream	4	154	24
13	uncaught typeerror properti addlistgroup objec...	javascript listbox	2	374	58
14	subqueri return row error new web program tri ...	sql subquery	2	546	100
15	feof file alway wrong start see lot post late ...	c feof	2	107	19
16	sum limit bigl bigl lfloor frac bigr rfloor bi...	number-theory functions inequality	3	338	59
17	frac work valu frac frac nwork valu frac	exponentiation	1	40	8
18	mathbb oplus mathbb isomorph mathbb mn mathbb ...	abstract-algebra modules	2	189	32

	question	tags	tag_count	questionlen	question_words
19	mathcal modulo normal closur mathbb time mathb...	group-theory	1	299	48
20	continu function show lim alpha int frac alpha...	calculus	1	370	64
21	cvcreatecontourtre dll opencv imgproc deprec g...	c# visual-c++ opencv emgucv emgu	5	270	38
22	fi fl result big like symbol use lyx xetex out...	fonts xetex lyx ligatures	4	155	29
23	selcurrentmanuf declar may inaccess due protec...	asp.net vb.net drop-down-menu	3	181	30
24	except prelud read pars haskel pars express re...	parsing haskell expression	3	246	42
25	append caus ie error follow code work fine ff ...	jquery append	2	235	41
26	html problem littl problem get function jqueri...	javascript jquery html jquery-ui jquery-plugins	5	305	52
27	tmp chang read sysadmin left day product box h...	linux filesystems	2	1083	211
28	invalid credenti oauth googl api throw apiserv...	php google google-api oauth-2.0 google-plus	5	199	29
29	intern server error asp net mvc work asp net m...	asp.net-mvc	1	134	24
30	thing divis zero c possibl duplic valu return ...	c++ c visual-c++	3	275	47
...	...	...	...	...	...
499969	chang sourc collect observablecollect properti...	vb.net data-binding observablecollection	3	435	67
499970	chang sourc dynam video js play flash make pla...	html5 flash video.js	3	335	59
499971	chang sourc file encod xcode develop primarili...	xcode ide xcode4	3	223	37
499972	chang sourc imag insid templat button silverli...	silverlight controltemplate	2	215	32
499973	chang sourc imag imag differ size resiz ie ni ...	javascript scripting dhtml	3	437	78
499974	chang sourc jqueri html video work ie issu cha...	jquery html5 html5-video	3	283	48
499975	chang space atom chemabov consid exampl lie le...	spacing fontspec chemfig	3	136	22
499976	chang space prxchang space need chang space te...	regex sas	2	152	26

	question	tags	tag_count	questionlen	question_words
499977	chang specif cell uitableview uitableview want...	objective-c xcode uitableview uitableviewcell	4	221	34
499978	chang specif color bitmap android suppos bitma...	android paint android-canvas	3	316	50
499979	chang specif row tabl void popul tableview wou...	ios uitableview	2	160	25
499980	chang speed durat anim stop look around find a...	javascript jquery performance duration	4	129	23
499981	chang speed usb devic use libusb usb test ni w...	linux libusb	2	98	18
499982	chang speed sound file look chang speed sound ...	algorithm audio multimedia	3	328	52
499983	chang speed ongo cakeyframeanim anim ad cakeyf...	objective-c ios core-animation	3	223	36
499984	chang speed ball drop process program ball dro...	java processing	2	192	33
499985	chang spinnermodel doubl valu use later outsid...	java eclipse swing gui jform-designer	5	281	42
499986	chang sprite difficulti depend level game make...	ios xcode cocos2d ccsprite arc4random	5	590	99
499987	chang sprite textur ccspritebatchnod tri chang...	objective-c cocos2d-iphone	2	443	68
499988	chang sql queri output depend column valu two ...	sql	1	398	66
499989	chang squar rectangl simpl swing program issu ...	java swing awt draw shapes	5	311	49
499990	chang src attribut html audio element tri use ...	javascript audio html5	3	255	45
499991	chang src attribut imag tri avoid creat duplic...	jquery dom slimbox	3	439	73
499992	chang src imag dynam jqueri im tri work imag c...	jquery image	2	420	79
499993	chang src img tag make work tri chang imag web...	javascript jquery html json	4	296	53
499994	chang src valu jqueri jqueri variabl like want...	javascript jquery tags replace	4	142	25
499995	chang stage size base orient devic creat appli...	android actionscript-3 flash	3	241	39
499996	chang stage empublit project eclips use book e...	android commonsware	2	347	57

	question	tags	tag_count	questionlen	question_words
499997	chang standard error color geom smooth plot da...	r ggplot2	2	291	49
499998	chang standard output present page size size w...	beamer paper-size	2	199	31

493322 rows × 5 columns

In [65]:

```
preprocessed_data.describe()
print(preprocessed_data.(include='all'))
print("*"*60)
print(preprocessed_data.info())
```

```
************************************************************
<class 'pandas.core.frame.DataFrame'>
Int64Index: 493322 entries, 0 to 499998
Data columns (total 5 columns):
question          493322 non-null object
tags              493322 non-null object
tag_count         493322 non-null int64
questionlen       493322 non-null int64
question_words    493322 non-null int64
dtypes: int64(3), object(2)
memory usage: 22.6+ MB
None
```

Here as you can notice mean value is less than median value of each column which is represented by 50% (50th percentile) in index column.

There is notably a large difference between 75th %tile and max values of predictors “residual sugar”,”free sulfur dioxide”,”total sulfur dioxide”.

Thus observations 1 and 2 suggests that there are extreme values-Outliers in our data set.

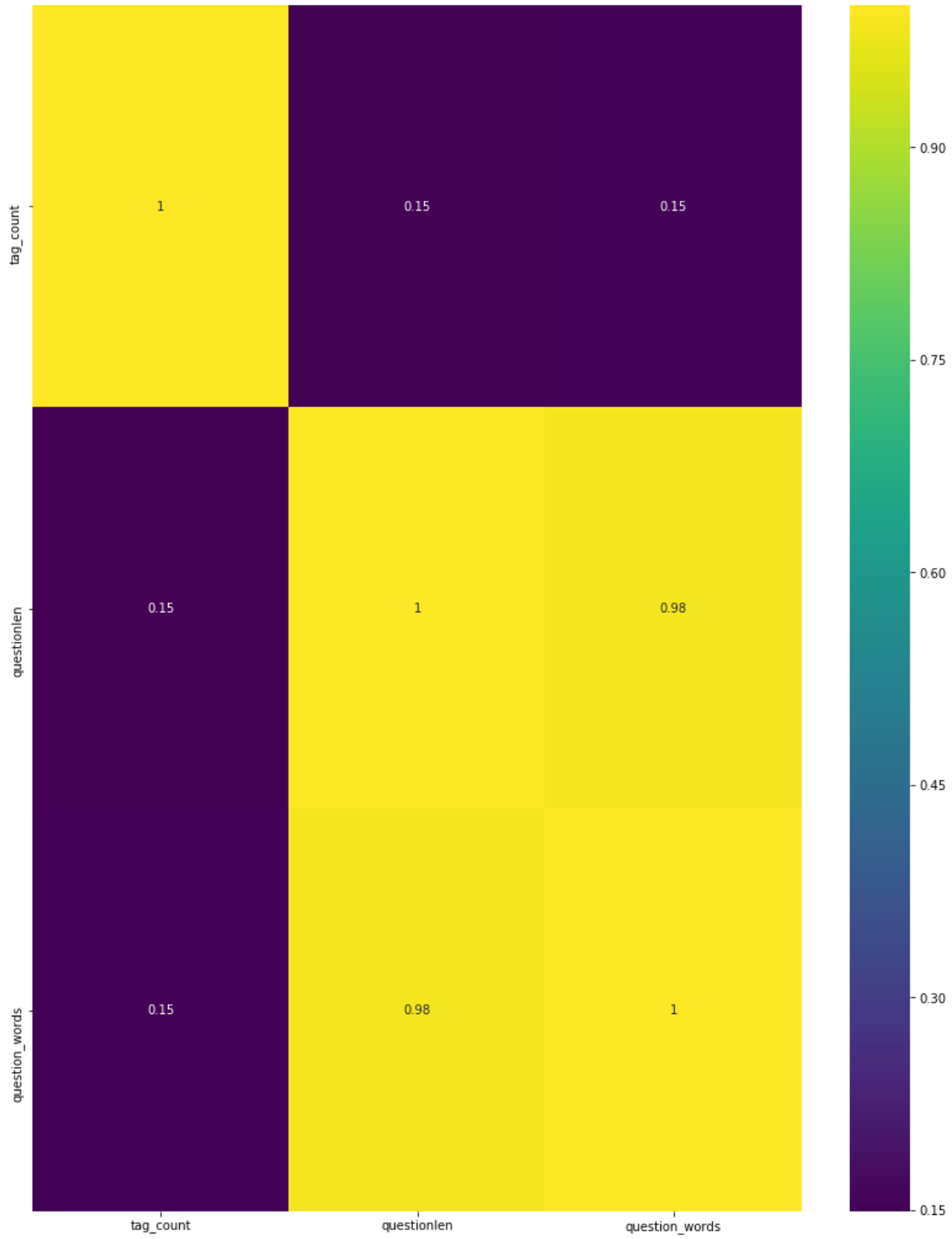
checking correlation

In [66]:

```
f, ax = plt.subplots(figsize=(14,18))
corr = preprocessed_data.corr()
sns.heatmap(corr,xticklabels=corr.columns.values,yticklabels=corr.columns.values,annot=True
```

Out[66]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f70d612acc0>
```



correlation value lies between -1 to +1. Highly correlated variables will have correlation value close to +1 and less correlated variables will have correlation value close to -1.The diagonal elements of the matrix value are always 1

5. Machine Learning Models

5.1 Converting tags for multilabel problems

X	y1	y2	y3	y4
x1	0	1	1	0
x1	1	0	0	0
x1	0	1	0	0

- Converting string Tags to multilable output variables __

In [67]:

```
vectorizer = CountVectorizer(tokenizer = lambda x: x.split(), binary='true')
multilabel_y = vectorizer.fit_transform(preprocessed_data['tags'])
```

- We will sample the number of tags instead considering all of them (due to limitation of computing power) __

In [68]:

```
def tags_to_choose(n):
    t = multilabel_y.sum(axis=0).tolist()[0]
    sorted_tags_i = sorted(range(len(t)), key=lambda i: t[i], reverse=True)
    multilabel_yn=multilabel_y[:,sorted_tags_i[:n]]
    return multilabel_yn

def questions_explained_fn(n):
    multilabel_yn = tags_to_choose(n)
    x= multilabel_yn.sum(axis=1)
    return (np.count_nonzero(x==0))
```

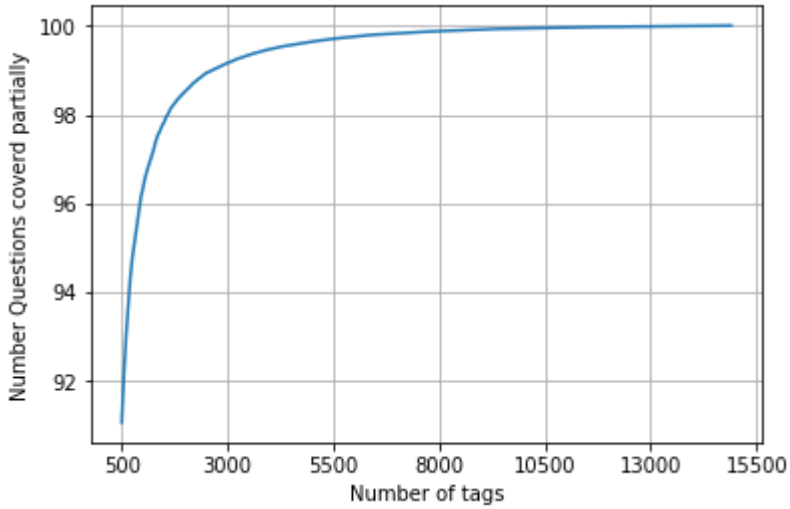
__ Selecting 500 Tags __

In [69]:

```
questions_explained = []
total_tags=multilabel_y.shape[1]
total_qs=preprocessed_data.shape[0]
for i in range(500, total_tags, 100):
    questions_explained.append(np.round(((total_qs-questions_explained_fn(i))/total_qs)*100
```

In [70]:

```
fig, ax = plt.subplots()
ax.plot(questions_explained)
xlabel = list(500+np.array(range(-50,450,50))*50)
ax.set_xticklabels(xlabel)
plt.xlabel("Number of tags")
plt.ylabel("Number Questions coverd partially")
plt.grid()
plt.show()
# you can choose any number of tags based on your computing power, minimun is 500(it covers
print("with ",5500,"tags we are covering ",questions_explained[50],"% of questions")
print("with ",500,"tags we are covering ",questions_explained[0],"% of questions")
```




```
with  5500 tags we are covering  99.161 % of questions
with  500 tags we are covering  91.07 % of questions
```

In [71]:

```
# we will be taking 500 tags
multilabel_yx = tags_to_choose(500)
print("number of questions that are not covered :", questions_explained_fn(500),"out of ",
```

```
number of questions that are not covered : 44052 out of  493322
```

In [72]:

```
print("Number of tags in sample :", multilabel_y.shape[1])
print("number of tags taken :", multilabel_yx.shape[1],"(",(multilabel_yx.shape[1]/multilab
```

```
Number of tags in sample : 29372
number of tags taken : 500 ( 1.7023015116437425 %)
```

__ We consider top 15% tags which covers 99% of the questions __

5.2 Split the data into test and train (80:20)

In [73]:

```
total_size=preprocessed_data.shape[0]
train_size=int(0.80*total_size)

x_train=preprocessed_data.head(train_size)
x_test=preprocessed_data.tail(total_size - train_size)

y_train = multilabel_yx[0:train_size,:]
y_test = multilabel_yx[train_size:preprocessed_data.shape[0],:]
```

In [74]:

```
print("Number of data points in train data :", y_train.shape)
print("Number of data points in test data :", y_test.shape)
```

```
Number of data points in train data : (394657, 500)
Number of data points in test data : (98665, 500)
```

5.2.1 Featurizing data with BOW vectorizer upto 4 grams and compute the micro f1 score with Logistic regression(OvR)

In [75]:

```
start = datetime.now()
vectorizer = CountVectorizer(min_df=0.00009, max_features=200000,tokenizer = lambda x: x.sp
x_train_multilabel = vectorizer.fit_transform(x_train['question'])
x_test_multilabel = vectorizer.transform(x_test['question'])
print("Time taken to run this cell :", datetime.now() - start)
```

```
Time taken to run this cell : 0:08:01.998748
```

In [76]:

```
print("Dimensions of train data X:",x_train_multilabel.shape, "Y :",y_train.shape)
print("Dimensions of test data X:",x_test_multilabel.shape,"Y:",y_test.shape)
```

```
Dimensions of train data X: (394657, 96397) Y : (394657, 500)
Dimensions of test data X: (98665, 96397) Y: (98665, 500)
```

In [ ]:

```
24
```

5.2.3 Applying Logistic Regression with OneVsRest Classifier

In [77]:

```
import pickle
start = datetime.now()
classifier = OneVsRestClassifier(SGDClassifier(loss='log', alpha=0.00001, penalty='l1'))
classifier.fit(x_train_multilabel, y_train)

# save the model to disk
filename = 'Applying Logistic Regression with OneVsRest Classifier.sav'
pickle.dump(classifier, open(filename, 'wb'))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Time taken to run this cell : 0:25:55.275728
```

In [78]:

```
# save the model to disk
filename = 'Applying Logistic Regression with OneVsRest Classifier.sav'
pickle.dump(classifier, open(filename, 'wb'))
print("Time taken to run this cell :", datetime.now() - start)

# some time later...

# load the model from disk
loaded_model = pickle.load(open(filename, 'rb'))
```

```
Time taken to run this cell : 0:25:56.689720
```

In [79]:

```
predictions = loaded_model.predict (x_test_multilabel)


print("Accuracy :",metrics.accuracy_score(y_test, predictions))
print("Hamming loss ",metrics.hamming_loss(y_test,predictions))


precision = precision_score(y_test, predictions, average='micro')
recall = recall_score(y_test, predictions, average='micro')
f1 = f1_score(y_test, predictions, average='micro')

print("Micro-average quality numbers")
print("Precision: {:.4f}, Recall: {:.4f}, F1-measure: {:.4f}".format(precision, recall, f1))

precision = precision_score(y_test, predictions, average='macro')
recall = recall_score(y_test, predictions, average='macro')
f1 = f1_score(y_test, predictions, average='macro')

print("Macro-average quality numbers")
print("Precision: {:.4f}, Recall: {:.4f}, F1-measure: {:.4f}".format(precision, recall, f1))

print (metrics.classification_report(y_test, predictions))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Accuracy : 0.10497136775959054
Hamming loss  0.005345583540262504
Micro-average quality numbers
Precision: 0.3159, Recall: 0.4573, F1-measure: 0.3737
Macro-average quality numbers
Precision: 0.2261, Recall: 0.3905, F1-measure: 0.2790
              precision    recall  f1-score   support

           0       0.75      0.83      0.79      5497
           1       0.43      0.45      0.44      8157
           2       0.50      0.52      0.51      6471
           3       0.45      0.50      0.47      3203
           4       0.56      0.49      0.52      6398
           5       0.46      0.51      0.48      2860
           6       0.61      0.58      0.59      5018
           7       0.64      0.65      0.64      4504
           8       0.24      0.22      0.23      2985
           9       0.56      0.66      0.61      2750
          10       0.32      0.29      0.30      3036
```

5.2.4 Applying Logistic Regression with OneVsRest Classifier Hyper Parameter Tunning

In [80]:

```
# applying grid search to find best c
from sklearn.model_selection import GridSearchCV
start = datetime.now()
tuned_parameters = [{'estimator__alpha': [0.000001, 0.00001, 0.0001, 0.001, 0.01]}]

model = GridSearchCV(OneVsRestClassifier(SGDClassifier(loss='log', penalty='l1')), tuned_pa
model.fit(x_train_multilabel, y_train)

# save the model to disk
filename = 'Applying Logistic Regression with OneVsRest Classifier Hyper Parameter Tunning.
pickle.dump(model, open(filename, 'wb'))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Time taken to run this cell : 2:33:33.574932
```

In [81]:

```
# some time later...

# load the model from disk
loaded_model = pickle.load(open(filename, 'rb'))

print(loaded_model.best_estimator_)
a = loaded_model.best_params_
optimal_alpha = a.get('estimator__alpha')
print(optimal_alpha)
```

```
OneVsRestClassifier(estimator=SGDClassifier(alpha=0.0001, average=False, cla
ss_weight=None,
       early_stopping=False, epsilon=0.1, eta0=0.0, fit_intercept=True,
       l1_ratio=0.15, learning_rate='optimal', loss='log', max_iter=None,
       n_iter=None, n_iter_no_change=5, n_jobs=None, penalty='l1',
       power_t=0.5, random_state=None, shuffle=True, tol=None,
       validation_fraction=0.1, verbose=0, warm_start=False),
          n_jobs=None)
0.0001
```

In [82]:

```
results = loaded_model.cv_results_
results['mean_test_score']
```

Out[82]:

```
array([0.38103605, 0.36787449, 0.42033243, 0.36129053, 0.151717  ])
```

```
C=0.000001, 0.00001, 0.0001, 0.001, 0.01
plt.plot(C,results['mean_test_score'],marker='o')
plt.xlabel('alpha')
plt.ylabel('f1score')
plt.title("F1score vs hyperparameter alpha")
plt.grid()
plt.show()
```

In [84]:

```
start = datetime.now()
classifier = OneVsRestClassifier(SGDClassifier(loss='hinge', alpha=optimal_alpha, penalty='
classifier.fit(x_train_multilabel, y_train)
# save the model to disk
filename = 'Applying Logistic Regression with OneVsRest Classifier Hyper Parameter Tunning
pickle.dump(classifier, open(filename, 'wb'))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Time taken to run this cell : 0:18:54.884941
```

In [85]:

```
# load the model from disk
loaded_model = pickle.load(open(filename, 'rb'))
```

In [86]:

```
predictions = loaded_model.predict (x_test_multilabel)


print("Accuracy :",metrics.accuracy_score(y_test, predictions))
print("Hamming loss ",metrics.hamming_loss(y_test,predictions))


precision = precision_score(y_test, predictions, average='micro')
recall = recall_score(y_test, predictions, average='micro')
f1 = f1_score(y_test, predictions, average='micro')

print("Micro-average quality numbers")
print("Precision: {:.4f}, Recall: {:.4f}, F1-measure: {:.4f}".format(precision, recall, f1))

precision = precision_score(y_test, predictions, average='macro')
recall = recall_score(y_test, predictions, average='macro')
f1 = f1_score(y_test, predictions, average='macro')

print("Macro-average quality numbers")
print("Precision: {:.4f}, Recall: {:.4f}, F1-measure: {:.4f}".format(precision, recall, f1))

print (metrics.classification_report(y_test, predictions))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Accuracy : 0.13149546445041302
Hamming loss  0.004047149445091978
Micro-average quality numbers
Precision: 0.4185, Recall: 0.4120, F1-measure: 0.4152
Macro-average quality numbers
Precision: 0.3159, Recall: 0.3437, F1-measure: 0.3138
              precision    recall  f1-score   support

           0       0.71      0.79      0.75      5497
           1       0.43      0.39      0.41      8157
           2       0.60      0.42      0.50      6471
           3       0.46      0.45      0.45      3203
           4       0.61      0.47      0.53      6398
           5       0.51      0.50      0.50      2860
           6       0.66      0.55      0.60      5018
           7       0.69      0.62      0.65      4504
           8       0.29      0.18      0.23      2985
           9       0.58      0.65      0.61      2750
          10       0.36      0.27      0.31      3036
```

5.2.5 Applying Linear SVM with OneVsRest Classifier

In [87]:

```
start = datetime.now()
classifier = OneVsRestClassifier(SGDClassifier(loss='hinge', alpha=0.00001, penalty='l1'))
classifier.fit(x_train_multilabel, y_train)
# save the model to disk
filename = 'Applying Linear SVM with OneVsRest Classifier.sav'
pickle.dump(classifier, open(filename, 'wb'))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Time taken to run this cell : 0:22:58.137894
```

In [88]:

```
# load the model from disk
loaded_model = pickle.load(open(filename, 'rb'))
```

In [89]:

```
predictions = loaded_model.predict (x_test_multilabel)


print("Accuracy :",metrics.accuracy_score(y_test, predictions))
print("Hamming loss ",metrics.hamming_loss(y_test,predictions))


precision = precision_score(y_test, predictions, average='micro')
recall = recall_score(y_test, predictions, average='micro')
f1 = f1_score(y_test, predictions, average='micro')

print("Micro-average quality numbers")
print("Precision: {:.4f}, Recall: {:.4f}, F1-measure: {:.4f}".format(precision, recall, f1))

precision = precision_score(y_test, predictions, average='macro')
recall = recall_score(y_test, predictions, average='macro')
f1 = f1_score(y_test, predictions, average='macro')

print("Macro-average quality numbers")
print("Precision: {:.4f}, Recall: {:.4f}, F1-measure: {:.4f}".format(precision, recall, f1))

print (metrics.classification_report(y_test, predictions))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Accuracy : 0.10553894491461005
Hamming loss  0.005279704049054883
Micro-average quality numbers
Precision: 0.3192, Recall: 0.4538, F1-measure: 0.3748
Macro-average quality numbers
Precision: 0.2265, Recall: 0.3871, F1-measure: 0.2785
              precision    recall  f1-score   support

           0       0.75      0.82      0.78      5497
           1       0.42      0.47      0.44      8157
           2       0.53      0.48      0.50      6471
           3       0.46      0.50      0.48      3203
           4       0.56      0.49      0.52      6398
           5       0.45      0.54      0.49      2860
           6       0.58      0.59      0.58      5018
           7       0.63      0.64      0.64      4504
           8       0.24      0.24      0.24      2985
           9       0.55      0.68      0.60      2750
          10       0.34      0.32      0.33      3036
```

5.2.6 Applying Linear SVM with OneVsRest Classifier Hyper Parameter Tunning

In [90]:

```
# applying grid search to find best c
from sklearn.model_selection import GridSearchCV
start = datetime.now()
tuned_parameters = [{'estimator__alpha': [0.000001, 0.00001, 0.0001, 0.001, 0.01]}]

model = GridSearchCV(OneVsRestClassifier(SGDClassifier(loss='hinge', penalty='l1')), tuned_
model.fit(x_train_multilabel, y_train)

# save the model to disk
filename = 'Applying Linear SVM with OneVsRest Classifier Hyper Parameter Tunning.sav'
pickle.dump(model, open(filename, 'wb'))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Time taken to run this cell : 2:06:59.542693
```

In [91]:

```
# some time later...

# load the model from disk
loaded_model = pickle.load(open(filename, 'rb'))

print(loaded_model.best_estimator_)
optimal_alpha = a.get('estimator__alpha')
print(optimal_alpha)
```

```
OneVsRestClassifier(estimator=SGDClassifier(alpha=0.0001, average=False, cla
ss_weight=None,
       early_stopping=False, epsilon=0.1, eta0=0.0, fit_intercept=True,
       l1_ratio=0.15, learning_rate='optimal', loss='hinge', max_iter=None,
       n_iter=None, n_iter_no_change=5, n_jobs=None, penalty='l1',
       power_t=0.5, random_state=None, shuffle=True, tol=None,
       validation_fraction=0.1, verbose=0, warm_start=False),
          n_jobs=None)
0.0001
```

In [92]:

```
results = loaded_model.cv_results_
results['mean_test_score']
```

Out[92]:

```
array([0.38069522, 0.37152447, 0.41355929, 0.37360015, 0.14186956])
```

C=0.000001, 0.00001, 0.0001, 0.001, 0.01 plt.plot(C,results['mean_test_score'],marker='o') plt.xlabel('alpha') plt.ylabel('f1score') plt.title("F1score vs hyperparameter alpha") plt.grid() plt.show()

In [93]:

```
start = datetime.now()
classifier = OneVsRestClassifier(SGDClassifier(loss='hinge', alpha=optimal_alpha, penalty='
classifier.fit(x_train_multilabel, y_train)
# save the model to disk
filename = 'Applying Linear SVM with OneVsRest Classifier Hyper Parameter Tunning with alph
pickle.dump(classifier, open(filename, 'wb'))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Time taken to run this cell : 0:18:45.884876
```

In [94]:

```
# load the model from disk
loaded_model = pickle.load(open(filename, 'rb'))
```

In [95]:

```
predictions = loaded_model.predict (x_test_multilabel)


print("Accuracy :",metrics.accuracy_score(y_test, predictions))
print("Hamming loss ",metrics.hamming_loss(y_test,predictions))


precision = precision_score(y_test, predictions, average='micro')
recall = recall_score(y_test, predictions, average='micro')
f1 = f1_score(y_test, predictions, average='micro')

print("Micro-average quality numbers")
print("Precision: {:.4f}, Recall: {:.4f}, F1-measure: {:.4f}".format(precision, recall, f1)

precision = precision_score(y_test, predictions, average='macro')
recall = recall_score(y_test, predictions, average='macro')
f1 = f1_score(y_test, predictions, average='macro')

print("Macro-average quality numbers")
print("Precision: {:.4f}, Recall: {:.4f}, F1-measure: {:.4f}".format(precision, recall, f1)

print (metrics.classification_report(y_test, predictions))
print("Time taken to run this cell :", datetime.now() - start)## Conclusion
```

```
Accuracy : 0.12932650889373132
Hamming loss  0.00407339988851163
Micro-average quality numbers
Precision: 0.4149, Recall: 0.4096, F1-measure: 0.4122
Macro-average quality numbers
Precision: 0.3146, Recall: 0.3438, F1-measure: 0.3131
              precision    recall  f1-score   support

           0       0.73      0.79      0.76      5497
           1       0.44      0.36      0.40      8157
           2       0.58      0.44      0.50      6471
           3       0.48      0.46      0.47      3203
           4       0.63      0.44      0.52      6398
           5       0.55      0.49      0.52      2860
           6       0.66      0.56      0.60      5018
           7       0.68      0.61      0.64      4504
           8       0.28      0.17      0.21      2985
           9       0.54      0.68      0.60      2750
          10       0.40      0.18      0.25      3036
```

6.Conclusion

In [101]:

```
from prettytable import PrettyTable

x = PrettyTable()

x.field_names = ["Classification model","Regularization","Hyperparameter", "Accuracy","prec

x.add_row(["Logistic Regression", "L1",0.00001,0.1049,0.3159,0.4573, 0.3737,0.0054])
x.add_row(["Logistic Regression with Hyperparameter", "L1",0.001,0.1314,0.4185,0.4120,0.415
x.add_row(["Linear SVM", "L1",0.00001,0.1055,0.3192,0.4538,0.3748,0.0052])
x.add_row(["Linear SVM with Hyperparameter", "L1", 0.001,0.1293,0.4149,0.4096,0.4122,0.0040
print(x)
```

```
+----------------------------------------+----------------+----------------
+----------+-----------+--------+----------+-------------+
|           Classification model          | Regularization | Hyperparameter
| Accuracy | precision | Recall | F1 micro | Hammingloss |
+----------------------------------------+----------------+----------------
+----------+-----------+--------+----------+-------------+
|           Logistic Regression           |       L1       |     1e-05
|  0.1049  |   0.3159  | 0.4573 |  0.3737  |    0.0054   |
| Logistic Regression with Hyperparameter |       L1       |     0.001
|  0.1314  |   0.4185  | 0.412  |  0.4152  |    0.004    |
|               Linear SVM                |       L1       |     1e-05
|  0.1055  |   0.3192  | 0.4538 |  0.3748  |    0.0052   |
|      Linear SVM with Hyperparameter     |       L1       |     0.001
|  0.1293  |   0.4149  | 0.4096 |  0.4122  |    0.004    |
+----------------------------------------+----------------+----------------
+----------+-----------+--------+----------+-------------+
```

we had nearly 50k tags don't require all tags to build our machine learning model.Hence we will use top 500 tags would cover 99% of questions which is

sufficient for train model and get reasonably good micro F1_score

<h3> Steps Involved:- </h3>:-

1) Connecting SQL file

2) Reading Data

3) performed Feature extraction

4) Preprocessing of Tags(only 0.5 million questions ae considered and we select top 500 tags due to less computational power)

5) Exploratory Data Analysis

6) Spliting data into train and test based on time (80:20)

7) Distribution of y_i's in Train, Test

8) Applying Machine learning Algorithms Logistice Regression and Linear SVM

9) Hyperparameter Tunning Model

10) calculating Accuracy,Precision Score,Recall Score,Classification Report

11) Conclusion

Here i skipped MSE vs alpha graph because its taking lot of time i had tried and waited 7hours so i skipped here

In [ ]: