StackOverflow Tag Prediction

Project Overview

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 Statement

This is a supervised learning problem where we need to suggest the tags based on the content of the question posted on Stackoverflow.The goal is to predict as many tags as possible with high precision and recall.We will train our model on a dataset containing million of questions presented as unstructured text.

To solve this problem we will perform the following tasks: * Exploratory Data Analysis * Preprocess the data. * Train and tune the hyperparameters of the Logistic Regression. * Test the precision and recall of the model on the testing set.

Problem Type

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.

In [14]:
```
import sqlite3
import pandas as pd
from sqlalchemy import create_engine
from datetime import datetime
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import CountVectorizer,TfidfVectorizer
from wordcloud import WordCloud
import numpy as np
from nltk.corpus import stopwords
import nltk,re,csv,os,pickle
from prettytable import PrettyTable
from nltk.tokenize import word_tokenize
from nltk.stem.snowball import SnowballStemmer
from sklearn.linear_model import LogisticRegression,SGDClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.multiclass import OneVsRestClassifier
from sklearn.metrics import accuracy_score,f1_score,recall_score,classificatio
n_report,precision_score,hamming_loss
import warnings
warnings.filterwarnings("ignore")
```

Exploratory Data Analysis

Before building our model we will start by exploring our dataset. You can download the train and test zip file from [here (https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction/data)](https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction/data) Let's start by loading the data from SQL lite

In [2]:
```
#creating a db file from csv
if not os.path.isfile("train.db"):
    disk_engine = create_engine('sqlite:///train.db')
    chunksize = 180000
    j=0
    index_start=1
    for df in pd.read_csv('Train.csv', names=['Id', 'Title', 'Body', 'Tags'],
chunksize=chunksize, iterator=True, encoding='utf-8', ):
        df.index += index_start
        j+=1
        df.to_sql('data', disk_engine, if_exists='append')
        index_start = df.index[-1] + 1
```

How many questions are present in this dataset?

In [4]:
```
if os.path.isfile('train.db'):
    con = sqlite3.connect('train.db')
    num_rows = pd.read_sql_query("""SELECT count(*) FROM data""", con)
    print("Dataset contains",num_rows['count(*)'].values[0],"rows")
    con.close()
else:
    print("Please download the train.db file from drive or run the above cell
 to genarate train.db file")
```

```
Dataset contains 6034196 rows
```

Checking for duplicates

In [5]:
```
if os.path.isfile('train.db'):
    con = sqlite3.connect('train.db')
    df_no_dup = pd.read_sql_query('SELECT Title, Body, Tags, COUNT(*) as cnt_d
up FROM data GROUP BY Title, Body, Tags', con)
    con.close()
else:
    print("File not found")
```

In [6]:
```
df_no_dup.head()
```

Out[6]:

	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?	<p>I should do binding for datagrid dynamicall...	c# silverlight data-binding	1
2	Dynamic Datagrid Binding in Silverlight?	<p>I should do binding for datagrid dynamicall...	c# silverlight data-binding columns	1
3	java.lang.NoClassDefFoundError: javax/serv...	<p>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

From the cnt_dup column we can observe that 4th question appeared 2 times in the dataset

In [7]:
```
print("number of duplicate questions :", num_rows['count(*)'].values[0]- df_no
_dup.shape[0], "(",(1-((df_no_dup.shape[0])/(num_rows['count(*)'].values[0])))
*100,"% )")
```

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

In [8]:
```
# number of times each question appeared in our database
df_no_dup.cnt_dup.value_counts()
```

Out[8]:
```
1    2656284
2    1272336
3     277575
4         90
5         25
6          5
Name: cnt_dup, dtype: int64
```

In [9]:
```
df_no_dup.fillna('',inplace=True)
df_no_dup["tag_count"] = df_no_dup["Tags"].apply(lambda text: len(text.split("
 ")))
# adding a new feature number of tags per question
df_no_dup.head()
```

Out[9]:

	Title	Body	Tags	cnt_dup	ta
0	Implementing Boundary Value Analysis of S...	<pre><code>#include<iostream>\n#include&...	c++ c	1	
1	Dynamic Datagrid Binding in Silverlight?	<p>I should do binding for datagrid dynamicall...	c# silverlight data-binding	1	
2	Dynamic Datagrid Binding in Silverlight?	<p>I should do binding for datagrid dynamicall...	c# silverlight data-binding columns	1	
3	java.lang.NoClassDefFoundError: javax/serv...	<p>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 [10]:
```
df_no_dup.tag_count.value_counts()
```

Out[10]:
```
3    1206157
2    1111706
4     814996
1     568298
5     505158
Name: tag_count, dtype: int64
```

In [11]:
```
if not os.path.isfile("train_no_dup.db"):
    disk_eng=create_engine("sqlite:///train_no_dup.db")
    no_dup=pd.DataFrame(df_no_dup,columns=['Title','Body','Tags'])
    no_dup.to_sql("no_dup_train",disk_eng)
```

In [12]:
```
#This method seems more appropriate to work with this much data.
#creating the connection with database file.
if os.path.isfile('train_no_dup.db'):
    con = sqlite3.connect('train_no_dup.db')
    tag_data = pd.read_sql_query("""SELECT Tags FROM no_dup_train""", con)
    con.close()

    tag_data.drop(tag_data.index[0], inplace=True)
else:
    print("No file found with name in the given directory")
tag_data.head()
```

Out[12]:

	Tags
1	c# silverlight data-binding
2	c# silverlight data-binding columns
3	jsp jstl
4	java jdbc
5	facebook api facebook-php-sdk

Total number of unique tags

In [14]:
```
vectorizer=CountVectorizer(tokenizer= lambda x:x.split())
tag_dtm=vectorizer.fit_transform(tag_data["Tags"])
```

In [30]:
```
print("Number of data points :", tag_dtm.shape[0])
print("Number of unique tags :", tag_dtm.shape[1])
```

```
Number of data points : 4206314
Number of unique tags : 42048
```

In [64]:
```
#'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', '.bash
-profile', '.class-file', '.cs-file', '.doc', '.drv', '.ds-store']
```

Number of times a tag appeared

In [65]:
```
freqs = tag_dtm.sum(axis=0).A1
result = dict(zip(tags, freqs))
```

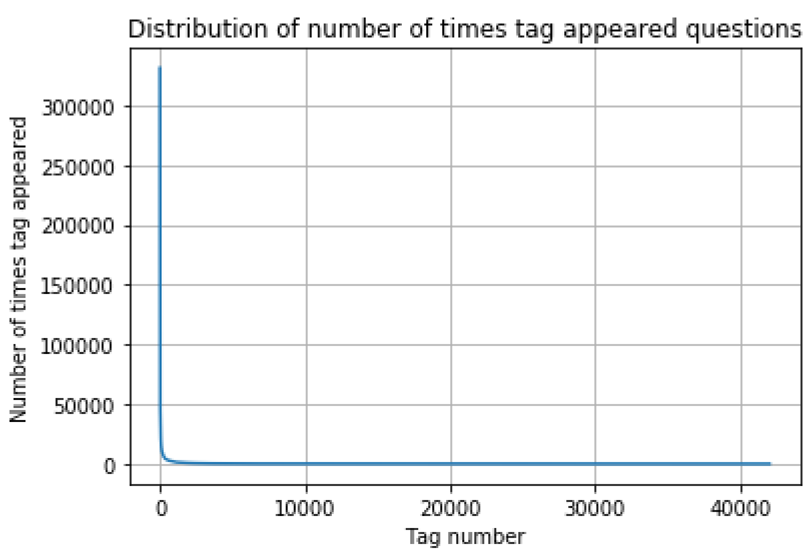
In [66]:
```
#Saving this dictionary to csv files.
if not os.path.isfile('tag_counts_dict_dtm.csv'):
    with open('tag_counts_dict_dtm.csv', 'w') as csv_file:
        writer = csv.writer(csv_file)
        for key, value in result.items():
            writer.writerow([key, value])
tag_df = pd.read_csv("tag_counts_dict_dtm.csv", names=['Tags', 'Counts'])
tag_df.head()
```

Out[66]:

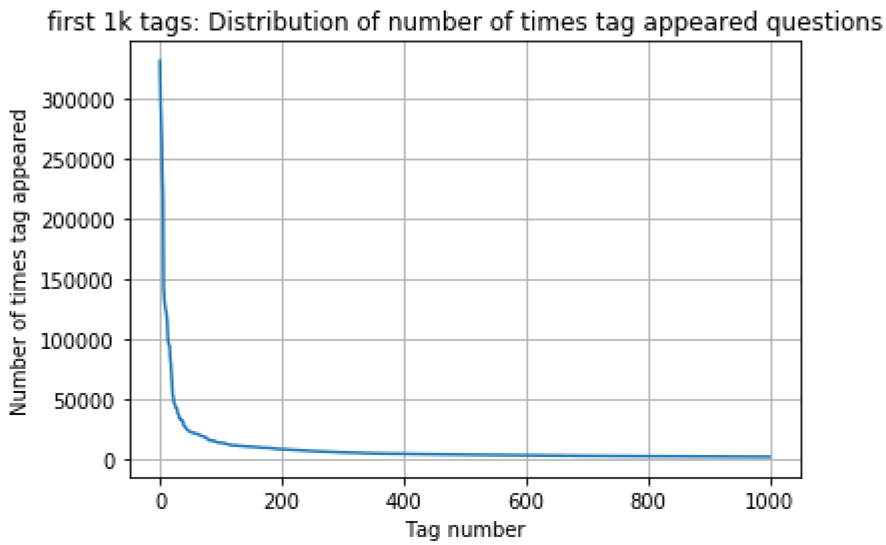
	Tags	Counts
0	groupwise-maximum	13
1	pisa-pdf	3
2	trn	1
3	filtering	1081
4	stringify	77

In [8]:
```
tag_df_sorted = tag_df.sort_values(['Counts'], ascending=False)
tag_counts = tag_df_sorted['Counts'].values
```

In [44]:
```
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()
```

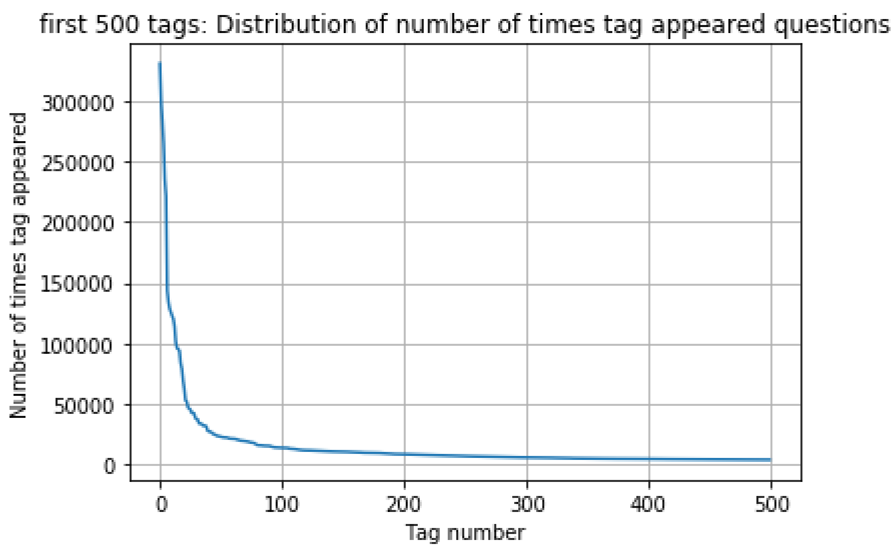


In [46]:
```
plt.plot(tag_counts[0:1000])
plt.title('first 1k tags: Distribution of number of times tag appeared questio
ns')
plt.grid()
plt.xlabel("Tag number")
plt.ylabel("Number of times tag appeared")
plt.show()
print(len(tag_counts[0:1000:5]), tag_counts[0:1000:5])
```



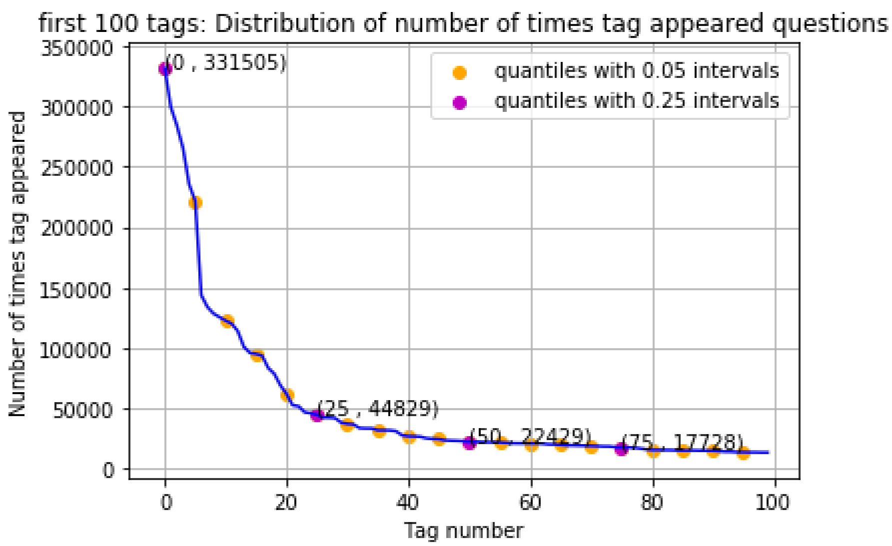

```
200 [331505 221533 122769  95160  62023  44829  37170  31897  26925  24537
  22429  21820  20957  19758  18905  17728  15533  15097  14884  13703
  13364  13157  12407  11658  11228  11162  10863  10600  10350  10224
  10029   9884   9719   9411   9252   9148   9040   8617   8361   8163
   8054   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 [47]:
```
plt.plot(tag_counts[0:500])
plt.title('first 500 tags: Distribution of number of times tag appeared questi
ons')
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 [10]:
```
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 with 0.05 intervals")
# quantiles with 0.25 difference
plt.scatter(x=list(range(0,100,25)), y=tag_counts[0:100:25], c='m', label = "q
uantiles with 0.25 intervals")

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 questi
ons')
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 [11]:
```
# 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.

Tags Per Question

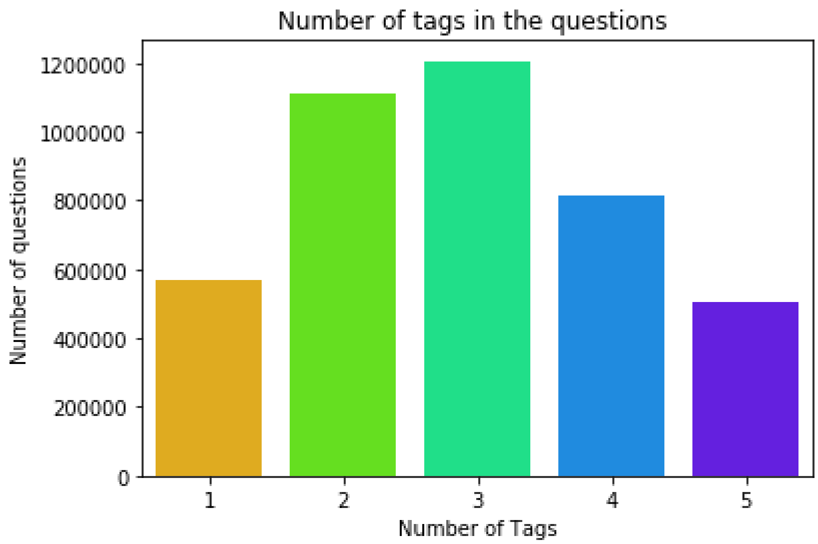
In [42]:
```
#Storing the count of tag in each question in list 'tag_count'
tag_quest_count = tag_dtm.sum(axis=1).tolist()
#Converting each value in the 'tag_quest_count' to integer.
tag_quest_count=[int(j) for i in tag_quest_count for j in i]
print ('We have total {} datapoints.'.format(len(tag_quest_count)))

print(tag_quest_count[:5])
```

```
We have total 4206314 datapoints.
[3, 4, 2, 2, 3]
```

In [60]:
```
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_count)))
```

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

In [61]:
```
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()
```

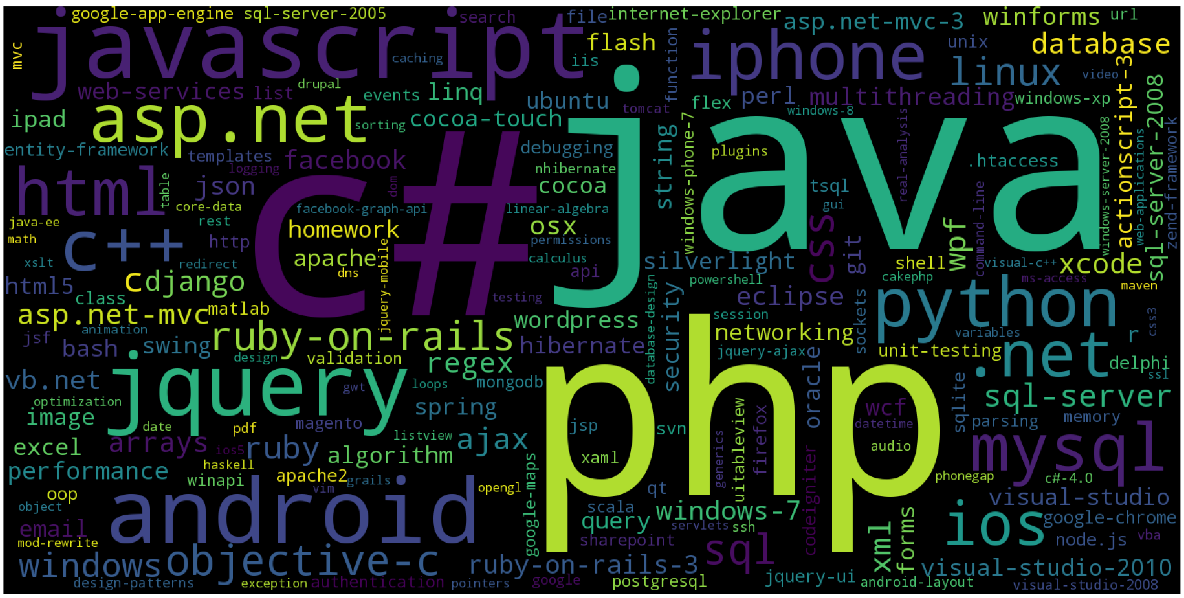


Observations:

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

Most frequent Tags:

In [73]:
```
# Ploting word cloud
start = datetime.now()

# converting 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)
```

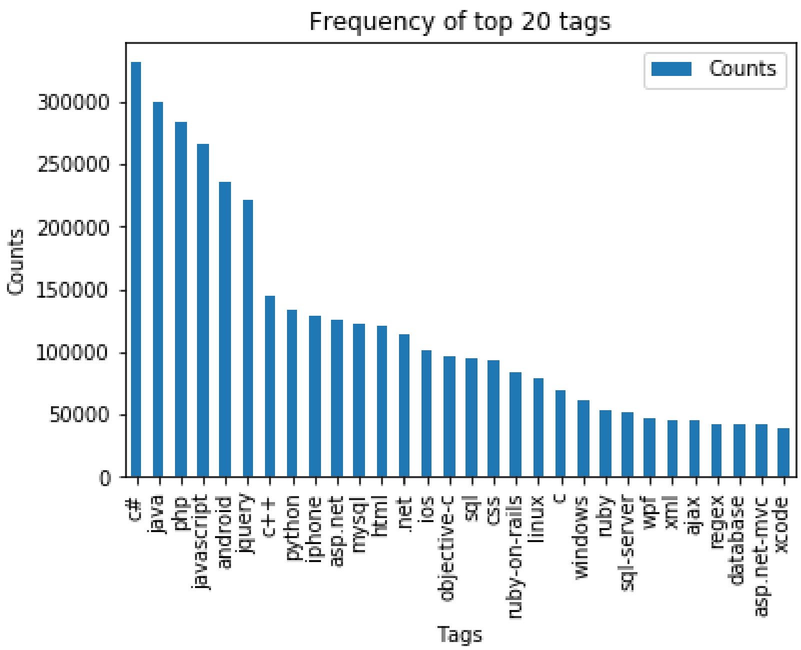


Time taken to run this cell : 0:00:04.572191

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

The top 20 tags

In [76]:
```
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.

Cleaning and preprocessing of Questions

Preprocessing

1. Sample 0.5M data points
2. Separate out code-snippets from Body
3. Give more weightage to title : Add title three times to the question
4. Remove Spcial characters from Question title and description (not in code)
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 [2]:
```
def stripHtml(data):
    cleanr=re.compile('<.*?>')
    cleantext=re.sub(cleanr,' ',str(data))
    return cleantext
stop_words=set(stopwords.words('english'))
stemmer=SnowballStemmer('english')
```

In [4]:
```
def createDBConnection(db_file):
    try:
        con=sqlite3.connect(db_file)
        return con
    except Error as e:
        print(e)
    return None

def createDBTable(con,create_sql_table):
    try:
        c=con.cursor()
        c.execute(create_sql_table)
    except Error as e:
        print(e)
def checkTableExists(db_con):
    cursr=db_con.cursor()
    str="select name from sqlite_master where type='table'"
    table_names=cursr.execute(str)
    print("Tables in the database:")
    tables=table_names.fetchall()
    print(tables[0][0])
    return (len(tables))

def create_database_table(database,query):
    con=createDBConnection(database)
    if con is not None:
        createDBTable(con,query)
        checkTableExists(con)

    else:
        print("Error! cannot create database connection")
        con.close()
sql_create_table = """CREATE TABLE IF NOT EXISTS QuestionsProcessed (question
 text NOT NULL, code text, tags text, words_pre integer, words_post integer, i
s_code integer);"""
create_database_table("Processed.db", sql_create_table)
```

```
Tables in the database:
QuestionsProcessed
```

In [4]:
```
read_db="train_no_dup.db"
write_db="Processed.db"
if os.path.isfile("train_no_dup.db"):
    con=createDBConnection(read_db)
    if con is not None:
        reader =con.cursor()
        reader.execute("SELECT Title, Body, Tags From no_dup_train ORDER BY RA
NDOM() LIMIT 500000;")

if os.path.isfile("Processed.db"):
    conn=createDBConnection(write_db)
    if conn is not None:
        tables = checkTableExists(conn)
        writer=conn.cursor()
        if tables != 0:
            writer.execute("DELETE FROM QuestionsProcessed WHERE 1")
            print("Cleared All the rows")
```

```
Tables in the database:
QuestionsProcessed
Cleared All the rows
```

Preprocessing the Questions

In [5]:
```
preprocessed_data_list=[]
reader.fetchone()
questions_with_code=0
len_pre=0
len_post=0
questions_proccesed = 0

for row in reader:
    is_code = 0
    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')
    #adding tittle 3 times to increase its weight
    question=str(title)+" "+str(title)+" "+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 exceptt for th
e letter 'c'
    question=' '.join(str(stemmer.stem(j)) for j in words if j not in stop_wor
ds and (len(j)!=1 or j=='c'))

    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_pr
e,words_post,is_code) values (?,?,?,?,?,?)",tup)
    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_proccesed))
```

```
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: 1172
Avg. length of questions(Title+Body) after processing: 398
Percent of questions containing code: 57
```

In [6]:
```
con.commit()
conn.commit()
con.close()
conn.close()
```

In [7]:
```
if os.path.isfile(write_db):
    conn_r = createDBConnection(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
==================================================================================================
('help understand automapp help understand automapp help understand automapp point automapp could someon give realli simpl exampl watch video click sam',)
--------------------------------------------------------------------------------------------------
('html ie html ie html ie ie upgrad window xp goe work need button look link work everi browser valid fine http valid org',)
--------------------------------------------------------------------------------------------------
('xml xqueri interfac exist xml file xml xqueri interfac exist xml file xml xqueri interfac exist xml file compani educ industri use xml store cours content also store cours relat inform most metainfo relat databas right process switch proprietari xml schema docbook along switch want move cours relat inform databas xml file reason cours data one place put subvers howev would like keep flexibl relat databas abl easili extract specif inform cours xml document xqueri seem task research databas support far could find need basic want xml file certain directori structur top would like system would index file let select anyth file use xqueri way cake eat xqueri interfac still keep file plain text version anyth least remot resembl want think ask nonsens pleas make altern suggest relat note xml databas prefer nativ open sourc experi would recommend',)
--------------------------------------------------------------------------------------------------
('mysql insert static dynam valu mix mysql insert static dynam valu mix mysql insert static dynam valu mix tri syphon data one tabl anoth problem run data go new tabl static data copi exist tabl let start show queri tri run obvious php variabl declar beforehand essenti run shop cart queri would take item previous order enter new shop cart custom start new order base previous order problem run insert record tabl data record static cartid ponumbernew email lined inform come differ tabl hope without creat loop repeat queri howev mani item order custom duplic think would bog site signific seen myriad awesom answer mani web dev question past hope stackoverflow communiti help thank',)
--------------------------------------------------------------------------------------------------
('hive view error hive view error hive view error new hadoop hive tri creat one one hive view hive tabl get unknown host except given ddl tabl ncreat tabl cliam nclaim key int nclaim name string store sequencefil given view definit ncreat view claimant view select claim select view get follow except thrown nselect claimant view ntotal mapreduc job nlaunch job nnumber reduc task set sinc reduc oper nwarn org apach hadoop metric jvm eventcount deprec pleas use org apach hadoop log metric eventcount log properti file nexecut log tmp root root cd ee log njava net unknownhostexcept server az region geo localdomain server az region geo localdomain java net inetaddress getlocalhost inetaddress java org apach hadoop mapr jobclient run jobclient java org apach hadoop mapr jobclient run jobclient java java secur accesscontrol doprivileg nativ method javax secur auth subject doa subject java org apach hadoop secur usergroupinform doa usergroupinform java org apach hadoop mapr jobclient submitjobintern jobclient java org apach hadoop mapr jobclient submitjob jobclient java org apach hadoop hive ql exec execdriv execut execdriv java org apach hadoop hive ql exec execdriv main execdriv java sun reflect nativemethodaccessorimpl invok nativ method sun reflect nativemethodaccessorimpl invok nativemethodaccessorimpl java sun reflect delegatingmethodaccessorimpl invok delegatingmethodaccessorimpl java java lang reflect method invok method java org apach hadoop util runjar main runjar java njob submiss fail except java net unknownhostexcept server az region geo localdomain server az region geo localdomain nexecut f
```

```
ail exit status nobtain error inform task fail ntask id stage log tmp root hi
ve log someon kind help understand issu pleas let know need inform nthank adv
anc',)
----------------------------------------------------------------------------
-----------------------
('cakephp chang auth type prefix rout cakephp chang auth type prefix rout cak
ephp chang auth type prefix rout got cakephp websit start build rest api seem
like would easiest build second applic top cake core other suggest bad practi
c http stackoverflow com instead tri creat api within applic use prefix rout
mysit com api almost entir site behind login user want get past homepag need
log use cake new blowfish authent feel like might import note make problem ru
n like use basic authent api wrap head around suppos work first ad api prefix
rout core php rout php appcontrol php specifi blowfish auth type array ad fol
low method found littl document part even sure correct far userscontrol php a
d login action go mysit com api user login browser window give authent challe
ng accept credenti assum need use usernam password would work normal form bas
e authent wonder accept password blowfish password hash cancel auth popup dis
play flash messag abl login gah confus thank help advanc',)
----------------------------------------------------------------------------
-----------------------
('oftyp work oftyp work oftyp work oftyp work read link go exact linq provid
know get object match specifi type know chain request evalu call right specif
want know framework quick type comparison wrote method net project went like
sinc support kind featur best implement edit main concern fast',)
----------------------------------------------------------------------------
-----------------------
('make haskel glut use freeglut window make haskel glut use freeglut window m
ake haskel glut use freeglut window make haskel glut bind use freeglut instea
d origin glut window',)
----------------------------------------------------------------------------
-----------------------
('strang zfs disk space usag report zvol strang zfs disk space usag report zv
ol strang zfs disk space usag report zvol zvol freebsd current host claim use
disk space look like bug consum snapshot reserv children mayb miss someth upd
result result result upd creat number zvol differ paramet use move content no
tic anoth odd thing disk usag normal zvol remain abnorm zvol even fragment is
su',)
----------------------------------------------------------------------------
-----------------------
```

Saving Preprocessed data to a Database

In [5]:
```python
write_db = 'Processed.db'
if os.path.isfile(write_db):
    conn_r = createDBConnection(write_db)
    if conn_r is not None:
        preprocessed_data = pd.read_sql_query("""SELECT question, Tags FROM Qu
estionsProcessed""", conn_r)
conn_r.commit()
conn_r.close()
```

In [6]:
```
preprocessed_data.head(3)
```

Out[6]:

	question	tags
0	libcurl use cocoa libcurl use cocoa libcurl us...	objective-c cocoa libcurl
1	help understand automapp help understand autom...	asp.net asp.net-mvc automapper
2	html ie html ie html ie ie upgrad window xp go...	html internet-explorer-8 hyperlink

In [7]:
```
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
```

Converting tags to binary vector

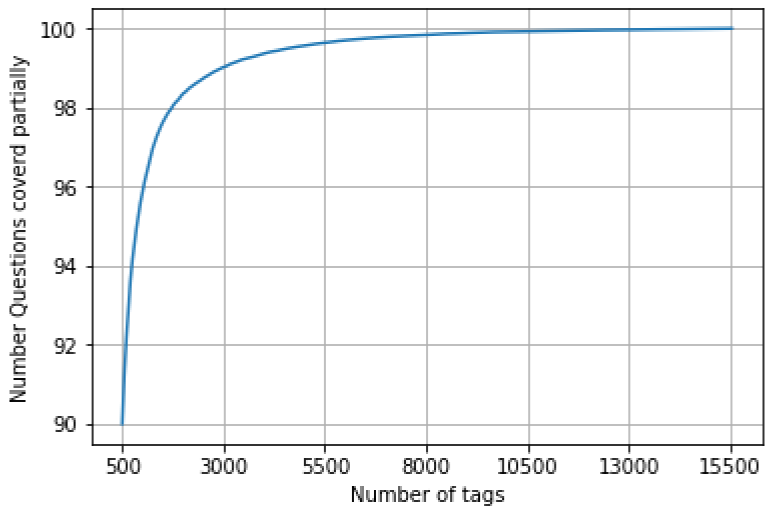
In [8]:
```
vectorizer=CountVectorizer(tokenizer= lambda x:x.split(" "),binary='true')
multilabel_y=vectorizer.fit_transform(preprocessed_data['tags'])
```

In [9]:
```
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 only 500 labels

In [10]:
```
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)*100,3))
```

In [16]:
```
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()
print("with ",5500,"tags we are covering ",questions_explained[50],"% of quest
ions")
```




```
with  5500 tags we are covering  99.023 % of questions
```

In [11]:
```
# we will be taking 5500 tags
multilabel_yx = tags_to_choose(500)
print("number of questions that are not covered :", questions_explained_fn(500
),"out of ", total_qs)
```

```
number of questions that are not covered : 49981 out of  499999
```

Splitting the data into train and test

In [12]:
```
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:total_size,:]
```

In [13]:
```
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 : (399999, 500)
Number of data points in test data : (100000, 500)
```

Featurizing the data using BOW

In [21]:
```
vectorizer=CountVectorizer(ngram_range=(1,4),max_features=200000,min_df=0.0000
9,tokenizer=lambda x:x.split())
X_train_multilabel=vectorizer.fit_transform(x_train['question'])
X_test_multilabel=vectorizer.transform(x_test['question'])
```

Logistic Regression OneVsRest Classifier

In [3]:
```
classfier_LR = OneVsRestClassifier(LogisticRegression(penalty='l1',C=0.1), n_j
obs=-1)
classfier_LR.fit(X_train_multilabel,y_train)
```

Out[3]:
```
OneVsRestClassifier(estimator=LogisticRegression(C=0.1, class_weight=None, du
al=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l1', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False),
          n_jobs=-1)
```

In [5]:
```
predictions = classfier_LR.predict(X_test_multilabel)
print("Accuracy :",accuracy_score(y_test, predictions))
print("Hamming loss ",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 (classification_report(y_test, predictions))
```

```
Accuracy : 0.2393
Hamming loss  0.00283104
Micro-average quality numbers
Precision: 0.7037, Recall: 0.3763, F1-measure: 0.4904
Macro-average quality numbers
Precision: 0.5275, Recall: 0.2963, F1-measure: 0.3679
             precision    recall  f1-score   support

          0       0.65      0.30      0.42      7841
          1       0.80      0.47      0.59      7130
          2       0.84      0.58      0.69      6823
          3       0.76      0.46      0.57      6329
          4       0.94      0.79      0.86      5555
          5       0.86      0.64      0.74      5275
          6       0.72      0.38      0.50      3461
          7       0.88      0.64      0.74      3148
          8       0.69      0.41      0.51      2989
          9       0.79      0.45      0.57      2933
         10       0.86      0.62      0.72      2852
         11       0.53      0.20      0.29      2930
         12       0.56      0.14      0.22      2715
         13       0.62      0.31      0.42      2427
         14       0.62      0.26      0.37      2283
         15       0.59      0.29      0.39      2187
         16       0.79      0.54      0.64      2296
         17       0.79      0.58      0.67      2037
         18       0.64      0.31      0.42      1808
         19       0.59      0.24      0.34      1648
         20       0.36      0.08      0.13      1554
         21       0.76      0.41      0.54      1311
         22       0.61      0.29      0.40      1220
         23       0.90      0.65      0.75      1081
         24       0.68      0.45      0.55      1116
         25       0.69      0.42      0.52      1083
         26       0.59      0.37      0.45      1019
         27       0.86      0.68      0.76       994
         28       0.36      0.08      0.13       992
         29       0.61      0.23      0.33       898
         30       0.96      0.76      0.85       884
         31       0.55      0.30      0.39       855
         32       0.63      0.34      0.44       815
         33       0.83      0.36      0.50       752
         34       0.53      0.21      0.30       783
         35       0.81      0.57      0.67       767
         36       0.76      0.64      0.69       770
         37       0.75      0.55      0.63       771
         38       0.37      0.16      0.22       760
         39       0.38      0.13      0.19       669
         40       0.71      0.42      0.53       664
         41       0.43      0.14      0.21       680
         42       0.67      0.32      0.43       614
         43       0.67      0.35      0.46       575
         44       0.39      0.12      0.18       603
         45       0.45      0.17      0.25       587
         46       0.51      0.17      0.25       537
         47       0.46      0.18      0.26       547
         48       0.28      0.03      0.06       567
```

49	0.30	0.08	0.13	590
50	0.89	0.75	0.82	564
51	0.53	0.20	0.29	569
52	0.81	0.46	0.58	511
53	0.73	0.50	0.60	550
54	0.41	0.17	0.24	493
55	0.58	0.34	0.43	520
56	0.39	0.12	0.18	527
57	0.57	0.21	0.31	542
58	0.75	0.52	0.61	530
59	0.40	0.09	0.15	500
60	0.92	0.84	0.87	516
61	0.16	0.03	0.05	482
62	0.78	0.54	0.64	485
63	0.95	0.69	0.80	479
64	0.81	0.30	0.44	487
65	0.72	0.29	0.41	423
66	0.65	0.30	0.41	441
67	0.78	0.53	0.63	484
68	0.60	0.25	0.35	437
69	0.44	0.17	0.24	466
70	0.79	0.53	0.64	438
71	0.70	0.41	0.52	427
72	0.80	0.51	0.62	425
73	0.19	0.02	0.04	427
74	0.68	0.47	0.56	399
75	0.92	0.74	0.82	431
76	0.47	0.24	0.32	450
77	0.56	0.35	0.43	399
78	0.11	0.01	0.03	408
79	0.35	0.10	0.16	380
80	0.41	0.25	0.31	385
81	0.60	0.33	0.43	347
82	0.44	0.16	0.24	362
83	0.66	0.42	0.51	335
84	0.58	0.26	0.36	360
85	0.76	0.54	0.63	383
86	0.85	0.57	0.69	354
87	0.92	0.67	0.77	392
88	0.96	0.61	0.75	363
89	0.83	0.65	0.73	381
90	0.81	0.50	0.62	347
91	0.52	0.10	0.17	336
92	0.51	0.18	0.26	340
93	0.65	0.47	0.55	336
94	0.72	0.54	0.62	336
95	0.30	0.07	0.11	331
96	0.16	0.04	0.06	311
97	0.86	0.75	0.80	313
98	0.47	0.25	0.33	315
99	0.95	0.70	0.80	346
100	0.62	0.22	0.33	307
101	0.43	0.14	0.21	341
102	0.68	0.45	0.54	354
103	0.91	0.65	0.76	315
104	0.92	0.76	0.83	288
105	0.69	0.47	0.56	316

106	0.19	0.04	0.06	296
107	0.88	0.65	0.75	313
108	0.39	0.19	0.25	286
109	0.41	0.10	0.16	320
110	0.80	0.43	0.56	325
111	0.69	0.41	0.51	291
112	0.54	0.26	0.35	294
113	0.67	0.38	0.48	296
114	0.39	0.21	0.27	310
115	0.51	0.22	0.31	264
116	0.46	0.19	0.27	282
117	0.41	0.14	0.20	280
118	0.91	0.76	0.83	249
119	0.34	0.14	0.20	242
120	0.55	0.17	0.26	270
121	0.66	0.37	0.48	262
122	0.83	0.68	0.75	268
123	0.69	0.45	0.55	249
124	0.61	0.29	0.40	273
125	0.95	0.85	0.90	258
126	0.93	0.70	0.80	287
127	0.37	0.17	0.23	227
128	0.34	0.08	0.13	267
129	0.44	0.23	0.30	257
130	0.14	0.00	0.01	242
131	0.19	0.05	0.08	244
132	0.39	0.13	0.20	240
133	0.47	0.28	0.35	251
134	0.31	0.12	0.18	238
135	0.52	0.14	0.21	266
136	0.64	0.45	0.53	262
137	0.58	0.42	0.49	239
138	0.80	0.57	0.67	273
139	0.82	0.54	0.65	225
140	0.50	0.30	0.37	257
141	0.64	0.32	0.42	272
142	0.92	0.76	0.83	228
143	0.24	0.05	0.09	256
144	0.37	0.09	0.15	235
145	0.52	0.25	0.34	262
146	0.56	0.30	0.39	217
147	0.21	0.06	0.10	247
148	0.15	0.03	0.05	236
149	0.29	0.07	0.12	261
150	0.38	0.14	0.20	229
151	0.53	0.31	0.39	237
152	0.70	0.45	0.55	233
153	0.86	0.71	0.78	227
154	0.66	0.40	0.50	245
155	0.76	0.49	0.60	235
156	0.39	0.12	0.19	227
157	0.50	0.11	0.18	261
158	0.45	0.22	0.29	240
159	0.37	0.11	0.17	253
160	0.94	0.82	0.88	226
161	0.45	0.11	0.18	235
162	0.51	0.20	0.28	227

163	0.62	0.45	0.52	221
164	0.26	0.06	0.10	231
165	0.54	0.17	0.26	241
166	0.25	0.08	0.12	226
167	0.62	0.26	0.37	257
168	0.53	0.17	0.26	226
169	0.33	0.09	0.15	213
170	0.41	0.21	0.28	225
171	0.78	0.46	0.57	224
172	0.74	0.49	0.59	207
173	0.70	0.48	0.57	219
174	0.53	0.20	0.29	232
175	0.90	0.77	0.83	205
176	0.90	0.74	0.81	196
177	0.33	0.07	0.12	196
178	0.85	0.73	0.78	193
179	0.81	0.64	0.72	225
180	0.47	0.21	0.29	224
181	0.20	0.04	0.07	186
182	0.25	0.08	0.12	200
183	0.80	0.54	0.64	216
184	0.34	0.07	0.12	202
185	0.71	0.51	0.59	226
186	0.62	0.21	0.32	219
187	0.34	0.14	0.19	214
188	0.50	0.20	0.29	214
189	0.81	0.51	0.63	243
190	0.93	0.69	0.79	200
191	0.60	0.23	0.34	188
192	0.83	0.73	0.78	159
193	0.56	0.31	0.40	189
194	0.26	0.04	0.07	208
195	0.88	0.64	0.74	183
196	0.26	0.08	0.12	222
197	0.28	0.08	0.13	207
198	0.51	0.19	0.27	194
199	0.52	0.24	0.33	190
200	0.24	0.08	0.12	197
201	0.55	0.15	0.23	181
202	0.71	0.45	0.55	185
203	0.94	0.66	0.78	180
204	0.74	0.43	0.54	175
205	0.49	0.20	0.29	200
206	0.72	0.40	0.51	199
207	0.93	0.81	0.86	177
208	0.50	0.15	0.23	184
209	0.20	0.02	0.04	175
210	0.37	0.13	0.19	183
211	0.35	0.09	0.15	172
212	0.05	0.01	0.01	164
213	0.78	0.54	0.64	186
214	0.95	0.80	0.87	208
215	0.29	0.09	0.13	184
216	0.72	0.36	0.48	192
217	0.61	0.26	0.37	179
218	0.35	0.11	0.16	196
219	0.68	0.51	0.58	174

220	0.93	0.74	0.83	185
221	0.37	0.09	0.15	172
222	0.49	0.25	0.33	165
223	0.54	0.38	0.44	170
224	0.61	0.40	0.48	162
225	0.54	0.27	0.36	187
226	0.63	0.38	0.47	172
227	0.38	0.12	0.18	197
228	0.93	0.78	0.85	174
229	0.67	0.50	0.57	158
230	0.29	0.03	0.05	166
231	0.75	0.57	0.65	160
232	0.63	0.34	0.44	169
233	0.65	0.36	0.46	155
234	0.79	0.55	0.65	169
235	0.69	0.36	0.47	152
236	0.56	0.40	0.47	162
237	0.29	0.05	0.09	174
238	0.67	0.38	0.49	172
239	0.16	0.03	0.05	155
240	0.30	0.11	0.16	170
241	0.43	0.20	0.27	165
242	0.54	0.24	0.33	168
243	0.59	0.40	0.48	154
244	0.11	0.02	0.03	163
245	0.33	0.16	0.22	170
246	0.12	0.01	0.01	172
247	0.75	0.58	0.65	138
248	0.66	0.36	0.46	143
249	0.29	0.08	0.12	145
250	0.19	0.04	0.07	158
251	0.62	0.21	0.32	160
252	0.37	0.18	0.24	159
253	0.35	0.13	0.19	154
254	0.23	0.09	0.13	159
255	0.41	0.19	0.26	172
256	0.76	0.43	0.55	159
257	0.33	0.09	0.14	152
258	0.59	0.35	0.44	153
259	0.79	0.57	0.66	158
260	0.36	0.05	0.09	153
261	0.89	0.68	0.78	146
262	0.21	0.03	0.05	131
263	0.43	0.14	0.22	140
264	0.36	0.22	0.27	134
265	0.70	0.50	0.58	135
266	0.60	0.28	0.38	149
267	0.77	0.34	0.48	154
268	0.89	0.75	0.81	130
269	0.44	0.19	0.27	135
270	0.94	0.82	0.88	139
271	0.77	0.47	0.59	135
272	0.44	0.34	0.38	136
273	0.34	0.15	0.20	137
274	0.58	0.31	0.40	134
275	0.68	0.58	0.63	136
276	0.59	0.15	0.24	144

277	0.61	0.34	0.44	140
278	0.00	0.00	0.00	148
279	0.68	0.43	0.53	150
280	0.20	0.01	0.01	133
281	0.19	0.03	0.05	131
282	0.72	0.37	0.49	161
283	0.69	0.50	0.58	138
284	0.36	0.08	0.13	124
285	0.83	0.69	0.75	137
286	0.29	0.08	0.13	137
287	0.94	0.70	0.80	145
288	0.56	0.30	0.39	133
289	0.47	0.31	0.37	107
290	0.36	0.15	0.21	136
291	0.11	0.01	0.02	146
292	0.29	0.14	0.19	123
293	0.50	0.20	0.29	145
294	0.11	0.02	0.03	132
295	0.17	0.02	0.04	125
296	0.40	0.08	0.13	125
297	0.28	0.10	0.14	126
298	0.06	0.01	0.01	125
299	0.81	0.61	0.69	125
300	0.38	0.08	0.13	135
301	0.64	0.27	0.38	142
302	0.46	0.27	0.34	107
303	0.21	0.02	0.04	138
304	0.51	0.23	0.32	127
305	0.78	0.54	0.64	116
306	0.94	0.74	0.83	128
307	0.20	0.09	0.12	127
308	0.72	0.38	0.50	132
309	0.40	0.14	0.20	124
310	0.43	0.22	0.29	118
311	0.38	0.14	0.21	125
312	0.56	0.30	0.39	122
313	0.56	0.36	0.44	124
314	0.27	0.08	0.12	124
315	0.51	0.30	0.38	132
316	0.73	0.51	0.60	127
317	0.29	0.12	0.17	100
318	0.33	0.10	0.16	108
319	0.40	0.08	0.13	103
320	0.19	0.09	0.13	95
321	0.20	0.07	0.10	117
322	0.52	0.30	0.38	119
323	0.57	0.30	0.39	121
324	0.42	0.18	0.26	120
325	0.12	0.03	0.04	119
326	0.39	0.22	0.28	109
327	0.54	0.38	0.44	119
328	0.53	0.29	0.37	121
329	0.82	0.60	0.69	103
330	0.30	0.11	0.16	124
331	0.21	0.03	0.05	117
332	0.15	0.05	0.08	113
333	0.69	0.43	0.53	126

334	0.19	0.04	0.07	114
335	0.43	0.25	0.31	105
336	0.51	0.28	0.36	127
337	0.43	0.21	0.28	109
338	0.41	0.23	0.29	102
339	0.50	0.19	0.28	118
340	0.32	0.06	0.10	107
341	0.35	0.12	0.18	106
342	0.09	0.02	0.03	129
343	0.24	0.04	0.07	117
344	0.42	0.16	0.23	115
345	0.34	0.09	0.14	109
346	0.29	0.02	0.03	116
347	0.21	0.07	0.11	99
348	0.95	0.69	0.80	101
349	0.38	0.16	0.22	108
350	0.67	0.41	0.51	115
351	0.46	0.31	0.37	105
352	0.12	0.05	0.07	99
353	0.93	0.89	0.91	96
354	0.12	0.02	0.03	111
355	0.43	0.08	0.14	112
356	0.80	0.34	0.47	119
357	0.46	0.23	0.31	118
358	0.69	0.40	0.51	105
359	0.52	0.35	0.41	113
360	0.17	0.02	0.03	115
361	0.87	0.58	0.70	113
362	0.97	0.82	0.89	109
363	0.44	0.22	0.29	110
364	0.62	0.34	0.44	100
365	0.79	0.50	0.62	103
366	0.26	0.15	0.19	102
367	0.64	0.34	0.44	113
368	0.64	0.43	0.52	99
369	0.09	0.01	0.02	84
370	0.58	0.39	0.47	109
371	0.16	0.05	0.07	109
372	0.74	0.56	0.64	98
373	0.72	0.41	0.52	122
374	0.23	0.04	0.07	115
375	0.56	0.29	0.38	122
376	0.21	0.08	0.11	105
377	0.11	0.03	0.05	103
378	0.40	0.21	0.28	89
379	0.21	0.07	0.11	110
380	0.75	0.55	0.64	89
381	0.25	0.11	0.15	98
382	0.38	0.12	0.18	108
383	0.11	0.02	0.04	95
384	0.57	0.22	0.32	108
385	0.21	0.06	0.09	99
386	0.45	0.30	0.36	88
387	0.91	0.68	0.78	107
388	0.71	0.45	0.55	104
389	0.50	0.09	0.15	90
390	0.45	0.19	0.26	102

391	0.81	0.61	0.69	102
392	0.40	0.16	0.23	104
393	0.91	0.53	0.67	110
394	0.28	0.09	0.14	95
395	0.15	0.04	0.06	103
396	0.33	0.07	0.11	104
397	0.93	0.82	0.87	93
398	0.89	0.66	0.76	85
399	0.21	0.04	0.07	114
400	0.78	0.55	0.64	102
401	0.00	0.00	0.00	102
402	0.25	0.06	0.10	101
403	0.57	0.29	0.39	93
404	0.11	0.02	0.03	100
405	0.81	0.54	0.65	100
406	0.37	0.16	0.23	80
407	0.83	0.51	0.63	106
408	0.80	0.61	0.69	94
409	0.50	0.13	0.21	97
410	0.19	0.07	0.10	104
411	0.62	0.22	0.32	106
412	0.36	0.17	0.23	81
413	0.75	0.44	0.55	100
414	0.24	0.07	0.11	102
415	0.29	0.08	0.12	88
416	0.53	0.37	0.43	79
417	0.23	0.09	0.13	95
418	0.61	0.37	0.46	91
419	0.43	0.06	0.11	99
420	0.44	0.28	0.34	92
421	0.31	0.14	0.19	81
422	0.57	0.34	0.43	85
423	0.86	0.61	0.72	83
424	0.41	0.13	0.20	105
425	0.66	0.46	0.54	91
426	0.62	0.16	0.26	98
427	0.40	0.18	0.25	103
428	0.30	0.13	0.18	90
429	0.22	0.07	0.10	92
430	0.57	0.14	0.22	86
431	0.42	0.18	0.26	93
432	0.48	0.14	0.22	97
433	0.25	0.11	0.15	94
434	0.09	0.01	0.02	81
435	0.72	0.35	0.47	103
436	0.36	0.09	0.15	96
437	0.31	0.13	0.18	84
438	0.92	0.75	0.83	81
439	0.46	0.24	0.32	74
440	0.15	0.05	0.08	75
441	0.06	0.01	0.02	89
442	0.52	0.29	0.37	90
443	0.31	0.05	0.09	75
444	0.34	0.12	0.18	91
445	0.24	0.08	0.12	96
446	0.56	0.22	0.31	92
447	0.80	0.61	0.69	83

448	0.72	0.40	0.51	86
449	0.59	0.26	0.36	100
450	0.74	0.46	0.57	84
451	0.50	0.17	0.26	98
452	0.14	0.01	0.02	94
453	0.00	0.00	0.00	96
454	0.17	0.04	0.06	78
455	0.91	0.72	0.80	102
456	0.85	0.65	0.74	82
457	0.38	0.13	0.19	94
458	0.70	0.38	0.49	88
459	0.69	0.50	0.58	90
460	0.31	0.17	0.22	72
461	0.45	0.18	0.26	83
462	0.85	0.52	0.64	91
463	0.33	0.04	0.07	82
464	0.33	0.03	0.06	95
465	0.58	0.16	0.25	89
466	0.67	0.53	0.59	72
467	0.86	0.53	0.65	104
468	0.17	0.05	0.08	94
469	0.71	0.57	0.63	80
470	0.48	0.25	0.33	80
471	0.96	0.64	0.77	84
472	0.11	0.02	0.04	85
473	0.43	0.24	0.31	84
474	0.25	0.09	0.14	85
475	0.16	0.04	0.06	76
476	0.42	0.22	0.29	99
477	0.88	0.79	0.83	89
478	0.40	0.22	0.28	88
479	0.14	0.03	0.04	80
480	0.45	0.14	0.21	72
481	0.67	0.41	0.51	76
482	0.32	0.08	0.13	83
483	0.76	0.76	0.76	80
484	0.09	0.02	0.04	81
485	0.19	0.06	0.09	81
486	0.79	0.40	0.53	82
487	0.16	0.04	0.06	81
488	0.26	0.12	0.17	82
489	0.72	0.54	0.62	78
490	0.96	0.82	0.89	96
491	0.52	0.26	0.34	90
492	0.52	0.29	0.37	83
493	0.24	0.06	0.10	79
494	0.40	0.19	0.26	90
495	0.55	0.53	0.54	77
496	0.55	0.28	0.37	76
497	0.48	0.28	0.35	83
498	0.39	0.08	0.13	88
499	0.33	0.09	0.14	92
avg / total	0.64	0.38	0.46	180980

SGD- Classifer with hinge loss (Linear SVM)

In [ ]:
```
classifier_SVM = OneVsRestClassifier(SGDClassifier(loss='hinge', alpha=0.00001
, penalty='l1'), n_jobs=-1)
classifier_SVM.fit(X_train_multilabel, y_train)
```

In [8]:
```
predictions_svm = classifier_SVM.predict(X_test_multilabel)
print("Accuracy :",accuracy_score(y_test, predictions_svm))
print("Hamming loss ",hamming_loss(y_test,predictions_svm))


precision = precision_score(y_test, predictions_svm, average='micro')
recall = recall_score(y_test, predictions_svm, average='micro')
f1 = f1_score(y_test, predictions_svm, 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_svm, average='macro')
recall = recall_score(y_test, predictions_svm, average='macro')
f1 = f1_score(y_test, predictions_svm, average='macro')

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

print (classification_report(y_test, predictions_svm))
```

```
Accuracy : 0.13656
Hamming loss  0.00475192
Micro-average quality numbers
Precision: 0.3747, Recall: 0.4676, F1-measure: 0.4160
Macro-average quality numbers
Precision: 0.2790, Recall: 0.3842, F1-measure: 0.3208
             precision    recall  f1-score   support

          0       0.46      0.43      0.44      7841
          1       0.57      0.57      0.57      7130
          2       0.64      0.65      0.64      6823
          3       0.54      0.56      0.55      6329
          4       0.83      0.84      0.84      5555
          5       0.70      0.70      0.70      5275
          6       0.48      0.50      0.49      3461
          7       0.66      0.72      0.69      3148
          8       0.50      0.51      0.50      2989
          9       0.51      0.55      0.53      2933
         10       0.63      0.67      0.65      2852
         11       0.34      0.33      0.34      2930
         12       0.26      0.24      0.25      2715
         13       0.42      0.44      0.43      2427
         14       0.40      0.44      0.42      2283
         15       0.38      0.39      0.38      2187
         16       0.60      0.59      0.59      2296
         17       0.59      0.61      0.60      2037
         18       0.34      0.40      0.37      1808
         19       0.36      0.41      0.38      1648
         20       0.24      0.26      0.25      1554
         21       0.47      0.53      0.50      1311
         22       0.38      0.45      0.41      1220
         23       0.67      0.73      0.70      1081
         24       0.48      0.51      0.49      1116
         25       0.38      0.46      0.42      1083
         26       0.37      0.42      0.39      1019
         27       0.65      0.73      0.69       994
         28       0.16      0.20      0.18       992
         29       0.25      0.34      0.29       898
         30       0.75      0.80      0.78       884
         31       0.34      0.45      0.39       855
         32       0.38      0.41      0.39       815
         33       0.33      0.47      0.39       752
         34       0.28      0.36      0.31       783
         35       0.54      0.59      0.57       767
         36       0.61      0.67      0.64       770
         37       0.47      0.62      0.54       771
         38       0.24      0.29      0.26       760
         39       0.21      0.27      0.24       669
         40       0.42      0.51      0.46       664
         41       0.18      0.23      0.20       680
         42       0.35      0.43      0.39       614
         43       0.29      0.39      0.33       575
         44       0.17      0.21      0.19       603
         45       0.26      0.33      0.29       587
         46       0.16      0.28      0.21       537
         47       0.24      0.31      0.27       547
         48       0.14      0.18      0.16       567
```

49	0.18	0.22	0.20	590
50	0.68	0.79	0.73	564
51	0.24	0.30	0.27	569
52	0.45	0.54	0.49	511
53	0.48	0.51	0.50	550
54	0.21	0.26	0.23	493
55	0.32	0.40	0.35	520
56	0.19	0.24	0.22	527
57	0.25	0.31	0.28	542
58	0.46	0.60	0.52	530
59	0.16	0.20	0.18	500
60	0.78	0.87	0.83	516
61	0.11	0.16	0.13	482
62	0.46	0.58	0.51	485
63	0.64	0.72	0.68	479
64	0.38	0.46	0.41	487
65	0.32	0.41	0.36	423
66	0.29	0.37	0.33	441
67	0.51	0.62	0.56	484
68	0.29	0.34	0.31	437
69	0.22	0.28	0.25	466
70	0.50	0.58	0.53	438
71	0.36	0.48	0.41	427
72	0.50	0.65	0.56	425
73	0.09	0.13	0.10	427
74	0.41	0.53	0.46	399
75	0.65	0.78	0.71	431
76	0.33	0.38	0.35	450
77	0.38	0.47	0.42	399
78	0.08	0.10	0.09	408
79	0.16	0.21	0.18	380
80	0.27	0.37	0.31	385
81	0.28	0.41	0.33	347
82	0.18	0.23	0.20	362
83	0.33	0.46	0.38	335
84	0.28	0.32	0.30	360
85	0.42	0.58	0.49	383
86	0.50	0.65	0.56	354
87	0.68	0.77	0.72	392
88	0.50	0.67	0.58	363
89	0.63	0.72	0.67	381
90	0.45	0.59	0.51	347
91	0.17	0.21	0.19	336
92	0.18	0.26	0.21	340
93	0.37	0.46	0.41	336
94	0.41	0.57	0.48	336
95	0.11	0.17	0.13	331
96	0.10	0.12	0.11	311
97	0.68	0.81	0.74	313
98	0.24	0.37	0.29	315
99	0.72	0.79	0.75	346
100	0.20	0.28	0.24	307
101	0.18	0.24	0.21	341
102	0.38	0.53	0.44	354
103	0.63	0.75	0.68	315
104	0.59	0.79	0.68	288
105	0.36	0.51	0.42	316

106	0.11	0.19	0.14	296
107	0.59	0.73	0.65	313
108	0.17	0.26	0.21	286
109	0.16	0.26	0.20	320
110	0.42	0.48	0.45	325
111	0.39	0.54	0.45	291
112	0.26	0.34	0.30	294
113	0.33	0.46	0.39	296
114	0.21	0.32	0.26	310
115	0.24	0.37	0.29	264
116	0.20	0.27	0.23	282
117	0.13	0.23	0.17	280
118	0.71	0.82	0.76	249
119	0.13	0.20	0.16	242
120	0.20	0.29	0.23	270
121	0.39	0.55	0.46	262
122	0.53	0.73	0.62	268
123	0.40	0.52	0.45	249
124	0.25	0.36	0.29	273
125	0.79	0.88	0.83	258
126	0.65	0.80	0.72	287
127	0.18	0.27	0.22	227
128	0.13	0.20	0.16	267
129	0.27	0.40	0.32	257
130	0.01	0.01	0.01	242
131	0.06	0.07	0.07	244
132	0.16	0.27	0.20	240
133	0.23	0.35	0.28	251
134	0.20	0.26	0.22	238
135	0.17	0.22	0.19	266
136	0.39	0.51	0.44	262
137	0.30	0.43	0.35	239
138	0.52	0.67	0.59	273
139	0.39	0.56	0.46	225
140	0.33	0.41	0.36	257
141	0.33	0.33	0.33	272
142	0.66	0.82	0.73	228
143	0.11	0.18	0.13	256
144	0.12	0.17	0.14	235
145	0.21	0.34	0.26	262
146	0.30	0.53	0.39	217
147	0.12	0.16	0.14	247
148	0.06	0.12	0.08	236
149	0.11	0.18	0.13	261
150	0.13	0.22	0.17	229
151	0.27	0.44	0.33	237
152	0.34	0.50	0.41	233
153	0.64	0.77	0.70	227
154	0.41	0.59	0.48	245
155	0.41	0.59	0.48	235
156	0.13	0.17	0.14	227
157	0.14	0.21	0.16	261
158	0.26	0.34	0.29	240
159	0.21	0.28	0.24	253
160	0.78	0.90	0.84	226
161	0.14	0.20	0.16	235
162	0.26	0.35	0.30	227

163	0.41	0.64	0.50	221
164	0.11	0.13	0.12	231
165	0.16	0.25	0.19	241
166	0.15	0.25	0.19	226
167	0.29	0.38	0.33	257
168	0.18	0.25	0.21	226
169	0.12	0.20	0.15	213
170	0.22	0.33	0.26	225
171	0.35	0.54	0.42	224
172	0.36	0.54	0.43	207
173	0.43	0.57	0.49	219
174	0.30	0.34	0.32	232
175	0.61	0.83	0.71	205
176	0.60	0.81	0.69	196
177	0.07	0.13	0.09	196
178	0.59	0.75	0.66	193
179	0.54	0.67	0.60	225
180	0.20	0.33	0.25	224
181	0.08	0.13	0.10	186
182	0.11	0.20	0.14	200
183	0.44	0.63	0.52	216
184	0.05	0.08	0.07	202
185	0.38	0.62	0.48	226
186	0.19	0.23	0.21	219
187	0.18	0.27	0.22	214
188	0.22	0.36	0.27	214
189	0.50	0.61	0.55	243
190	0.68	0.78	0.72	200
191	0.18	0.28	0.22	188
192	0.54	0.82	0.65	159
193	0.35	0.46	0.40	189
194	0.05	0.09	0.07	208
195	0.51	0.70	0.59	183
196	0.14	0.20	0.17	222
197	0.10	0.15	0.12	207
198	0.20	0.31	0.24	194
199	0.29	0.41	0.34	190
200	0.14	0.18	0.16	197
201	0.16	0.28	0.21	181
202	0.31	0.45	0.36	185
203	0.54	0.77	0.64	180
204	0.38	0.51	0.43	175
205	0.20	0.29	0.24	200
206	0.41	0.50	0.45	199
207	0.69	0.82	0.75	177
208	0.17	0.28	0.21	184
209	0.08	0.13	0.10	175
210	0.17	0.30	0.22	183
211	0.08	0.15	0.10	172
212	0.03	0.05	0.04	164
213	0.42	0.64	0.51	186
214	0.79	0.85	0.82	208
215	0.13	0.22	0.17	184
216	0.31	0.49	0.38	192
217	0.28	0.39	0.32	179
218	0.09	0.15	0.11	196
219	0.37	0.53	0.43	174

220	0.66	0.79	0.72	185
221	0.10	0.20	0.13	172
222	0.21	0.35	0.26	165
223	0.26	0.41	0.32	170
224	0.30	0.45	0.36	162
225	0.28	0.41	0.33	187
226	0.31	0.47	0.37	172
227	0.16	0.25	0.20	197
228	0.66	0.77	0.71	174
229	0.35	0.58	0.44	158
230	0.06	0.10	0.07	166
231	0.37	0.63	0.47	160
232	0.34	0.43	0.38	169
233	0.30	0.47	0.37	155
234	0.47	0.59	0.52	169
235	0.31	0.45	0.37	152
236	0.33	0.55	0.41	162
237	0.04	0.08	0.06	174
238	0.32	0.48	0.39	172
239	0.06	0.12	0.08	155
240	0.17	0.28	0.21	170
241	0.15	0.26	0.19	165
242	0.29	0.39	0.33	168
243	0.32	0.47	0.38	154
244	0.05	0.08	0.06	163
245	0.16	0.26	0.20	170
246	0.04	0.07	0.05	172
247	0.45	0.64	0.53	138
248	0.29	0.43	0.35	143
249	0.09	0.15	0.11	145
250	0.06	0.12	0.08	158
251	0.15	0.24	0.18	160
252	0.23	0.37	0.28	159
253	0.17	0.29	0.21	154
254	0.13	0.27	0.18	159
255	0.12	0.20	0.15	172
256	0.38	0.58	0.46	159
257	0.09	0.16	0.12	152
258	0.32	0.45	0.37	153
259	0.50	0.58	0.54	158
260	0.03	0.06	0.04	153
261	0.60	0.74	0.66	146
262	0.05	0.08	0.06	131
263	0.11	0.27	0.15	140
264	0.21	0.34	0.26	134
265	0.39	0.60	0.47	135
266	0.29	0.46	0.36	149
267	0.33	0.45	0.38	154
268	0.58	0.81	0.68	130
269	0.17	0.27	0.21	135
270	0.61	0.85	0.71	139
271	0.50	0.63	0.56	135
272	0.17	0.29	0.21	136
273	0.20	0.31	0.24	137
274	0.23	0.40	0.29	134
275	0.45	0.62	0.52	136
276	0.18	0.27	0.21	144

277	0.29	0.44	0.35	140
278	0.00	0.01	0.00	148
279	0.33	0.44	0.37	150
280	0.03	0.05	0.04	133
281	0.03	0.05	0.04	131
282	0.40	0.47	0.43	161
283	0.42	0.54	0.47	138
284	0.04	0.08	0.05	124
285	0.57	0.77	0.65	137
286	0.10	0.21	0.14	137
287	0.59	0.70	0.64	145
288	0.21	0.43	0.28	133
289	0.24	0.46	0.31	107
290	0.12	0.22	0.16	136
291	0.05	0.07	0.06	146
292	0.12	0.23	0.16	123
293	0.20	0.28	0.23	145
294	0.08	0.10	0.09	132
295	0.10	0.17	0.12	125
296	0.10	0.16	0.12	125
297	0.08	0.11	0.09	126
298	0.07	0.10	0.08	125
299	0.54	0.68	0.60	125
300	0.14	0.24	0.18	135
301	0.26	0.37	0.31	142
302	0.16	0.33	0.22	107
303	0.06	0.12	0.08	138
304	0.21	0.33	0.26	127
305	0.39	0.59	0.47	116
306	0.80	0.80	0.80	128
307	0.09	0.15	0.11	127
308	0.26	0.42	0.32	132
309	0.12	0.21	0.15	124
310	0.17	0.31	0.22	118
311	0.14	0.26	0.18	125
312	0.28	0.48	0.35	122
313	0.31	0.44	0.37	124
314	0.09	0.17	0.12	124
315	0.21	0.32	0.25	132
316	0.48	0.65	0.55	127
317	0.17	0.29	0.21	100
318	0.07	0.14	0.09	108
319	0.07	0.14	0.09	103
320	0.06	0.14	0.09	95
321	0.05	0.08	0.06	117
322	0.26	0.43	0.33	119
323	0.28	0.41	0.33	121
324	0.15	0.32	0.21	120
325	0.09	0.16	0.12	119
326	0.17	0.33	0.23	109
327	0.34	0.52	0.41	119
328	0.21	0.34	0.26	121
329	0.51	0.61	0.56	103
330	0.15	0.27	0.20	124
331	0.01	0.03	0.02	117
332	0.08	0.17	0.11	113
333	0.39	0.55	0.46	126

334	0.09	0.17	0.12	114
335	0.23	0.39	0.29	105
336	0.26	0.35	0.30	127
337	0.35	0.43	0.38	109
338	0.27	0.38	0.32	102
339	0.14	0.20	0.16	118
340	0.06	0.12	0.08	107
341	0.13	0.22	0.16	106
342	0.06	0.09	0.07	129
343	0.05	0.07	0.06	117
344	0.22	0.24	0.23	115
345	0.12	0.21	0.15	109
346	0.02	0.05	0.03	116
347	0.09	0.18	0.12	99
348	0.57	0.81	0.67	101
349	0.17	0.31	0.22	108
350	0.32	0.50	0.39	115
351	0.22	0.43	0.29	105
352	0.07	0.17	0.10	99
353	0.78	0.93	0.85	96
354	0.07	0.11	0.09	111
355	0.10	0.13	0.11	112
356	0.23	0.37	0.29	119
357	0.27	0.41	0.32	118
358	0.22	0.43	0.29	105
359	0.31	0.50	0.38	113
360	0.05	0.08	0.06	115
361	0.52	0.71	0.60	113
362	0.72	0.83	0.77	109
363	0.21	0.35	0.26	110
364	0.22	0.44	0.29	100
365	0.45	0.59	0.51	103
366	0.11	0.22	0.15	102
367	0.29	0.44	0.35	113
368	0.33	0.53	0.41	99
369	0.02	0.04	0.02	84
370	0.39	0.58	0.47	109
371	0.09	0.17	0.12	109
372	0.42	0.60	0.50	98
373	0.29	0.43	0.35	122
374	0.06	0.10	0.07	115
375	0.23	0.43	0.30	122
376	0.10	0.16	0.13	105
377	0.05	0.11	0.07	103
378	0.23	0.35	0.28	89
379	0.15	0.21	0.17	110
380	0.34	0.58	0.43	89
381	0.20	0.30	0.24	98
382	0.19	0.28	0.22	108
383	0.02	0.05	0.03	95
384	0.33	0.36	0.35	108
385	0.04	0.07	0.05	99
386	0.21	0.35	0.26	88
387	0.57	0.75	0.65	107
388	0.37	0.51	0.43	104
389	0.07	0.12	0.09	90
390	0.12	0.16	0.13	102

391	0.45	0.69	0.55	102
392	0.10	0.22	0.14	104
393	0.49	0.65	0.56	110
394	0.12	0.26	0.17	95
395	0.06	0.13	0.08	103
396	0.07	0.12	0.09	104
397	0.65	0.86	0.74	93
398	0.53	0.75	0.62	85
399	0.06	0.10	0.07	114
400	0.47	0.71	0.56	102
401	0.07	0.14	0.09	102
402	0.09	0.16	0.12	101
403	0.17	0.27	0.21	93
404	0.10	0.11	0.10	100
405	0.40	0.63	0.49	100
406	0.11	0.26	0.16	80
407	0.49	0.62	0.55	106
408	0.46	0.66	0.54	94
409	0.07	0.13	0.09	97
410	0.16	0.22	0.18	104
411	0.20	0.32	0.25	106
412	0.11	0.31	0.17	81
413	0.46	0.61	0.52	100
414	0.09	0.19	0.12	102
415	0.08	0.16	0.11	88
416	0.19	0.42	0.26	79
417	0.12	0.17	0.14	95
418	0.31	0.46	0.37	91
419	0.06	0.11	0.08	99
420	0.26	0.36	0.30	92
421	0.19	0.41	0.26	81
422	0.29	0.52	0.37	85
423	0.42	0.67	0.52	83
424	0.16	0.25	0.20	105
425	0.44	0.64	0.52	91
426	0.16	0.21	0.18	98
427	0.26	0.33	0.29	103
428	0.15	0.27	0.19	90
429	0.07	0.13	0.09	92
430	0.13	0.26	0.18	86
431	0.16	0.26	0.19	93
432	0.11	0.20	0.14	97
433	0.12	0.27	0.17	94
434	0.02	0.05	0.03	81
435	0.35	0.41	0.38	103
436	0.10	0.18	0.13	96
437	0.11	0.20	0.14	84
438	0.58	0.79	0.67	81
439	0.21	0.30	0.24	74
440	0.05	0.13	0.07	75
441	0.02	0.04	0.03	89
442	0.22	0.39	0.28	90
443	0.03	0.05	0.04	75
444	0.09	0.19	0.12	91
445	0.13	0.26	0.18	96
446	0.27	0.48	0.35	92
447	0.47	0.71	0.57	83

448	0.34	0.43	0.38	86
449	0.27	0.39	0.32	100
450	0.42	0.64	0.51	84
451	0.16	0.28	0.20	98
452	0.01	0.02	0.02	94
453	0.03	0.04	0.03	96
454	0.03	0.05	0.04	78
455	0.69	0.69	0.69	102
456	0.43	0.66	0.52	82
457	0.17	0.29	0.21	94
458	0.31	0.50	0.39	88
459	0.36	0.48	0.41	90
460	0.18	0.32	0.23	72
461	0.18	0.30	0.23	83
462	0.41	0.55	0.47	91
463	0.02	0.05	0.03	82
464	0.07	0.13	0.09	95
465	0.13	0.22	0.17	89
466	0.45	0.57	0.50	72
467	0.53	0.64	0.58	104
468	0.10	0.16	0.12	94
469	0.39	0.59	0.47	80
470	0.20	0.39	0.27	80
471	0.56	0.70	0.62	84
472	0.02	0.02	0.02	85
473	0.18	0.33	0.24	84
474	0.13	0.27	0.18	85
475	0.02	0.05	0.03	76
476	0.15	0.23	0.18	99
477	0.60	0.83	0.69	89
478	0.18	0.34	0.23	88
479	0.04	0.06	0.05	80
480	0.13	0.24	0.17	72
481	0.26	0.42	0.32	76
482	0.12	0.22	0.15	83
483	0.53	0.81	0.64	80
484	0.04	0.06	0.05	81
485	0.10	0.19	0.13	81
486	0.35	0.51	0.42	82
487	0.08	0.16	0.11	81
488	0.19	0.35	0.24	82
489	0.45	0.64	0.53	78
490	0.70	0.91	0.79	96
491	0.19	0.39	0.25	90
492	0.24	0.41	0.30	83
493	0.04	0.09	0.05	79
494	0.17	0.24	0.20	90
495	0.38	0.53	0.45	77
496	0.21	0.37	0.27	76
497	0.20	0.33	0.25	83
498	0.16	0.28	0.20	88
499	0.08	0.15	0.10	92
avg / total	0.41	0.47	0.43	180980

LinearSVM using GridSearchCV

In [ ]:
```
classifier2_SVM = OneVsRestClassifier(SGDClassifier(loss='hinge',penalty='l1'
))
parameters = {'estimator__alpha':[0.001,0.01,0.1,10,100]}
gridsearch_SVM = GridSearchCV(classifier2_SVM,param_grid = parameters, scoring
='f1_micro',n_jobs = -1)
gridsearch_SVM.fit(X_train_multilabel,y_train)
```

In [5]:
```
predictions_svm_grid = gridsearch_SVM.predict(X_test_multilabel)
print("Accuracy :",accuracy_score(y_test, predictions_svm_grid))
print("Hamming loss ",hamming_loss(y_test,predictions_svm_grid))


precision = precision_score(y_test, predictions_svm_grid, average='micro')
recall = recall_score(y_test, predictions_svm_grid, average='micro')
f1 = f1_score(y_test, predictions_svm_grid, 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_svm_grid, average='macro')
recall = recall_score(y_test, predictions_svm_grid, average='macro')
f1 = f1_score(y_test, predictions_svm_grid, average='macro')

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

print (classification_report(y_test, predictions_svm_grid))
```

```
Accuracy : 0.18219
Hamming loss  0.00326606
Micro-average quality numbers
Precision: 0.5817, Recall: 0.3366, F1-measure: 0.4264
Macro-average quality numbers
Precision: 0.3366, Recall: 0.2435, F1-measure: 0.2684
             precision    recall  f1-score   support

          0       0.43      0.24      0.31      7740
          1       0.64      0.45      0.53      7046
          2       0.72      0.56      0.63      6670
          3       0.64      0.42      0.51      6309
          4       0.83      0.76      0.79      5580
          5       0.80      0.65      0.72      5315
          6       0.70      0.18      0.29      3473
          7       0.85      0.64      0.73      3262
          8       0.65      0.39      0.49      3102
          9       0.73      0.41      0.53      2986
         10       0.78      0.68      0.73      2974
         11       0.24      0.04      0.06      2862
         12       0.27      0.01      0.03      2690
         13       0.54      0.25      0.34      2452
         14       0.68      0.17      0.27      2300
         15       0.44      0.30      0.35      2357
         16       0.67      0.46      0.54      2252
         17       0.74      0.60      0.67      2020
         18       0.42      0.35      0.38      1764
         19       0.28      0.06      0.10      1660
         20       0.19      0.12      0.15      1489
         21       0.58      0.44      0.50      1272
         22       0.50      0.35      0.41      1334
         23       0.72      0.69      0.70      1055
         24       0.42      0.43      0.43      1078
         25       0.53      0.48      0.50      1021
         26       0.10      0.02      0.03      1031
         27       0.81      0.71      0.76       999
         28       0.47      0.39      0.42       974
         29       0.56      0.28      0.38       936
         30       0.75      0.00      0.01       839
         31       0.90      0.86      0.88       883
         32       0.54      0.39      0.46       823
         33       0.51      0.30      0.38       808
         34       0.13      0.01      0.01       778
         35       0.60      0.60      0.60       743
         36       0.60      0.58      0.59       727
         37       0.67      0.70      0.68       732
         38       0.34      0.23      0.27       741
         39       0.00      0.00      0.00       666
         40       0.64      0.31      0.42       626
         41       0.00      0.00      0.00       640
         42       0.55      0.28      0.37       634
         43       0.16      0.03      0.06       601
         44       1.00      0.00      0.00       586
         45       0.44      0.44      0.44       599
         46       0.17      0.16      0.16       569
         47       0.33      0.03      0.05       525
         48       0.00      0.00      0.00       560
```

49	0.04	0.03	0.04	550
50	0.52	0.60	0.56	549
51	0.29	0.13	0.18	487
52	0.70	0.82	0.76	514
53	0.58	0.32	0.41	520
54	0.74	0.47	0.58	525
55	0.00	0.00	0.00	524
56	0.00	0.00	0.00	499
57	0.78	0.77	0.77	507
58	0.62	0.61	0.62	460
59	0.37	0.18	0.24	513
60	0.11	0.04	0.06	539
61	0.64	0.59	0.61	452
62	0.81	0.75	0.78	467
63	0.45	0.13	0.20	507
64	0.00	0.00	0.00	506
65	0.61	0.33	0.43	430
66	0.01	0.00	0.00	454
67	0.63	0.57	0.59	430
68	0.43	0.36	0.39	419
69	0.00	0.00	0.00	474
70	0.71	0.66	0.68	407
71	0.43	0.42	0.43	426
72	0.70	0.21	0.32	427
73	0.31	0.19	0.23	431
74	0.47	0.56	0.51	426
75	0.78	0.77	0.78	433
76	0.42	0.34	0.37	429
77	0.57	0.62	0.60	386
78	0.17	0.01	0.03	404
79	0.66	0.39	0.49	395
80	0.00	0.00	0.00	384
81	0.37	0.26	0.30	367
82	0.00	0.00	0.00	398
83	0.30	0.30	0.30	362
84	0.35	0.24	0.29	388
85	0.93	0.55	0.69	352
86	0.77	0.56	0.65	361
87	0.60	0.62	0.61	389
88	0.50	0.61	0.55	340
89	0.86	0.54	0.66	364
90	0.07	0.01	0.01	364
91	0.71	0.64	0.67	355
92	0.70	0.75	0.73	325
93	0.65	0.43	0.51	331
94	0.43	0.52	0.47	324
95	0.72	0.06	0.12	332
96	0.12	0.04	0.06	332
97	0.47	0.08	0.14	333
98	0.87	0.79	0.83	304
99	0.00	0.00	0.00	321
100	0.83	0.70	0.76	306
101	0.70	0.64	0.67	318
102	0.81	0.80	0.80	319
103	0.49	0.22	0.30	307
104	0.43	0.45	0.44	288
105	0.82	0.61	0.70	303

106	0.08	0.03	0.04	327
107	0.70	0.26	0.37	294
108	0.00	0.00	0.00	316
109	0.63	0.64	0.63	274
110	0.00	0.00	0.00	287
111	0.00	0.00	0.00	268
112	0.33	0.37	0.35	272
113	0.00	0.00	0.00	280
114	0.61	0.54	0.57	266
115	0.25	0.33	0.28	298
116	0.32	0.34	0.33	258
117	0.50	0.00	0.01	264
118	0.46	0.29	0.36	275
119	0.00	0.00	0.00	259
120	0.86	0.69	0.76	251
121	0.25	0.25	0.25	294
122	0.63	0.41	0.50	258
123	0.86	0.68	0.76	268
124	0.70	0.51	0.59	255
125	0.72	0.70	0.71	267
126	0.23	0.15	0.18	297
127	0.93	0.87	0.90	245
128	0.00	0.00	0.00	271
129	0.27	0.14	0.18	269
130	0.00	0.00	0.00	235
131	0.00	0.00	0.00	245
132	0.00	0.00	0.00	262
133	0.66	0.08	0.14	265
134	0.25	0.35	0.29	236
135	0.00	0.00	0.00	262
136	0.87	0.87	0.87	259
137	0.00	0.00	0.00	281
138	0.48	0.57	0.52	253
139	0.61	0.58	0.59	268
140	0.56	0.60	0.58	277
141	0.23	0.35	0.28	235
142	0.34	0.20	0.25	255
143	0.84	0.83	0.83	253
144	0.00	0.00	0.00	255
145	0.00	0.00	0.00	243
146	0.39	0.44	0.42	232
147	0.00	0.00	0.00	232
148	0.00	0.00	0.00	234
149	0.31	0.36	0.33	231
150	0.89	0.82	0.85	229
151	0.14	0.09	0.11	226
152	0.74	0.50	0.60	257
153	0.00	0.00	0.00	223
154	0.09	0.11	0.10	238
155	0.69	0.16	0.26	234
156	0.23	0.08	0.11	222
157	0.31	0.08	0.12	205
158	0.00	0.00	0.00	244
159	0.46	0.08	0.13	239
160	0.21	0.06	0.09	230
161	0.46	0.34	0.39	223
162	0.57	0.56	0.57	238

163	0.70	0.68	0.69	211
164	0.47	0.44	0.46	208
165	1.00	0.00	0.01	233
166	0.00	0.00	0.00	227
167	0.35	0.31	0.33	217
168	0.00	0.00	0.00	208
169	0.26	0.16	0.20	245
170	0.18	0.24	0.21	245
171	0.27	0.26	0.27	208
172	0.00	0.00	0.00	210
173	0.50	0.04	0.08	215
174	0.47	0.18	0.26	203
175	0.00	0.00	0.00	196
176	0.83	0.73	0.78	214
177	0.45	0.36	0.40	224
178	0.78	0.81	0.79	212
179	0.00	0.00	0.00	200
180	0.00	0.00	0.00	220
181	0.00	0.00	0.00	188
182	0.48	0.29	0.36	217
183	0.58	0.57	0.57	206
184	0.57	0.32	0.41	215
185	0.11	0.13	0.12	181
186	0.30	0.26	0.27	192
187	0.60	0.56	0.58	191
188	0.87	0.72	0.79	211
189	0.00	0.00	0.00	192
190	0.70	0.66	0.68	194
191	0.89	0.85	0.87	191
192	1.00	0.01	0.01	195
193	0.30	0.15	0.20	197
194	0.49	0.47	0.48	174
195	0.00	0.00	0.00	199
196	0.00	0.00	0.00	197
197	0.81	0.59	0.68	192
198	0.83	0.53	0.65	204
199	0.00	0.00	0.00	203
200	0.00	0.00	0.00	184
201	0.97	0.63	0.76	181
202	0.36	0.30	0.33	183
203	0.41	0.16	0.23	202
204	0.00	0.00	0.00	172
205	0.74	0.42	0.54	183
206	0.00	0.00	0.00	181
207	0.51	0.43	0.47	201
208	0.00	0.00	0.00	197
209	0.54	0.23	0.32	181
210	0.25	0.26	0.26	185
211	0.00	0.00	0.00	185
212	0.47	0.37	0.41	174
213	0.74	0.26	0.39	185
214	0.00	0.00	0.00	189
215	0.60	0.54	0.57	173
216	0.00	0.00	0.00	175
217	0.00	0.00	0.00	168
218	0.34	0.38	0.36	172
219	0.34	0.34	0.34	184

220	0.00	0.00	0.00	162
221	0.00	0.00	0.00	159
222	0.00	0.00	0.00	158
223	0.79	0.77	0.78	176
224	0.00	0.00	0.00	182
225	0.51	0.49	0.50	166
226	0.00	0.00	0.00	195
227	0.91	0.73	0.81	184
228	0.62	0.56	0.59	177
229	0.36	0.18	0.24	190
230	0.13	0.13	0.13	186
231	0.94	0.72	0.82	170
232	0.00	0.00	0.00	180
233	0.31	0.37	0.34	150
234	0.26	0.12	0.17	160
235	0.71	0.65	0.67	172
236	0.00	0.00	0.00	158
237	0.65	0.67	0.66	169
238	0.78	0.50	0.61	160
239	0.35	0.21	0.27	169
240	0.00	0.00	0.00	156
241	0.10	0.13	0.11	172
242	0.00	0.00	0.00	157
243	0.57	0.40	0.47	169
244	0.68	0.67	0.68	158
245	0.00	0.00	0.00	169
246	0.16	0.13	0.14	151
247	0.31	0.24	0.27	163
248	0.44	0.50	0.47	138
249	0.00	0.00	0.00	149
250	0.37	0.23	0.28	159
251	0.69	0.25	0.37	160
252	0.00	0.00	0.00	158
253	0.37	0.20	0.26	169
254	0.29	0.18	0.22	151
255	0.43	0.25	0.31	155
256	0.00	0.00	0.00	154
257	0.00	0.00	0.00	148
258	0.39	0.55	0.46	137
259	0.10	0.10	0.10	159
260	0.59	0.34	0.43	130
261	0.80	0.49	0.60	154
262	0.65	0.62	0.63	144
263	0.05	0.05	0.05	151
264	0.83	0.61	0.70	148
265	0.32	0.40	0.36	141
266	0.34	0.38	0.36	152
267	0.00	0.00	0.00	152
268	0.29	0.40	0.34	122
269	0.00	0.00	0.00	143
270	0.60	0.56	0.58	126
271	0.37	0.18	0.24	159
272	0.51	0.53	0.52	120
273	0.00	0.00	0.00	137
274	0.00	0.00	0.00	150
275	0.51	0.39	0.44	136
276	0.44	0.41	0.42	153

277	0.54	0.37	0.44	142
278	0.00	0.00	0.00	150
279	0.56	0.52	0.54	122
280	0.83	0.83	0.83	134
281	0.54	0.35	0.43	141
282	0.00	0.00	0.00	153
283	0.91	0.67	0.78	159
284	0.29	0.13	0.18	141
285	0.00	0.00	0.00	115
286	0.00	0.00	0.00	129
287	0.03	0.01	0.01	141
288	0.88	0.74	0.80	142
289	0.32	0.15	0.20	143
290	0.27	0.34	0.30	118
291	0.30	0.11	0.16	134
292	0.00	0.00	0.00	124
293	0.00	0.00	0.00	113
294	0.54	0.70	0.61	125
295	0.38	0.11	0.17	137
296	0.25	0.13	0.17	116
297	0.76	0.55	0.64	148
298	0.49	0.27	0.35	131
299	0.37	0.25	0.30	139
300	0.00	0.00	0.00	126
301	0.00	0.00	0.00	141
302	0.00	0.00	0.00	131
303	0.00	0.00	0.00	123
304	0.00	0.00	0.00	133
305	0.25	0.15	0.19	115
306	0.00	0.00	0.00	114
307	0.00	0.00	0.00	130
308	0.44	0.10	0.16	125
309	0.26	0.09	0.14	131
310	0.51	0.32	0.39	128
311	0.00	0.00	0.00	121
312	0.07	0.01	0.01	140
313	0.00	0.00	0.00	107
314	0.36	0.35	0.35	117
315	0.00	0.00	0.00	119
316	0.38	0.16	0.22	119
317	0.00	0.00	0.00	120
318	0.16	0.11	0.13	116
319	0.72	0.49	0.58	133
320	0.00	0.00	0.00	122
321	0.68	0.12	0.21	124
322	0.20	0.12	0.15	120
323	0.00	0.00	0.00	108
324	0.00	0.00	0.00	117
325	0.00	0.00	0.00	98
326	0.33	0.01	0.02	106
327	0.00	0.00	0.00	135
328	0.35	0.23	0.28	127
329	0.26	0.19	0.22	121
330	0.67	0.14	0.23	113
331	0.00	0.00	0.00	135
332	0.00	0.00	0.00	116
333	0.00	0.00	0.00	101

334	0.39	0.28	0.33	118
335	0.00	0.00	0.00	124
336	0.00	0.00	0.00	109
337	0.00	0.00	0.00	113
338	0.00	0.00	0.00	118
339	0.00	0.00	0.00	105
340	0.33	0.42	0.37	103
341	0.89	0.06	0.12	124
342	0.49	0.38	0.43	115
343	0.00	0.00	0.00	111
344	0.00	0.00	0.00	118
345	0.00	0.00	0.00	118
346	0.35	0.35	0.35	105
347	0.00	0.00	0.00	106
348	0.00	0.00	0.00	117
349	0.00	0.00	0.00	102
350	0.17	0.12	0.14	119
351	0.41	0.47	0.44	113
352	0.00	0.00	0.00	95
353	0.00	0.00	0.00	110
354	0.00	0.00	0.00	116
355	0.67	0.07	0.13	114
356	0.87	0.35	0.50	116
357	0.35	0.20	0.25	122
358	0.65	0.50	0.56	131
359	0.92	0.47	0.62	105
360	0.14	0.03	0.05	108
361	0.00	0.00	0.00	110
362	0.85	0.34	0.48	115
363	0.00	0.00	0.00	105
364	0.07	0.10	0.09	107
365	0.38	0.33	0.35	107
366	0.00	0.00	0.00	105
367	0.37	0.30	0.33	113
368	0.46	0.32	0.37	101
369	0.00	0.00	0.00	102
370	0.94	0.68	0.79	98
371	0.48	0.37	0.42	101
372	0.00	0.00	0.00	104
373	0.24	0.18	0.21	114
374	0.00	0.00	0.00	104
375	0.00	0.00	0.00	92
376	0.00	0.00	0.00	105
377	0.00	0.00	0.00	109
378	0.00	0.00	0.00	105
379	0.08	0.11	0.09	98
380	0.00	0.00	0.00	109
381	0.00	0.00	0.00	93
382	0.00	0.00	0.00	91
383	0.00	0.00	0.00	100
384	0.95	0.76	0.85	108
385	0.21	0.23	0.22	87
386	0.60	0.16	0.25	95
387	0.59	0.45	0.51	98
388	0.30	0.10	0.15	94
389	0.00	0.00	0.00	117
390	0.87	0.28	0.43	92

391	0.08	0.07	0.08	94
392	0.08	0.05	0.06	92
393	0.05	0.03	0.04	93
394	0.26	0.40	0.31	90
395	0.65	0.72	0.69	120
396	0.53	0.46	0.49	83
397	0.00	0.00	0.00	82
398	0.64	0.38	0.48	97
399	0.68	0.33	0.45	96
400	0.00	0.00	0.00	95
401	0.52	0.27	0.36	96
402	0.25	0.22	0.23	95
403	0.46	0.19	0.27	91
404	0.00	0.00	0.00	96
405	0.00	0.00	0.00	92
406	0.00	0.00	0.00	97
407	0.00	0.00	0.00	97
408	0.32	0.27	0.29	93
409	0.76	0.52	0.62	81
410	0.00	0.00	0.00	104
411	0.14	0.11	0.12	91
412	0.30	0.20	0.24	101
413	0.00	0.00	0.00	98
414	0.00	0.00	0.00	94
415	0.37	0.24	0.29	94
416	0.00	0.00	0.00	92
417	0.00	0.00	0.00	81
418	0.00	0.00	0.00	100
419	0.89	0.56	0.69	84
420	0.00	0.00	0.00	103
421	0.00	0.00	0.00	94
422	0.45	0.48	0.46	95
423	0.14	0.12	0.13	97
424	0.06	0.17	0.09	87
425	0.00	0.00	0.00	94
426	0.56	0.43	0.49	92
427	0.64	0.76	0.69	89
428	0.38	0.34	0.36	93
429	0.33	0.03	0.05	78
430	0.81	0.47	0.59	94
431	0.00	0.00	0.00	94
432	0.90	0.41	0.56	91
433	0.00	0.00	0.00	99
434	0.00	0.00	0.00	83
435	0.00	0.00	0.00	92
436	0.00	0.00	0.00	79
437	0.00	0.00	0.00	99
438	0.56	0.63	0.60	84
439	0.71	0.60	0.65	84
440	0.20	0.21	0.20	92
441	0.00	0.00	0.00	84
442	0.00	0.00	0.00	90
443	0.11	0.12	0.11	82
444	0.60	0.48	0.53	90
445	0.27	0.28	0.28	85
446	0.35	0.22	0.27	103
447	0.76	0.30	0.43	87

448	0.00	0.00	0.00	80
449	0.00	0.00	0.00	88
450	0.00	0.00	0.00	89
451	0.76	0.77	0.77	100
452	0.00	0.00	0.00	81
453	0.00	0.00	0.00	93
454	0.00	0.00	0.00	90
455	0.11	0.04	0.05	111
456	0.36	0.22	0.28	76
457	0.52	0.57	0.54	92
458	0.00	0.00	0.00	90
459	0.00	0.00	0.00	81
460	0.50	0.21	0.30	70
461	0.57	0.40	0.47	62
462	0.00	0.00	0.00	89
463	0.00	0.00	0.00	103
464	0.93	0.50	0.65	82
465	0.36	0.19	0.25	96
466	0.00	0.00	0.00	93
467	0.00	0.00	0.00	84
468	0.39	0.26	0.31	70
469	0.00	0.00	0.00	95
470	0.00	0.00	0.00	87
471	0.00	0.00	0.00	77
472	0.69	0.37	0.49	91
473	0.34	0.25	0.29	91
474	0.68	0.79	0.73	89
475	0.08	0.12	0.10	85
476	0.93	0.68	0.79	76
477	0.35	0.36	0.36	83
478	0.52	0.52	0.52	82
479	0.20	0.16	0.18	91
480	0.80	0.37	0.50	87
481	0.83	0.50	0.62	90
482	0.00	0.00	0.00	92
483	0.00	0.00	0.00	80
484	0.00	0.00	0.00	84
485	0.14	0.14	0.14	92
486	0.00	0.00	0.00	92
487	0.92	0.42	0.57	79
488	0.00	0.00	0.00	84
489	0.77	0.64	0.70	76
490	0.00	0.00	0.00	67
491	0.24	0.23	0.23	74
492	0.27	0.35	0.30	100
493	0.44	0.41	0.42	76
494	0.66	0.52	0.58	83
495	0.00	0.00	0.00	76
496	0.46	0.42	0.44	76
497	0.00	0.00	0.00	82
498	0.00	0.00	0.00	81
499	0.17	0.03	0.05	69
avg / total	0.48	0.34	0.38	180360

Featurizing the data using TFIDF

In [14]:
```
vectorizer_tfidf = TfidfVectorizer(min_df=0.00009, max_features=200000, smooth
_idf=True, norm="l2",tokenizer = lambda x: x.split(), sublinear_tf=False, ngra
m_range=(1,3))
x_train_multilabel = vectorizer_tfidf.fit_transform(x_train['question'])
x_test_multilabel = vectorizer_tfidf.transform(x_test['question'])
```

Logistic Regression with One Vs Rest Classifier

In [16]:
```
start = datetime.now()
classifier2_LR = OneVsRestClassifier(LogisticRegression(penalty='l1'), n_jobs=
-1)
classifier2_LR.fit(x_train_multilabel, y_train)
pickle.dump(classifier2_LR,open('classifier2_LR_tfidf.sav','wb'))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Time taken to run this cell : 0:32:09.349928
```

In [8]:
```
predictions2_LR = classifier2_LR.predict(x_test_multilabel)
print("Accuracy :",accuracy_score(y_test, predictions2_LR))
print("Hamming loss ",hamming_loss(y_test,predictions2_LR))


precision = precision_score(y_test, predictions2_LR, average='micro')
recall = recall_score(y_test, predictions2_LR, average='micro')
f1 = f1_score(y_test, predictions2_LR, average='micro')

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

precision = precision_score(y_test, predictions2_LR, average='macro')
recall = recall_score(y_test, predictions2_LR, average='macro')
f1 = f1_score(y_test, predictions2_LR, average='macro')

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

print (classification_report(y_test, predictions2_LR))
```

```
Accuracy : 0.24945
Hamming loss  0.00274646
Micro-average quality numbers
Precision: 0.7170, Recall: 0.3943, F1-measure: 0.5088
Macro-average quality numbers
Precision: 0.5638, Recall: 0.3161, F1-measure: 0.3897
             precision    recall  f1-score   support

          0       0.65      0.33      0.43      7740
          1       0.80      0.50      0.62      7046
          2       0.85      0.59      0.69      6670
          3       0.76      0.45      0.57      6309
          4       0.95      0.81      0.87      5580
          5       0.87      0.65      0.75      5315
          6       0.72      0.40      0.51      3473
          7       0.89      0.66      0.76      3262
          8       0.69      0.47      0.56      3102
          9       0.80      0.44      0.57      2986
         10       0.86      0.64      0.73      2974
         11       0.58      0.24      0.34      2862
         12       0.59      0.14      0.23      2690
         13       0.60      0.32      0.41      2452
         14       0.59      0.26      0.36      2300
         15       0.61      0.35      0.45      2357
         16       0.80      0.57      0.67      2252
         17       0.80      0.61      0.69      2020
         18       0.63      0.30      0.40      1764
         19       0.60      0.25      0.35      1660
         20       0.41      0.10      0.17      1489
         21       0.74      0.43      0.55      1272
         22       0.60      0.34      0.43      1334
         23       0.89      0.65      0.75      1055
         24       0.67      0.47      0.55      1078
         25       0.68      0.45      0.54      1021
         26       0.41      0.10      0.17      1031
         27       0.87      0.71      0.78       999
         28       0.61      0.38      0.47       974
         29       0.70      0.26      0.38       936
         30       0.55      0.31      0.40       839
         31       0.94      0.80      0.86       883
         32       0.81      0.32      0.46       823
         33       0.64      0.39      0.49       808
         34       0.51      0.20      0.29       778
         35       0.76      0.58      0.66       743
         36       0.78      0.57      0.66       727
         37       0.78      0.69      0.73       732
         38       0.39      0.18      0.25       741
         39       0.48      0.17      0.25       666
         40       0.71      0.33      0.45       626
         41       0.41      0.14      0.21       640
         42       0.65      0.42      0.51       634
         43       0.38      0.13      0.19       601
         44       0.43      0.19      0.26       586
         45       0.60      0.32      0.42       599
         46       0.28      0.10      0.15       569
         47       0.54      0.20      0.29       525
         48       0.66      0.19      0.30       560
```

49	0.50	0.03	0.06	550
50	0.70	0.49	0.58	549
51	0.50	0.21	0.30	487
52	0.88	0.77	0.82	514
53	0.62	0.39	0.48	520
54	0.79	0.48	0.60	525
55	0.59	0.24	0.34	524
56	0.38	0.13	0.19	499
57	0.93	0.80	0.86	507
58	0.75	0.54	0.63	460
59	0.44	0.15	0.22	513
60	0.22	0.03	0.05	539
61	0.79	0.56	0.66	452
62	0.94	0.71	0.81	467
63	0.90	0.35	0.50	507
64	0.28	0.06	0.10	506
65	0.72	0.32	0.44	430
66	0.41	0.03	0.06	454
67	0.77	0.58	0.66	430
68	0.79	0.57	0.66	419
69	0.42	0.16	0.23	474
70	0.78	0.56	0.65	407
71	0.62	0.39	0.48	426
72	0.77	0.31	0.44	427
73	0.59	0.23	0.33	431
74	0.56	0.41	0.47	426
75	0.89	0.69	0.78	433
76	0.57	0.36	0.44	429
77	0.70	0.54	0.61	386
78	0.20	0.02	0.03	404
79	0.73	0.43	0.54	395
80	0.38	0.11	0.17	384
81	0.59	0.38	0.46	367
82	0.47	0.20	0.28	398
83	0.46	0.22	0.29	362
84	0.54	0.22	0.31	388
85	0.96	0.61	0.74	352
86	0.80	0.54	0.64	361
87	0.77	0.51	0.61	389
88	0.73	0.55	0.62	340
89	0.91	0.67	0.77	364
90	0.53	0.12	0.19	364
91	0.81	0.59	0.69	355
92	0.84	0.67	0.75	325
93	0.64	0.46	0.54	331
94	0.66	0.51	0.57	324
95	0.67	0.19	0.29	332
96	0.47	0.17	0.25	332
97	0.55	0.27	0.36	333
98	0.94	0.73	0.82	304
99	0.30	0.07	0.11	321
100	0.90	0.74	0.82	306
101	0.88	0.74	0.80	318
102	0.92	0.73	0.81	319
103	0.66	0.22	0.33	307
104	0.69	0.46	0.55	288
105	0.89	0.67	0.77	303

```
        106       0.22      0.03      0.05       327
        107       0.72      0.40      0.52       294
        108       0.26      0.03      0.06       316
        109       0.83      0.54      0.65       274
        110       0.31      0.10      0.15       287
        111       0.44      0.22      0.30       268
        112       0.44      0.26      0.33       272
        113       0.64      0.23      0.34       280
        114       0.64      0.48      0.55       266
        115       0.61      0.34      0.44       298
        116       0.64      0.34      0.44       258
        117       0.53      0.16      0.24       264
        118       0.56      0.23      0.33       275
        119       0.44      0.18      0.25       259
        120       0.92      0.75      0.83       251
        121       0.40      0.15      0.22       294
        122       0.65      0.45      0.54       258
        123       0.93      0.74      0.82       268
        124       0.70      0.47      0.57       255
        125       0.85      0.64      0.73       267
        126       0.34      0.12      0.17       297
        127       0.95      0.89      0.92       245
        128       0.24      0.08      0.12       271
        129       0.39      0.16      0.23       269
        130       0.29      0.09      0.13       235
        131       0.34      0.05      0.09       245
        132       0.00      0.00      0.00       262
        133       0.65      0.25      0.36       265
        134       0.41      0.26      0.32       236
        135       0.17      0.02      0.04       262
        136       0.94      0.78      0.85       259
        137       0.54      0.14      0.22       281
        138       0.77      0.59      0.67       253
        139       0.66      0.43      0.52       268
        140       0.78      0.61      0.68       277
        141       0.65      0.46      0.53       235
        142       0.56      0.33      0.41       255
        143       0.91      0.81      0.86       253
        144       0.31      0.07      0.11       255
        145       0.29      0.10      0.15       243
        146       0.64      0.48      0.55       232
        147       0.42      0.06      0.11       232
        148       0.28      0.08      0.13       234
        149       0.53      0.34      0.42       231
        150       0.94      0.86      0.89       229
        151       0.58      0.16      0.26       226
        152       0.81      0.53      0.64       257
        153       0.48      0.19      0.27       223
        154       0.32      0.10      0.15       238
        155       0.49      0.27      0.35       234
        156       0.52      0.11      0.19       222
        157       0.47      0.18      0.26       205
        158       0.37      0.13      0.19       244
        159       0.50      0.26      0.34       239
        160       0.45      0.13      0.20       230
        161       0.78      0.51      0.62       223
        162       0.66      0.53      0.59       238
```

163	0.91	0.78	0.84	211
164	0.50	0.37	0.42	208
165	0.50	0.21	0.30	233
166	0.37	0.14	0.20	227
167	0.54	0.35	0.43	217
168	0.56	0.36	0.44	208
169	0.63	0.20	0.30	245
170	0.54	0.31	0.39	245
171	0.30	0.11	0.16	208
172	0.36	0.04	0.07	210
173	0.43	0.23	0.30	215
174	0.56	0.20	0.30	203
175	0.38	0.19	0.25	196
176	0.92	0.74	0.82	214
177	0.78	0.50	0.61	224
178	0.84	0.74	0.79	212
179	0.18	0.01	0.03	200
180	0.28	0.04	0.07	220
181	0.25	0.04	0.06	188
182	0.77	0.45	0.57	217
183	0.74	0.50	0.59	206
184	0.77	0.52	0.62	215
185	0.30	0.09	0.14	181
186	0.43	0.21	0.28	192
187	0.80	0.60	0.69	191
188	0.91	0.69	0.79	211
189	0.52	0.15	0.23	192
190	0.73	0.56	0.63	194
191	0.96	0.83	0.89	191
192	0.71	0.33	0.45	195
193	0.43	0.20	0.28	197
194	0.69	0.42	0.52	174
195	0.46	0.19	0.27	199
196	0.58	0.18	0.27	197
197	0.86	0.65	0.74	192
198	0.88	0.71	0.78	204
199	0.21	0.03	0.06	203
200	0.34	0.08	0.12	184
201	0.93	0.72	0.81	181
202	0.69	0.38	0.49	183
203	0.56	0.26	0.36	202
204	0.43	0.09	0.15	172
205	0.82	0.49	0.61	183
206	0.14	0.04	0.07	181
207	0.77	0.42	0.54	201
208	0.39	0.19	0.26	197
209	0.65	0.31	0.42	181
210	0.76	0.44	0.56	185
211	0.38	0.12	0.19	185
212	0.67	0.38	0.49	174
213	0.72	0.48	0.57	185
214	0.37	0.10	0.16	189
215	0.71	0.40	0.51	173
216	0.00	0.00	0.00	175
217	0.38	0.10	0.15	168
218	0.82	0.57	0.67	172
219	0.56	0.35	0.43	184

220	0.24	0.03	0.05	162
221	0.57	0.22	0.32	159
222	0.23	0.04	0.07	158
223	0.93	0.81	0.87	176
224	0.48	0.15	0.23	182
225	0.70	0.50	0.58	166
226	0.43	0.12	0.19	195
227	0.97	0.77	0.86	184
228	0.69	0.48	0.56	177
229	0.57	0.27	0.36	190
230	0.39	0.13	0.20	186
231	0.96	0.69	0.81	170
232	0.44	0.19	0.27	180
233	0.57	0.39	0.46	150
234	0.38	0.18	0.24	160
235	0.81	0.59	0.68	172
236	0.29	0.04	0.07	158
237	0.78	0.60	0.68	169
238	0.81	0.59	0.68	160
239	0.46	0.19	0.27	169
240	0.63	0.25	0.36	156
241	0.53	0.28	0.37	172
242	0.34	0.14	0.20	157
243	0.66	0.49	0.56	169
244	0.86	0.61	0.71	158
245	0.35	0.08	0.13	169
246	0.38	0.10	0.16	151
247	0.69	0.34	0.46	163
248	0.67	0.45	0.54	138
249	0.52	0.21	0.30	149
250	0.52	0.30	0.38	159
251	0.75	0.42	0.54	160
252	0.70	0.04	0.08	158
253	0.59	0.22	0.32	169
254	0.63	0.38	0.47	151
255	0.52	0.28	0.37	155
256	0.07	0.01	0.01	154
257	0.36	0.03	0.06	148
258	0.72	0.55	0.63	137
259	0.39	0.16	0.22	159
260	0.79	0.41	0.54	130
261	0.90	0.60	0.72	154
262	0.87	0.70	0.78	144
263	0.17	0.02	0.04	151
264	0.91	0.68	0.78	148
265	0.70	0.43	0.54	141
266	0.58	0.42	0.49	152
267	0.35	0.04	0.07	152
268	0.48	0.34	0.39	122
269	0.37	0.07	0.12	143
270	0.83	0.62	0.71	126
271	0.56	0.21	0.30	159
272	0.76	0.49	0.60	120
273	0.00	0.00	0.00	137
274	0.47	0.14	0.22	150
275	0.68	0.56	0.62	136
276	0.68	0.33	0.44	153

277	0.70	0.32	0.44	142
278	0.65	0.21	0.32	150
279	0.77	0.61	0.68	122
280	0.93	0.86	0.89	134
281	0.71	0.47	0.56	141
282	0.25	0.01	0.01	153
283	0.99	0.83	0.90	159
284	0.56	0.30	0.39	141
285	0.27	0.09	0.13	115
286	0.42	0.10	0.16	129
287	0.32	0.09	0.13	141
288	0.89	0.74	0.81	142
289	0.51	0.24	0.33	143
290	0.53	0.27	0.36	118
291	0.59	0.18	0.27	134
292	0.43	0.10	0.17	124
293	0.00	0.00	0.00	113
294	0.83	0.67	0.74	125
295	0.51	0.33	0.40	137
296	0.23	0.09	0.13	116
297	0.79	0.56	0.66	148
298	0.64	0.34	0.44	131
299	0.72	0.44	0.54	139
300	0.22	0.04	0.07	126
301	0.45	0.13	0.20	141
302	0.16	0.02	0.04	131
303	0.29	0.11	0.16	123
304	0.31	0.14	0.19	133
305	0.35	0.20	0.25	115
306	0.64	0.31	0.41	114
307	0.29	0.08	0.12	130
308	0.50	0.23	0.32	125
309	0.56	0.24	0.33	131
310	0.67	0.44	0.53	128
311	0.22	0.03	0.06	121
312	0.36	0.03	0.05	140
313	0.22	0.07	0.10	107
314	0.68	0.43	0.52	117
315	0.00	0.00	0.00	119
316	0.58	0.43	0.49	119
317	0.20	0.03	0.04	120
318	0.79	0.53	0.63	116
319	0.80	0.58	0.67	133
320	0.65	0.36	0.46	122
321	0.72	0.45	0.55	124
322	0.34	0.12	0.18	120
323	0.52	0.23	0.32	108
324	0.38	0.05	0.09	117
325	0.00	0.00	0.00	98
326	0.13	0.04	0.06	106
327	0.09	0.01	0.01	135
328	0.42	0.22	0.29	127
329	0.57	0.31	0.40	121
330	0.68	0.42	0.52	113
331	0.45	0.14	0.21	135
332	0.35	0.08	0.13	116
333	0.55	0.18	0.27	101

334	0.57	0.36	0.44	118
335	0.49	0.19	0.27	124
336	0.33	0.01	0.02	109
337	0.67	0.29	0.41	113
338	0.26	0.05	0.09	118
339	0.27	0.07	0.11	105
340	0.63	0.40	0.49	103
341	0.66	0.27	0.38	124
342	0.62	0.35	0.44	115
343	0.52	0.24	0.33	111
344	0.31	0.04	0.07	118
345	0.17	0.03	0.06	118
346	0.49	0.31	0.38	105
347	0.29	0.08	0.12	106
348	0.72	0.33	0.46	117
349	0.15	0.02	0.03	102
350	0.49	0.29	0.37	119
351	0.68	0.42	0.52	113
352	0.44	0.18	0.25	95
353	0.37	0.15	0.22	110
354	0.22	0.07	0.10	116
355	0.62	0.35	0.45	114
356	0.88	0.64	0.74	116
357	0.61	0.33	0.43	122
358	0.82	0.51	0.63	131
359	0.86	0.57	0.69	105
360	0.50	0.12	0.19	108
361	0.49	0.16	0.24	110
362	0.84	0.66	0.74	115
363	0.35	0.06	0.10	105
364	0.38	0.14	0.21	107
365	0.56	0.32	0.40	107
366	0.08	0.01	0.02	105
367	0.43	0.23	0.30	113
368	0.66	0.38	0.48	101
369	0.41	0.17	0.24	102
370	0.94	0.87	0.90	98
371	0.53	0.36	0.43	101
372	0.10	0.02	0.03	104
373	0.38	0.14	0.21	114
374	0.27	0.03	0.05	104
375	0.11	0.04	0.06	92
376	0.42	0.05	0.09	105
377	0.33	0.06	0.11	109
378	0.82	0.43	0.56	105
379	0.05	0.01	0.02	98
380	0.84	0.56	0.67	109
381	0.09	0.02	0.03	93
382	0.26	0.10	0.14	91
383	0.29	0.11	0.16	100
384	0.95	0.94	0.94	108
385	0.30	0.15	0.20	87
386	0.60	0.27	0.38	95
387	0.78	0.55	0.65	98
388	0.54	0.16	0.25	94
389	0.72	0.22	0.34	117
390	0.93	0.76	0.84	92

391	0.13	0.03	0.05	94
392	0.12	0.03	0.05	92
393	0.75	0.06	0.12	93
394	0.72	0.56	0.63	90
395	0.78	0.57	0.66	120
396	0.77	0.57	0.65	83
397	0.18	0.02	0.04	82
398	0.75	0.53	0.62	97
399	0.82	0.55	0.66	96
400	0.45	0.11	0.17	95
401	0.96	0.53	0.68	96
402	0.46	0.22	0.30	95
403	0.52	0.35	0.42	91
404	0.11	0.03	0.05	96
405	0.14	0.03	0.05	92
406	0.58	0.29	0.39	97
407	0.20	0.05	0.08	97
408	0.64	0.32	0.43	93
409	0.90	0.68	0.77	81
410	0.52	0.12	0.19	104
411	0.24	0.12	0.16	91
412	0.44	0.23	0.30	101
413	1.00	0.02	0.04	98
414	0.70	0.37	0.49	94
415	0.65	0.26	0.37	94
416	0.44	0.08	0.13	92
417	0.64	0.09	0.15	81
418	0.56	0.18	0.27	100
419	0.94	0.58	0.72	84
420	0.20	0.03	0.05	103
421	0.50	0.01	0.02	94
422	0.93	0.54	0.68	95
423	0.33	0.13	0.19	97
424	0.65	0.36	0.46	87
425	0.00	0.00	0.00	94
426	0.78	0.57	0.65	92
427	0.66	0.53	0.59	89
428	0.62	0.39	0.48	93
429	0.42	0.21	0.28	78
430	0.89	0.54	0.68	94
431	0.75	0.55	0.64	94
432	0.95	0.63	0.75	91
433	0.28	0.08	0.12	99
434	0.50	0.18	0.27	83
435	0.33	0.17	0.23	92
436	0.47	0.10	0.17	79
437	0.00	0.00	0.00	99
438	0.90	0.71	0.79	84
439	0.81	0.73	0.77	84
440	0.43	0.22	0.29	92
441	0.58	0.25	0.35	84
442	0.18	0.06	0.08	90
443	0.29	0.12	0.17	82
444	0.80	0.62	0.70	90
445	0.56	0.28	0.38	85
446	0.69	0.43	0.53	103
447	0.85	0.57	0.68	87

```
        448       0.08      0.01      0.02        80
        449       0.50      0.15      0.23        88
        450       0.33      0.01      0.02        89
        451       0.82      0.80      0.81       100
        452       0.38      0.16      0.23        81
        453       0.48      0.17      0.25        93
        454       0.37      0.12      0.18        90
        455       0.27      0.03      0.05       111
        456       0.50      0.37      0.42        76
        457       0.64      0.49      0.56        92
        458       0.33      0.02      0.04        90
        459       0.50      0.26      0.34        81
        460       0.65      0.29      0.40        70
        461       0.56      0.53      0.55        62
        462       0.29      0.04      0.08        89
        463       0.50      0.14      0.21       103
        464       0.93      0.76      0.83        82
        465       0.55      0.22      0.31        96
        466       0.51      0.29      0.37        93
        467       1.00      0.02      0.05        84
        468       0.51      0.36      0.42        70
        469       0.52      0.36      0.42        95
        470       0.44      0.09      0.15        87
        471       0.33      0.10      0.16        77
        472       0.91      0.70      0.80        91
        473       0.52      0.26      0.35        91
        474       0.80      0.73      0.76        89
        475       0.47      0.19      0.27        85
        476       0.97      0.76      0.85        76
        477       0.45      0.31      0.37        83
        478       0.72      0.44      0.55        82
        479       0.39      0.21      0.27        91
        480       0.74      0.56      0.64        87
        481       0.87      0.69      0.77        90
        482       0.40      0.11      0.17        92
        483       0.42      0.12      0.19        80
        484       0.62      0.15      0.25        84
        485       0.48      0.14      0.22        92
        486       0.65      0.22      0.33        92
        487       0.87      0.52      0.65        79
        488       0.56      0.06      0.11        84
        489       0.86      0.71      0.78        76
        490       0.54      0.22      0.32        67
        491       0.29      0.20      0.24        74
        492       0.47      0.22      0.30       100
        493       0.55      0.36      0.43        76
        494       0.88      0.52      0.65        83
        495       0.52      0.22      0.31        76
        496       0.57      0.42      0.48        76
        497       0.00      0.00      0.00        82
        498       0.00      0.00      0.00        81
        499       0.75      0.39      0.51        69

avg / total       0.66      0.39      0.48    180360
```

SGD Classifier with log loss

In [10]:
```
start = datetime.now()
classifier_log = OneVsRestClassifier(SGDClassifier(loss='log', alpha=0.00001,
penalty='l1'), n_jobs=-1)
classifier_log.fit(x_train_multilabel, y_train)
predictions = classifier_log.predict (x_test_multilabel)


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

```
Accuracy : 0.23451
Hamming loss  0.00281596
Micro-average quality numbers
Precision: 0.7264, Recall: 0.3519, F1-measure: 0.4741
Macro-average quality numbers
Precision: 0.5486, Recall: 0.2724, F1-measure: 0.3478
             precision    recall  f1-score   support

          0       0.63      0.23      0.34      7740
          1       0.81      0.45      0.58      7046
          2       0.84      0.55      0.67      6670
          3       0.78      0.42      0.54      6309
          4       0.95      0.76      0.84      5580
          5       0.87      0.63      0.73      5315
          6       0.70      0.34      0.46      3473
          7       0.91      0.62      0.73      3262
          8       0.72      0.41      0.52      3102
          9       0.81      0.40      0.54      2986
         10       0.87      0.59      0.71      2974
         11       0.56      0.20      0.29      2862
         12       0.57      0.11      0.19      2690
         13       0.61      0.27      0.37      2452
         14       0.60      0.23      0.33      2300
         15       0.61      0.32      0.42      2357
         16       0.79      0.54      0.64      2252
         17       0.80      0.58      0.67      2020
         18       0.66      0.28      0.40      1764
         19       0.56      0.18      0.28      1660
         20       0.36      0.08      0.13      1489
         21       0.75      0.41      0.53      1272
         22       0.62      0.31      0.42      1334
         23       0.90      0.60      0.72      1055
         24       0.67      0.44      0.53      1078
         25       0.67      0.42      0.52      1021
         26       0.36      0.08      0.13      1031
         27       0.87      0.70      0.78       999
         28       0.59      0.35      0.44       974
         29       0.73      0.25      0.37       936
         30       0.55      0.28      0.37       839
         31       0.94      0.77      0.85       883
         32       0.83      0.30      0.44       823
         33       0.65      0.33      0.44       808
         34       0.46      0.13      0.20       778
         35       0.76      0.57      0.65       743
         36       0.78      0.55      0.65       727
         37       0.78      0.67      0.72       732
         38       0.40      0.16      0.22       741
         39       0.42      0.14      0.20       666
         40       0.71      0.28      0.41       626
         41       0.41      0.12      0.18       640
         42       0.66      0.39      0.49       634
         43       0.39      0.11      0.18       601
         44       0.39      0.13      0.19       586
         45       0.61      0.30      0.41       599
         46       0.24      0.07      0.11       569
         47       0.52      0.17      0.26       525
         48       0.61      0.12      0.19       560
```

49	0.58	0.01	0.02	550
50	0.70	0.47	0.57	549
51	0.50	0.18	0.27	487
52	0.87	0.76	0.81	514
53	0.62	0.37	0.47	520
54	0.81	0.45	0.57	525
55	0.58	0.17	0.26	524
56	0.37	0.11	0.16	499
57	0.93	0.75	0.83	507
58	0.75	0.50	0.60	460
59	0.47	0.17	0.25	513
60	0.22	0.02	0.04	539
61	0.80	0.52	0.63	452
62	0.95	0.69	0.80	467
63	0.89	0.22	0.36	507
64	0.27	0.05	0.08	506
65	0.74	0.31	0.44	430
66	0.59	0.02	0.04	454
67	0.77	0.56	0.65	430
68	0.83	0.46	0.59	419
69	0.41	0.15	0.21	474
70	0.80	0.54	0.64	407
71	0.62	0.36	0.46	426
72	0.80	0.25	0.38	427
73	0.52	0.17	0.25	431
74	0.58	0.41	0.48	426
75	0.92	0.65	0.77	433
76	0.56	0.34	0.42	429
77	0.69	0.46	0.55	386
78	0.29	0.01	0.03	404
79	0.73	0.39	0.51	395
80	0.37	0.08	0.13	384
81	0.55	0.31	0.39	367
82	0.44	0.15	0.23	398
83	0.46	0.21	0.29	362
84	0.52	0.20	0.29	388
85	0.97	0.52	0.68	352
86	0.82	0.49	0.61	361
87	0.78	0.49	0.60	389
88	0.73	0.54	0.62	340
89	0.94	0.58	0.72	364
90	0.71	0.08	0.14	364
91	0.80	0.56	0.66	355
92	0.86	0.67	0.75	325
93	0.67	0.42	0.51	331
94	0.65	0.50	0.56	324
95	0.67	0.14	0.24	332
96	0.46	0.14	0.22	332
97	0.52	0.23	0.32	333
98	0.95	0.69	0.80	304
99	0.19	0.03	0.05	321
100	0.94	0.67	0.78	306
101	0.88	0.69	0.77	318
102	0.94	0.70	0.80	319
103	0.72	0.21	0.32	307
104	0.70	0.40	0.51	288
105	0.92	0.61	0.73	303

106	0.14	0.02	0.03	327
107	0.69	0.32	0.44	294
108	0.23	0.02	0.04	316
109	0.83	0.50	0.62	274
110	0.31	0.09	0.14	287
111	0.42	0.20	0.27	268
112	0.44	0.22	0.29	272
113	0.59	0.13	0.21	280
114	0.65	0.45	0.53	266
115	0.56	0.27	0.37	298
116	0.59	0.25	0.35	258
117	0.48	0.12	0.19	264
118	0.65	0.22	0.33	275
119	0.44	0.15	0.22	259
120	0.93	0.71	0.81	251
121	0.44	0.16	0.23	294
122	0.67	0.40	0.50	258
123	0.96	0.67	0.79	268
124	0.74	0.43	0.55	255
125	0.85	0.60	0.70	267
126	0.27	0.08	0.12	297
127	0.95	0.84	0.89	245
128	0.20	0.06	0.09	271
129	0.44	0.15	0.23	269
130	0.26	0.07	0.11	235
131	0.35	0.04	0.08	245
132	0.00	0.00	0.00	262
133	0.69	0.19	0.30	265
134	0.36	0.18	0.24	236
135	0.17	0.02	0.03	262
136	0.94	0.73	0.82	259
137	0.55	0.10	0.17	281
138	0.79	0.55	0.65	253
139	0.70	0.40	0.51	268
140	0.78	0.59	0.67	277
141	0.66	0.43	0.52	235
142	0.54	0.26	0.35	255
143	0.90	0.76	0.83	253
144	0.33	0.05	0.09	255
145	0.32	0.11	0.17	243
146	0.61	0.47	0.53	232
147	0.35	0.05	0.08	232
148	0.15	0.03	0.06	234
149	0.48	0.29	0.36	231
150	0.95	0.80	0.87	229
151	0.58	0.10	0.17	226
152	0.80	0.46	0.58	257
153	0.35	0.12	0.18	223
154	0.35	0.09	0.15	238
155	0.51	0.21	0.30	234
156	0.64	0.07	0.13	222
157	0.49	0.13	0.20	205
158	0.43	0.12	0.19	244
159	0.47	0.23	0.31	239
160	0.38	0.08	0.13	230
161	0.77	0.34	0.47	223
162	0.69	0.50	0.58	238

163	0.92	0.74	0.82	211
164	0.52	0.32	0.40	208
165	0.51	0.17	0.25	233
166	0.39	0.12	0.19	227
167	0.52	0.30	0.38	217
168	0.51	0.23	0.31	208
169	0.63	0.18	0.28	245
170	0.50	0.27	0.35	245
171	0.31	0.09	0.13	208
172	0.33	0.01	0.03	210
173	0.40	0.20	0.26	215
174	0.56	0.18	0.27	203
175	0.34	0.16	0.22	196
176	0.93	0.70	0.80	214
177	0.77	0.46	0.58	224
178	0.84	0.68	0.76	212
179	0.00	0.00	0.00	200
180	0.29	0.03	0.05	220
181	0.39	0.04	0.07	188
182	0.79	0.36	0.50	217
183	0.73	0.47	0.57	206
184	0.73	0.49	0.58	215
185	0.29	0.09	0.14	181
186	0.41	0.17	0.24	192
187	0.76	0.48	0.59	191
188	0.94	0.63	0.75	211
189	0.53	0.09	0.16	192
190	0.73	0.54	0.62	194
191	0.97	0.79	0.87	191
192	0.72	0.30	0.42	195
193	0.41	0.16	0.23	197
194	0.74	0.44	0.55	174
195	0.41	0.15	0.22	199
196	0.61	0.09	0.15	197
197	0.84	0.58	0.69	192
198	0.89	0.62	0.73	204
199	0.10	0.01	0.02	203
200	0.35	0.07	0.11	184
201	0.98	0.61	0.75	181
202	0.71	0.38	0.49	183
203	0.56	0.20	0.29	202
204	0.32	0.04	0.07	172
205	0.78	0.40	0.53	183
206	0.16	0.04	0.06	181
207	0.80	0.41	0.54	201
208	0.30	0.12	0.17	197
209	0.69	0.28	0.40	181
210	0.79	0.34	0.48	185
211	0.37	0.09	0.14	185
212	0.68	0.34	0.46	174
213	0.72	0.36	0.48	185
214	0.28	0.06	0.10	189
215	0.73	0.37	0.49	173
216	0.00	0.00	0.00	175
217	0.32	0.07	0.11	168
218	0.85	0.51	0.64	172
219	0.50	0.29	0.37	184

220	0.20	0.02	0.04	162
221	0.52	0.15	0.23	159
222	0.28	0.03	0.06	158
223	0.94	0.76	0.84	176
224	0.54	0.07	0.13	182
225	0.70	0.52	0.60	166
226	0.47	0.09	0.15	195
227	0.98	0.71	0.82	184
228	0.70	0.40	0.51	177
229	0.51	0.19	0.28	190
230	0.31	0.08	0.13	186
231	0.98	0.58	0.73	170
232	0.33	0.12	0.17	180
233	0.52	0.33	0.40	150
234	0.29	0.12	0.17	160
235	0.82	0.58	0.68	172
236	0.38	0.03	0.06	158
237	0.80	0.59	0.68	169
238	0.81	0.55	0.65	160
239	0.44	0.14	0.22	169
240	0.67	0.13	0.22	156
241	0.55	0.16	0.25	172
242	0.31	0.10	0.15	157
243	0.66	0.46	0.54	169
244	0.85	0.58	0.69	158
245	0.44	0.08	0.14	169
246	0.40	0.07	0.11	151
247	0.63	0.25	0.35	163
248	0.69	0.44	0.54	138
249	0.47	0.11	0.18	149
250	0.46	0.26	0.34	159
251	0.78	0.36	0.50	160
252	0.83	0.03	0.06	158
253	0.61	0.20	0.30	169
254	0.60	0.19	0.29	151
255	0.54	0.25	0.34	155
256	0.36	0.03	0.06	154
257	0.40	0.01	0.03	148
258	0.70	0.47	0.56	137
259	0.29	0.07	0.11	159
260	0.83	0.37	0.51	130
261	0.91	0.56	0.69	154
262	0.83	0.65	0.73	144
263	0.20	0.02	0.04	151
264	0.91	0.67	0.77	148
265	0.61	0.35	0.44	141
266	0.53	0.33	0.41	152
267	0.27	0.02	0.04	152
268	0.42	0.25	0.31	122
269	0.13	0.01	0.03	143
270	0.84	0.61	0.71	126
271	0.47	0.11	0.18	159
272	0.72	0.39	0.51	120
273	0.00	0.00	0.00	137
274	0.47	0.09	0.16	150
275	0.71	0.50	0.59	136
276	0.69	0.30	0.42	153

277	0.67	0.30	0.41	142
278	0.57	0.09	0.15	150
279	0.81	0.58	0.68	122
280	0.93	0.82	0.87	134
281	0.70	0.38	0.49	141
282	0.00	0.00	0.00	153
283	0.98	0.78	0.87	159
284	0.50	0.20	0.28	141
285	0.26	0.06	0.10	115
286	0.63	0.09	0.16	129
287	0.30	0.04	0.07	141
288	0.88	0.68	0.76	142
289	0.57	0.21	0.31	143
290	0.48	0.21	0.29	118
291	0.60	0.16	0.25	134
292	0.43	0.07	0.12	124
293	0.00	0.00	0.00	113
294	0.82	0.62	0.71	125
295	0.49	0.28	0.36	137
296	0.37	0.12	0.18	116
297	0.82	0.55	0.66	148
298	0.63	0.31	0.41	131
299	0.62	0.33	0.43	139
300	0.25	0.03	0.06	126
301	0.50	0.09	0.16	141
302	0.10	0.01	0.01	131
303	0.38	0.15	0.21	123
304	0.27	0.08	0.12	133
305	0.28	0.10	0.14	115
306	0.81	0.11	0.20	114
307	0.33	0.08	0.12	130
308	0.51	0.21	0.30	125
309	0.60	0.24	0.34	131
310	0.60	0.38	0.46	128
311	0.14	0.02	0.03	121
312	0.12	0.01	0.01	140
313	0.17	0.04	0.06	107
314	0.55	0.30	0.39	117
315	0.00	0.00	0.00	119
316	0.62	0.30	0.41	119
317	0.18	0.02	0.03	120
318	0.80	0.45	0.57	116
319	0.80	0.48	0.60	133
320	0.58	0.30	0.40	122
321	0.78	0.38	0.51	124
322	0.24	0.06	0.09	120
323	0.49	0.19	0.28	108
324	0.25	0.02	0.03	117
325	0.00	0.00	0.00	98
326	0.08	0.02	0.03	106
327	0.00	0.00	0.00	135
328	0.47	0.18	0.26	127
329	0.50	0.19	0.28	121
330	0.76	0.31	0.44	113
331	0.23	0.05	0.08	135
332	0.21	0.03	0.06	116
333	0.50	0.15	0.23	101

334	0.58	0.36	0.44	118
335	0.50	0.14	0.22	124
336	0.00	0.00	0.00	109
337	0.69	0.16	0.26	113
338	0.31	0.03	0.06	118
339	0.28	0.05	0.08	105
340	0.60	0.34	0.43	103
341	0.63	0.25	0.36	124
342	0.60	0.31	0.41	115
343	0.53	0.22	0.31	111
344	0.12	0.02	0.03	118
345	0.25	0.03	0.05	118
346	0.45	0.24	0.31	105
347	0.28	0.05	0.08	106
348	0.76	0.29	0.42	117
349	0.00	0.00	0.00	102
350	0.42	0.22	0.29	119
351	0.69	0.39	0.50	113
352	0.48	0.16	0.24	95
353	0.32	0.11	0.16	110
354	0.12	0.02	0.03	116
355	0.52	0.24	0.33	114
356	0.95	0.52	0.67	116
357	0.62	0.28	0.38	122
358	0.80	0.50	0.61	131
359	0.90	0.50	0.64	105
360	0.59	0.09	0.16	108
361	0.38	0.05	0.10	110
362	0.92	0.63	0.75	115
363	0.31	0.04	0.07	105
364	0.36	0.11	0.17	107
365	0.54	0.33	0.41	107
366	0.00	0.00	0.00	105
367	0.38	0.19	0.26	113
368	0.64	0.37	0.47	101
369	0.45	0.15	0.22	102
370	0.95	0.83	0.89	98
371	0.56	0.40	0.47	101
372	0.06	0.01	0.02	104
373	0.38	0.11	0.18	114
374	0.20	0.02	0.04	104
375	0.04	0.01	0.02	92
376	0.00	0.00	0.00	105
377	0.27	0.04	0.06	109
378	0.88	0.34	0.49	105
379	0.10	0.02	0.03	98
380	0.86	0.39	0.54	109
381	0.16	0.03	0.05	93
382	0.21	0.07	0.10	91
383	0.40	0.10	0.16	100
384	0.95	0.89	0.92	108
385	0.38	0.17	0.24	87
386	0.61	0.24	0.35	95
387	0.76	0.45	0.56	98
388	0.57	0.13	0.21	94
389	0.85	0.19	0.31	117
390	0.92	0.66	0.77	92

391	0.06	0.01	0.02	94
392	0.10	0.02	0.04	92
393	0.00	0.00	0.00	93
394	0.79	0.49	0.60	90
395	0.87	0.54	0.67	120
396	0.78	0.48	0.60	83
397	0.00	0.00	0.00	82
398	0.78	0.47	0.59	97
399	0.87	0.50	0.64	96
400	0.33	0.01	0.02	95
401	1.00	0.44	0.61	96
402	0.38	0.17	0.23	95
403	0.48	0.23	0.31	91
404	0.00	0.00	0.00	96
405	0.17	0.02	0.04	92
406	0.59	0.23	0.33	97
407	0.10	0.02	0.03	97
408	0.67	0.28	0.39	93
409	0.98	0.51	0.67	81
410	0.77	0.10	0.17	104
411	0.23	0.11	0.15	91
412	0.46	0.16	0.24	101
413	0.00	0.00	0.00	98
414	0.62	0.11	0.18	94
415	0.55	0.18	0.27	94
416	0.20	0.02	0.04	92
417	0.50	0.01	0.02	81
418	0.50	0.09	0.15	100
419	0.98	0.54	0.69	84
420	0.00	0.00	0.00	103
421	0.00	0.00	0.00	94
422	0.94	0.53	0.68	95
423	0.24	0.08	0.12	97
424	0.59	0.22	0.32	87
425	0.00	0.00	0.00	94
426	0.82	0.58	0.68	92
427	0.63	0.45	0.53	89
428	0.59	0.32	0.42	93
429	0.27	0.10	0.15	78
430	0.94	0.50	0.65	94
431	0.74	0.48	0.58	94
432	0.98	0.47	0.64	91
433	0.23	0.05	0.08	99
434	0.39	0.08	0.14	83
435	0.38	0.15	0.22	92
436	0.00	0.00	0.00	79
437	0.00	0.00	0.00	99
438	0.88	0.62	0.73	84
439	0.85	0.65	0.74	84
440	0.46	0.18	0.26	92
441	0.62	0.18	0.28	84
442	0.15	0.03	0.05	90
443	0.22	0.07	0.11	82
444	0.81	0.60	0.69	90
445	0.52	0.27	0.36	85
446	0.59	0.36	0.45	103
447	0.91	0.45	0.60	87

448	0.25	0.01	0.02	80
449	0.43	0.10	0.17	88
450	0.00	0.00	0.00	89
451	0.83	0.77	0.80	100
452	0.35	0.11	0.17	81
453	0.55	0.12	0.19	93
454	0.41	0.10	0.16	90
455	0.29	0.02	0.03	111
456	0.49	0.36	0.41	76
457	0.67	0.48	0.56	92
458	0.40	0.02	0.04	90
459	0.52	0.21	0.30	81
460	0.62	0.21	0.32	70
461	0.54	0.45	0.49	62
462	0.25	0.03	0.06	89
463	0.50	0.14	0.21	103
464	0.93	0.68	0.79	82
465	0.57	0.17	0.26	96
466	0.46	0.25	0.32	93
467	0.00	0.00	0.00	84
468	0.65	0.34	0.45	70
469	0.48	0.23	0.31	95
470	0.38	0.03	0.06	87
471	0.25	0.06	0.10	77
472	0.95	0.62	0.75	91
473	0.55	0.24	0.34	91
474	0.80	0.74	0.77	89
475	0.56	0.16	0.25	85
476	0.96	0.66	0.78	76
477	0.51	0.30	0.38	83
478	0.74	0.48	0.58	82
479	0.37	0.15	0.22	91
480	0.78	0.54	0.64	87
481	0.89	0.63	0.74	90
482	0.42	0.09	0.14	92
483	0.43	0.07	0.13	80
484	0.59	0.12	0.20	84
485	0.59	0.11	0.18	92
486	1.00	0.05	0.10	92
487	0.97	0.48	0.64	79
488	0.33	0.02	0.04	84
489	0.86	0.64	0.74	76
490	0.67	0.21	0.32	67
491	0.29	0.14	0.18	74
492	0.46	0.18	0.26	100
493	0.48	0.29	0.36	76
494	0.88	0.45	0.59	83
495	0.44	0.14	0.22	76
496	0.56	0.38	0.45	76
497	0.00	0.00	0.00	82
498	0.08	0.01	0.02	81
499	0.78	0.30	0.44	69
avg / total	0.65	0.35	0.44	180360

```
Time taken to run this cell : 0:11:11.714885
```

SGD Classifier with hinge loss

In [11]:
```
start = datetime.now()
classifier_hinge = OneVsRestClassifier(SGDClassifier(loss='hinge', alpha=0.000
01, penalty='l1'), n_jobs=-1)
classifier_hinge.fit(x_train_multilabel, y_train)
predictions = classifier_hinge.predict (x_test_multilabel)


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

```
Accuracy : 0.24654
Hamming loss  0.0027158
Micro-average quality numbers
Precision: 0.8102, Recall: 0.3227, F1-measure: 0.4616
Macro-average quality numbers
Precision: 0.4128, Recall: 0.2395, F1-measure: 0.2790
             precision    recall  f1-score   support

          0       0.68      0.14      0.23      7740
          1       0.82      0.45      0.58      7046
          2       0.85      0.56      0.68      6670
          3       0.84      0.38      0.52      6309
          4       0.94      0.78      0.86      5580
          5       0.88      0.62      0.73      5315
          6       0.81      0.23      0.36      3473
          7       0.91      0.67      0.77      3262
          8       0.78      0.40      0.53      3102
          9       0.82      0.40      0.54      2986
         10       0.88      0.59      0.71      2974
         11       0.71      0.03      0.06      2862
         12       0.78      0.00      0.01      2690
         13       0.71      0.27      0.39      2452
         14       0.78      0.19      0.31      2300
         15       0.70      0.25      0.36      2357
         16       0.81      0.51      0.63      2252
         17       0.81      0.65      0.72      2020
         18       0.70      0.28      0.40      1764
         19       0.64      0.03      0.07      1660
         20       0.00      0.00      0.00      1489
         21       0.85      0.40      0.54      1272
         22       0.64      0.35      0.46      1334
         23       0.89      0.61      0.73      1055
         24       0.69      0.46      0.56      1078
         25       0.67      0.48      0.56      1021
         26       0.00      0.00      0.00      1031
         27       0.87      0.71      0.78       999
         28       0.64      0.36      0.46       974
         29       0.76      0.29      0.42       936
         30       0.69      0.04      0.08       839
         31       0.93      0.82      0.87       883
         32       0.88      0.33      0.48       823
         33       0.69      0.23      0.34       808
         34       1.00      0.01      0.02       778
         35       0.75      0.65      0.70       743
         36       0.77      0.60      0.67       727
         37       0.76      0.80      0.78       732
         38       0.00      0.00      0.00       741
         39       0.00      0.00      0.00       666
         40       0.75      0.24      0.37       626
         41       0.00      0.00      0.00       640
         42       0.68      0.39      0.49       634
         43       0.00      0.00      0.00       601
         44       0.55      0.03      0.06       586
         45       0.61      0.33      0.43       599
         46       0.00      0.00      0.00       569
         47       0.00      0.00      0.00       525
         48       0.77      0.10      0.18       560
```

49	0.00	0.00	0.00	550
50	0.72	0.60	0.65	549
51	0.00	0.00	0.00	487
52	0.88	0.76	0.82	514
53	0.62	0.43	0.51	520
54	0.86	0.45	0.59	525
55	0.00	0.00	0.00	524
56	0.00	0.00	0.00	499
57	0.92	0.79	0.85	507
58	0.76	0.59	0.66	460
59	0.00	0.00	0.00	513
60	0.00	0.00	0.00	539
61	0.77	0.62	0.69	452
62	0.93	0.72	0.81	467
63	0.94	0.23	0.38	507
64	0.00	0.00	0.00	506
65	0.73	0.34	0.46	430
66	0.92	0.02	0.05	454
67	0.75	0.62	0.68	430
68	0.79	0.58	0.67	419
69	0.00	0.00	0.00	474
70	0.79	0.63	0.70	407
71	0.63	0.42	0.51	426
72	0.81	0.32	0.46	427
73	0.66	0.20	0.31	431
74	0.55	0.49	0.52	426
75	0.90	0.71	0.79	433
76	0.62	0.02	0.04	429
77	0.69	0.60	0.64	386
78	0.00	0.00	0.00	404
79	0.76	0.43	0.55	395
80	0.00	0.00	0.00	384
81	0.64	0.04	0.07	367
82	0.25	0.01	0.01	398
83	0.83	0.01	0.03	362
84	0.69	0.18	0.28	388
85	0.91	0.66	0.77	352
86	0.81	0.55	0.66	361
87	0.75	0.58	0.66	389
88	0.72	0.64	0.67	340
89	0.89	0.71	0.79	364
90	0.88	0.08	0.15	364
91	0.82	0.65	0.73	355
92	0.87	0.66	0.75	325
93	0.72	0.42	0.53	331
94	0.66	0.56	0.60	324
95	0.74	0.09	0.17	332
96	0.00	0.00	0.00	332
97	0.00	0.00	0.00	333
98	0.94	0.75	0.83	304
99	0.00	0.00	0.00	321
100	0.88	0.77	0.82	306
101	0.86	0.79	0.82	318
102	0.88	0.81	0.85	319
103	0.92	0.04	0.07	307
104	0.69	0.42	0.52	288
105	0.89	0.66	0.76	303

```
         106       0.00      0.00      0.00       327
         107       0.69      0.33      0.45       294
         108       0.00      0.00      0.00       316
         109       0.82      0.58      0.68       274
         110       0.00      0.00      0.00       287
         111       0.00      0.00      0.00       268
         112       0.00      0.00      0.00       272
         113       0.79      0.09      0.17       280
         114       0.66      0.59      0.62       266
         115       0.71      0.18      0.29       298
         116       0.74      0.05      0.10       258
         117       0.00      0.00      0.00       264
         118       0.83      0.02      0.04       275
         119       0.00      0.00      0.00       259
         120       0.88      0.76      0.82       251
         121       0.00      0.00      0.00       294
         122       0.75      0.45      0.56       258
         123       0.89      0.75      0.81       268
         124       0.76      0.53      0.63       255
         125       0.84      0.75      0.79       267
         126       0.00      0.00      0.00       297
         127       0.95      0.90      0.92       245
         128       0.00      0.00      0.00       271
         129       0.00      0.00      0.00       269
         130       0.00      0.00      0.00       235
         131       0.00      0.00      0.00       245
         132       0.00      0.00      0.00       262
         133       0.78      0.14      0.23       265
         134       0.00      0.00      0.00       236
         135       0.00      0.00      0.00       262
         136       0.91      0.85      0.88       259
         137       0.00      0.00      0.00       281
         138       0.77      0.66      0.71       253
         139       0.69      0.50      0.58       268
         140       0.76      0.65      0.70       277
         141       0.63      0.46      0.53       235
         142       0.00      0.00      0.00       255
         143       0.90      0.83      0.87       253
         144       0.00      0.00      0.00       255
         145       0.00      0.00      0.00       243
         146       0.63      0.60      0.61       232
         147       0.00      0.00      0.00       232
         148       0.00      0.00      0.00       234
         149       0.74      0.09      0.16       231
         150       0.94      0.86      0.90       229
         151       0.60      0.08      0.14       226
         152       0.78      0.45      0.57       257
         153       0.00      0.00      0.00       223
         154       0.00      0.00      0.00       238
         155       0.66      0.16      0.26       234
         156       0.65      0.05      0.09       222
         157       0.00      0.00      0.00       205
         158       0.00      0.00      0.00       244
         159       0.00      0.00      0.00       239
         160       0.00      0.00      0.00       230
         161       0.84      0.24      0.38       223
         162       0.68      0.66      0.67       238
```

163	0.89	0.82	0.86	211
164	0.67	0.10	0.17	208
165	0.00	0.00	0.00	233
166	0.00	0.00	0.00	227
167	0.40	0.01	0.02	217
168	0.80	0.02	0.04	208
169	0.00	0.00	0.00	245
170	0.00	0.00	0.00	245
171	0.00	0.00	0.00	208
172	0.00	0.00	0.00	210
173	0.00	0.00	0.00	215
174	0.00	0.00	0.00	203
175	0.00	0.00	0.00	196
176	0.92	0.76	0.83	214
177	0.91	0.45	0.60	224
178	0.83	0.78	0.81	212
179	0.00	0.00	0.00	200
180	0.00	0.00	0.00	220
181	0.00	0.00	0.00	188
182	0.79	0.37	0.51	217
183	0.71	0.58	0.64	206
184	0.80	0.40	0.54	215
185	0.00	0.00	0.00	181
186	0.00	0.00	0.00	192
187	0.77	0.59	0.67	191
188	0.89	0.75	0.82	211
189	0.72	0.07	0.12	192
190	0.71	0.71	0.71	194
191	0.91	0.91	0.91	191
192	0.40	0.01	0.02	195
193	0.75	0.02	0.03	197
194	0.72	0.49	0.59	174
195	0.00	0.00	0.00	199
196	0.80	0.12	0.21	197
197	0.85	0.69	0.76	192
198	0.85	0.76	0.80	204
199	0.00	0.00	0.00	203
200	0.00	0.00	0.00	184
201	0.87	0.73	0.80	181
202	0.73	0.38	0.50	183
203	0.00	0.00	0.00	202
204	0.00	0.00	0.00	172
205	0.84	0.56	0.67	183
206	0.00	0.00	0.00	181
207	0.84	0.42	0.56	201
208	0.00	0.00	0.00	197
209	0.70	0.34	0.46	181
210	0.84	0.34	0.48	185
211	0.00	0.00	0.00	185
212	0.62	0.40	0.49	174
213	0.77	0.34	0.47	185
214	0.00	0.00	0.00	189
215	0.72	0.36	0.48	173
216	0.00	0.00	0.00	175
217	0.00	0.00	0.00	168
218	0.83	0.59	0.69	172
219	0.54	0.12	0.20	184

220	0.00	0.00	0.00	162
221	0.67	0.05	0.09	159
222	0.00	0.00	0.00	158
223	0.92	0.80	0.86	176
224	0.88	0.04	0.07	182
225	0.69	0.69	0.69	166
226	0.00	0.00	0.00	195
227	0.95	0.79	0.86	184
228	0.66	0.58	0.62	177
229	0.00	0.00	0.00	190
230	0.00	0.00	0.00	186
231	0.93	0.78	0.85	170
232	0.00	0.00	0.00	180
233	0.57	0.40	0.47	150
234	0.00	0.00	0.00	160
235	0.80	0.73	0.77	172
236	0.00	0.00	0.00	158
237	0.78	0.66	0.71	169
238	0.81	0.68	0.74	160
239	0.50	0.01	0.01	169
240	0.00	0.00	0.00	156
241	0.00	0.00	0.00	172
242	0.00	0.00	0.00	157
243	0.63	0.54	0.58	169
244	0.82	0.68	0.74	158
245	0.00	0.00	0.00	169
246	0.00	0.00	0.00	151
247	0.80	0.15	0.25	163
248	0.69	0.51	0.59	138
249	0.33	0.01	0.01	149
250	0.38	0.03	0.06	159
251	0.84	0.36	0.50	160
252	0.00	0.00	0.00	158
253	0.00	0.00	0.00	169
254	0.51	0.15	0.23	151
255	0.50	0.01	0.01	155
256	0.00	0.00	0.00	154
257	0.00	0.00	0.00	148
258	0.64	0.57	0.60	137
259	0.00	0.00	0.00	159
260	0.81	0.39	0.53	130
261	0.91	0.61	0.73	154
262	0.80	0.76	0.78	144
263	0.00	0.00	0.00	151
264	0.91	0.77	0.84	148
265	0.70	0.31	0.43	141
266	0.59	0.29	0.39	152
267	0.00	0.00	0.00	152
268	0.47	0.17	0.25	122
269	0.00	0.00	0.00	143
270	0.82	0.65	0.73	126
271	0.00	0.00	0.00	159
272	0.79	0.41	0.54	120
273	0.00	0.00	0.00	137
274	0.00	0.00	0.00	150
275	0.71	0.78	0.74	136
276	1.00	0.01	0.01	153

277	0.70	0.37	0.49	142
278	0.00	0.00	0.00	150
279	0.78	0.70	0.74	122
280	0.94	0.87	0.90	134
281	0.72	0.47	0.57	141
282	0.00	0.00	0.00	153
283	0.96	0.82	0.89	159
284	0.00	0.00	0.00	141
285	0.00	0.00	0.00	115
286	0.00	0.00	0.00	129
287	0.00	0.00	0.00	141
288	0.87	0.81	0.84	142
289	0.54	0.10	0.18	143
290	0.57	0.14	0.22	118
291	0.64	0.10	0.18	134
292	0.00	0.00	0.00	124
293	0.00	0.00	0.00	113
294	0.80	0.70	0.74	125
295	0.63	0.19	0.29	137
296	0.00	0.00	0.00	116
297	0.80	0.68	0.74	148
298	0.65	0.27	0.38	131
299	0.78	0.35	0.49	139
300	0.00	0.00	0.00	126
301	1.00	0.01	0.01	141
302	0.00	0.00	0.00	131
303	0.00	0.00	0.00	123
304	0.00	0.00	0.00	133
305	0.00	0.00	0.00	115
306	0.60	0.16	0.25	114
307	0.00	0.00	0.00	130
308	0.00	0.00	0.00	125
309	0.00	0.00	0.00	131
310	0.67	0.34	0.45	128
311	0.00	0.00	0.00	121
312	0.00	0.00	0.00	140
313	0.00	0.00	0.00	107
314	1.00	0.01	0.02	117
315	0.00	0.00	0.00	119
316	0.00	0.00	0.00	119
317	0.00	0.00	0.00	120
318	0.81	0.63	0.71	116
319	0.80	0.66	0.72	133
320	0.61	0.27	0.38	122
321	0.79	0.44	0.57	124
322	0.00	0.00	0.00	120
323	0.00	0.00	0.00	108
324	0.00	0.00	0.00	117
325	0.00	0.00	0.00	98
326	0.00	0.00	0.00	106
327	0.00	0.00	0.00	135
328	0.80	0.06	0.12	127
329	0.00	0.00	0.00	121
330	0.78	0.34	0.47	113
331	0.00	0.00	0.00	135
332	0.00	0.00	0.00	116
333	0.00	0.00	0.00	101

334	0.00	0.00	0.00	118
335	0.33	0.01	0.02	124
336	0.00	0.00	0.00	109
337	0.81	0.15	0.25	113
338	0.00	0.00	0.00	118
339	0.00	0.00	0.00	105
340	0.78	0.07	0.12	103
341	0.71	0.04	0.08	124
342	0.00	0.00	0.00	115
343	0.00	0.00	0.00	111
344	0.00	0.00	0.00	118
345	0.00	0.00	0.00	118
346	0.00	0.00	0.00	105
347	0.00	0.00	0.00	106
348	1.00	0.03	0.07	117
349	0.00	0.00	0.00	102
350	0.00	0.00	0.00	119
351	0.71	0.40	0.51	113
352	0.00	0.00	0.00	95
353	0.00	0.00	0.00	110
354	0.00	0.00	0.00	116
355	0.68	0.15	0.24	114
356	0.94	0.58	0.72	116
357	0.00	0.00	0.00	122
358	0.75	0.69	0.72	131
359	0.86	0.60	0.71	105
360	0.00	0.00	0.00	108
361	0.00	0.00	0.00	110
362	0.88	0.67	0.76	115
363	0.00	0.00	0.00	105
364	0.00	0.00	0.00	107
365	0.54	0.37	0.44	107
366	0.00	0.00	0.00	105
367	0.00	0.00	0.00	113
368	0.64	0.49	0.55	101
369	0.00	0.00	0.00	102
370	0.95	0.88	0.91	98
371	0.53	0.31	0.39	101
372	0.00	0.00	0.00	104
373	0.00	0.00	0.00	114
374	0.00	0.00	0.00	104
375	0.00	0.00	0.00	92
376	0.00	0.00	0.00	105
377	0.00	0.00	0.00	109
378	0.87	0.38	0.53	105
379	0.00	0.00	0.00	98
380	0.83	0.40	0.54	109
381	0.00	0.00	0.00	93
382	0.00	0.00	0.00	91
383	0.00	0.00	0.00	100
384	0.92	0.94	0.93	108
385	0.00	0.00	0.00	87
386	0.66	0.28	0.40	95
387	0.74	0.61	0.67	98
388	0.00	0.00	0.00	94
389	0.00	0.00	0.00	117
390	0.92	0.77	0.84	92

391	0.00	0.00	0.00	94
392	0.00	0.00	0.00	92
393	0.00	0.00	0.00	93
394	0.77	0.54	0.64	90
395	0.83	0.68	0.75	120
396	0.72	0.61	0.66	83
397	0.00	0.00	0.00	82
398	0.71	0.57	0.63	97
399	0.83	0.55	0.66	96
400	0.00	0.00	0.00	95
401	0.95	0.58	0.72	96
402	1.00	0.04	0.08	95
403	0.00	0.00	0.00	91
404	0.00	0.00	0.00	96
405	0.00	0.00	0.00	92
406	0.00	0.00	0.00	97
407	0.00	0.00	0.00	97
408	0.00	0.00	0.00	93
409	0.87	0.73	0.79	81
410	0.93	0.13	0.24	104
411	0.00	0.00	0.00	91
412	0.00	0.00	0.00	101
413	0.00	0.00	0.00	98
414	0.75	0.16	0.26	94
415	0.00	0.00	0.00	94
416	0.00	0.00	0.00	92
417	0.00	0.00	0.00	81
418	0.00	0.00	0.00	100
419	0.96	0.58	0.73	84
420	0.00	0.00	0.00	103
421	0.00	0.00	0.00	94
422	0.92	0.58	0.71	95
423	0.00	0.00	0.00	97
424	0.76	0.25	0.38	87
425	0.00	0.00	0.00	94
426	0.79	0.67	0.73	92
427	0.68	0.70	0.69	89
428	0.52	0.17	0.26	93
429	0.00	0.00	0.00	78
430	0.91	0.62	0.73	94
431	0.76	0.55	0.64	94
432	0.84	0.67	0.74	91
433	0.00	0.00	0.00	99
434	0.00	0.00	0.00	83
435	0.00	0.00	0.00	92
436	0.00	0.00	0.00	79
437	0.00	0.00	0.00	99
438	0.85	0.76	0.81	84
439	0.81	0.79	0.80	84
440	0.00	0.00	0.00	92
441	0.00	0.00	0.00	84
442	0.00	0.00	0.00	90
443	0.00	0.00	0.00	82
444	0.76	0.68	0.72	90
445	0.63	0.20	0.30	85
446	0.93	0.13	0.22	103
447	0.89	0.55	0.68	87

448	0.00	0.00	0.00	80
449	0.00	0.00	0.00	88
450	0.00	0.00	0.00	89
451	0.82	0.82	0.82	100
452	0.00	0.00	0.00	81
453	0.00	0.00	0.00	93
454	0.00	0.00	0.00	90
455	0.00	0.00	0.00	111
456	0.55	0.50	0.52	76
457	0.67	0.72	0.69	92
458	0.00	0.00	0.00	90
459	0.00	0.00	0.00	81
460	0.65	0.24	0.35	70
461	0.56	0.47	0.51	62
462	0.00	0.00	0.00	89
463	0.00	0.00	0.00	103
464	0.93	0.79	0.86	82
465	0.00	0.00	0.00	96
466	0.00	0.00	0.00	93
467	0.00	0.00	0.00	84
468	0.50	0.06	0.10	70
469	0.00	0.00	0.00	95
470	0.00	0.00	0.00	87
471	0.00	0.00	0.00	77
472	0.90	0.66	0.76	91
473	0.50	0.01	0.02	91
474	0.76	0.81	0.78	89
475	0.00	0.00	0.00	85
476	0.97	0.75	0.84	76
477	0.61	0.24	0.34	83
478	0.66	0.59	0.62	82
479	0.00	0.00	0.00	91
480	0.70	0.66	0.68	87
481	0.89	0.72	0.80	90
482	0.00	0.00	0.00	92
483	0.00	0.00	0.00	80
484	0.00	0.00	0.00	84
485	0.00	0.00	0.00	92
486	0.88	0.08	0.14	92
487	0.89	0.52	0.66	79
488	0.00	0.00	0.00	84
489	0.82	0.78	0.80	76
490	0.00	0.00	0.00	67
491	0.00	0.00	0.00	74
492	0.00	0.00	0.00	100
493	0.00	0.00	0.00	76
494	0.81	0.60	0.69	83
495	0.00	0.00	0.00	76
496	0.55	0.55	0.55	76
497	0.00	0.00	0.00	82
498	0.00	0.00	0.00	81
499	0.75	0.39	0.51	69
avg / total	0.60	0.32	0.39	180360

```
Time taken to run this cell : 0:12:22.488322
```

Conclusion

In [33]:
```
x = PrettyTable()

x.field_names = ["Model","Featurization","F1-Micro"]

x.add_row(["SGD with hingeloss", "BOW", "0.41"])
x.add_row(["SGD with gridsearch(alpha = 0.001)", "BOW", "0.42"])
x.add_row(["SGD with log loss", "TF-IDF", "0.47"])
x.add_row(["SGD with hingeloss", "TF-IDF", "0.46"])
e="0.49"
f="BOW"
g="Logistic Regression"
e = "\033[1;31m%s\033[0m" %e
f = "\033[1;31m%s\033[0m" %f
g = "\033[1;31m%s\033[0m" %g
x.add_row([g,f,e])
a="0.5"
b="TF-IDF"
c="Logistic Regression"
a = "\033[1;32m%s\033[0m" %a
b = "\033[1;32m%s\033[0m" %b
c = "\033[1;32m%s\033[0m" %c
x.add_row([c, b, a])
print(x)
```

```
+------------------------------------+---------------+----------+
|               Model                | Featurization | F1-Micro |
+------------------------------------+---------------+----------+
|         SGD with hingeloss         |      BOW      |   0.41   |
| SGD with gridsearch(alpha = 0.001) |      BOW      |   0.42   |
|         SGD with log loss          |     TF-IDF    |   0.47   |
|         SGD with hingeloss         |     TF-IDF    |   0.46   |
|        Logistic Regression         |      BOW      |   0.49   |
|        Logistic Regression         |     TF-IDF    |   0.5    |
+------------------------------------+---------------+----------+
```

- Analyzed the Stack Overflow tag prediction dataset taken from the Facebook Kaggle competition to find the best model with high precision and recall to predict the tags of a given body and title of question.
- Created train.db database file from csv file.
- Performed exploratory data analysis on the dataset like removing duplicates, finding total no of unique tags, most frequent tags.
- Considered only 0.5 million points due to computational limitations and Preprocessed the data by removing special characters, removing stop words, separating the code snippets and stemming the words.
- Performed featurization on input data using BOW and TF-IDF.
- Splitted the data into train and test
- Implemented Logistic regression and SGD classifiers for both BOW and TF-IDF
- Of all the models logistic regreession with BOW and TF-IDF performed better than rest of the models with F1-Micro of 0.49 and 0.5 respectively.
- Tried tuning hyperparameters using gridsearch for Logistic regreession but it is taking more than 24hrs to train, so implemented on Linear SVM.
- SGD classifier is comparatively fatser than logistic regression.

Future Enhancements

- Due to higher dimensions we used logistic regression, SGD classifier as of now. Should try to train the model using RandomForest or XGBoost classifier for Word2Vec featurization because the dimensionality is less for Word2Vec and it may take less time to train and might work well.