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

Stack Overflow: Tag Prediction

1. Business Problem

1.1 Description

Description

Stack Overflow is the largest, most trusted online community for developers to learn, share their progra

Stack Overflow is something which every programmer use one way or another. Each month, over 50 m

to learn, share their knowledge, and build their careers. It features questions and answers on a wide ra
The website serves as a platform for users to ask and answer questions, and, through membership an
and answers up or down and edit questions and answers in a fashion similar to a wiki or Digg. As of A
4,000,000 registered users, and it exceeded 10,000,000 questions in late August 2015. Based on the ty
eight most discussed topics on the site are: Java, JavaScript, C#, PHP, Android, jQuery, Python and HT

Problem Statemtent

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

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

1.2 Source / useful links

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

3.3 Cleaning and preprocessing of Questions

3.3.1 Preprocessing

1. Sample 1M data points
2. Separate out code-snippets from Body
3. Remove Spcial characters from Question title and description (not in code)
4. Remove stop words (Except 'C')
5. Remove HTML Tags
6. Convert all the characters into small letters
7. Use SnowballStemmer to stem the words

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

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

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\DELL\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
#http://www.sqlitetutorial.net/sqlite-python/create-tables/
def create_connection(db_file):
    """ create a database connection to the SQLite database
        specified by db_file
    :param db_file: database file
    :return: Connection object or None
    """
    try:
        conn = sqlite3.connect(db_file)
        return conn
    except Error as e:
        print(e)

    return None

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

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

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

sql_create_table = """CREATE TABLE IF NOT EXISTS QuestionsProcessed (question text NOT NUl
```

```
create_database_table("Processed.db", sql_create_table)
```

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

__ we create a new data base to store the sampled and preprocessed questions __

4. Machine Learning Models

4.1 Converting tags for multilabel problems

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

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

```
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))
```

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

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

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

4.3 Featurizing data

4.5 Modeling with less data points (0.1M data points) and more weig

```
#Taking 0.1 Million entries to a dataframe.
write_db = 'D:\AppliedAI\Homework-n-Assignments\# 19 Stack overflow tagging\Titlemoreweigh
if os.path.isfile(write_db):
    conn_r = create_connection(write_db)
    if conn_r is not None:
        preprocessed_data = pd.read_sql_query("""SELECT question, Tags FROM QuestionsProce
        preprocessed_data = preprocessed_data.head(100000)
conn_r.commit()
conn_r.close()
```

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

```
preprocessed_data.head()
```

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

```
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 : 100000
number of dimensions : 2
```

```
type(preprocessed_data)
```

```
pandas.core.frame.DataFrame
```

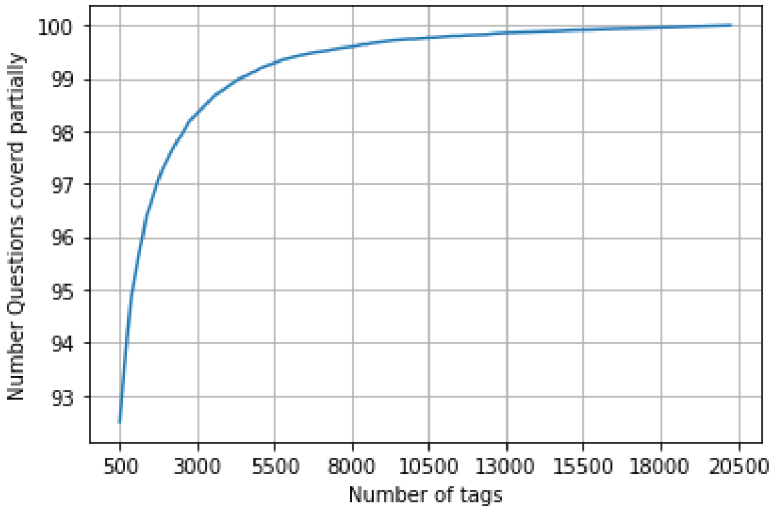
```
total_size=preprocessed_data.shape[0]
train_size=int(0.80*total_size)

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

__ Converting string Tags to multilable output variables __

__ Selecting 500 Tags __

```
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)*10
```

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




```
with  5500 tags we are covering  99.481 % of questions
with  500 tags we are covering  92.5 % of questions
```

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

```
number of questions that are not covered : 7500 out of  100000
```

```
train_datasize = 80000
```

```
x_train=preprocessed_data.head(train_datasize)
x_test=preprocessed_data.tail(preprocessed_data.shape[0] - 80000)
```

```
y_train = multilabel_yx[0:train_datasize,:]
```

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


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 : (80000, 500)
Number of data points in test data : (20000, 500)
```

Use bag of words upto 2 grams and compute the micro f1 sco regression(OvR)

4.5.2 Featurizing data with BOW vectorizer

```
start = datetime.now()
#vectorizer = TfidfVectorizer(min_df=0.00009, max_features=200000, smooth_idf=True, norm='
#                             tokenizer = lambda x: x.split(), sublinear_tf=False, ngram_r
vectorizer = CountVectorizer(min_df=0.00009, max_features=20000,tokenizer = lambda x: x.sp
x_train_multilabel = vectorizer.fit_transform(x_train['question'])
x_test_multilabel = vectorizer.transform(x_test['question'])
print("Time taken to run this cell :", datetime.now() - start)
```

```
Time taken to run this cell : 0:00:29.595038
```

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

```
Dimensions of train data X: (80000, 20000) Y : (80000, 500)
Dimensions of test data X: (20000, 20000) Y: (20000, 500)
```

4.5.3 Applying Logistic Regression with OneVsRest Classifier

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


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


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

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

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

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

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

```
Accuracy : 0.09885
Hamming loss  0.00580972
Micro-average quality numbers
Precision: 0.2915, Recall: 0.4694, F1-measure: 0.3597
Macro-average quality numbers
Precision: 0.2078, Recall: 0.4114, F1-measure: 0.2675
              precision    recall  f1-score   support

           0       0.75      0.79      0.77      5519
           1       0.45      0.42      0.43      8190
           2       0.53      0.49      0.51      6529
           3       0.52      0.60      0.56      3231
           4       0.55      0.51      0.53      6430
           5       0.44      0.45      0.44      2879
           6       0.59      0.60      0.60      5086
           7       0.64      0.63      0.63      4533
           8       0.24      0.21      0.23      3000
           9       0.55      0.65      0.60      2765
          10       0.32      0.30      0.31      3051
          11       0.45      0.50      0.47      3009
          12       0.38      0.41      0.39      2630
          13       0.37      0.43      0.40      1426
          14       0.60      0.67      0.63      2548
          15       0.36      0.37      0.37      2371
          16       0.28      0.36      0.32       873
          17       0.61      0.69      0.65      2151
          18       0.31      0.35      0.33      2204
          19       0.30      0.49      0.37       831
          20       0.52      0.56      0.54      1860
          21       0.19      0.24      0.22      2023
          22       0.30      0.36      0.33      1513
          23       0.50      0.67      0.57      1207
          24       0.25      0.34      0.28       506
          25       0.24      0.46      0.31       425
          26       0.38      0.49      0.43       793
          27       0.38      0.48      0.42      1291
          28       0.42      0.51      0.46      1208
          29       0.12      0.20      0.15       406
          30       0.20      0.34      0.25       504
          31       0.12      0.19      0.15       732
          32       0.23      0.45      0.30       441
          33       0.36      0.45      0.40      1645
          34       0.29      0.37      0.32      1058
          35       0.44      0.62      0.51       946
          36       0.24      0.36      0.29       644
          37       0.25      0.74      0.37       136
          38       0.29      0.49      0.36       570
          39       0.29      0.42      0.34       766
          40       0.35      0.46      0.40      1132
          41       0.14      0.33      0.20       174
          42       0.34      0.60      0.44       210
          43       0.33      0.52      0.41       433
          44       0.32      0.53      0.40       626
          45       0.34      0.48      0.39       852
          46       0.34      0.54      0.41       534
          47       0.16      0.30      0.21       350
          48       0.33      0.58      0.42       496
```

49	0.54	0.71	0.62	785
50	0.11	0.21	0.15	475
51	0.13	0.28	0.17	305
52	0.09	0.15	0.11	251
53	0.31	0.49	0.38	914
54	0.24	0.28	0.26	728
55	0.09	0.13	0.11	258
56	0.21	0.37	0.27	821
57	0.15	0.26	0.19	541
58	0.33	0.43	0.37	748
59	0.59	0.76	0.67	724
60	0.19	0.27	0.22	660
61	0.17	0.35	0.23	235
62	0.53	0.78	0.63	718
63	0.42	0.76	0.54	468
64	0.17	0.45	0.25	191
65	0.13	0.23	0.17	429
66	0.12	0.25	0.16	415
67	0.27	0.58	0.37	274
68	0.38	0.60	0.46	510
69	0.30	0.53	0.38	466
70	0.12	0.26	0.17	305
71	0.11	0.28	0.16	247
72	0.38	0.55	0.45	401
73	0.29	0.84	0.44	86
74	0.16	0.50	0.25	120
75	0.35	0.68	0.47	129
76	0.08	0.12	0.10	473
77	0.08	0.28	0.13	143
78	0.37	0.58	0.45	347
79	0.22	0.35	0.27	479
80	0.21	0.47	0.29	279
81	0.23	0.38	0.28	461
82	0.09	0.15	0.11	298
83	0.33	0.57	0.42	396
84	0.18	0.50	0.26	184
85	0.27	0.42	0.33	573
86	0.10	0.18	0.13	325
87	0.21	0.46	0.29	273
88	0.10	0.37	0.16	135
89	0.13	0.26	0.18	232
90	0.29	0.48	0.36	409
91	0.22	0.42	0.29	420
92	0.40	0.63	0.49	408
93	0.21	0.58	0.31	241
94	0.09	0.17	0.12	211
95	0.13	0.24	0.17	277
96	0.11	0.18	0.14	410
97	0.44	0.62	0.51	501
98	0.20	0.67	0.31	136
99	0.22	0.43	0.29	239
100	0.13	0.26	0.17	324
101	0.44	0.78	0.56	277
102	0.67	0.82	0.74	613
103	0.12	0.34	0.18	157
104	0.10	0.23	0.14	295
105	0.28	0.51	0.36	334
106	0.25	0.44	0.32	335

107	0.32	0.57	0.41	389
108	0.19	0.38	0.26	251
109	0.28	0.50	0.36	317
110	0.07	0.21	0.11	187
111	0.08	0.26	0.12	140
112	0.19	0.56	0.29	154
113	0.22	0.39	0.28	332
114	0.22	0.37	0.27	323
115	0.17	0.35	0.23	344
116	0.37	0.57	0.45	370
117	0.20	0.37	0.26	313
118	0.58	0.77	0.66	874
119	0.17	0.34	0.23	293
120	0.06	0.14	0.08	200
121	0.37	0.58	0.45	463
122	0.11	0.29	0.15	119
123	0.03	0.05	0.04	256
124	0.45	0.79	0.57	195
125	0.13	0.38	0.19	138
126	0.36	0.60	0.45	376
127	0.06	0.15	0.09	122
128	0.07	0.13	0.09	252
129	0.25	0.47	0.32	144
130	0.09	0.35	0.14	150
131	0.05	0.11	0.07	210
132	0.24	0.37	0.29	361
133	0.56	0.70	0.62	453
134	0.40	0.83	0.54	124
135	0.06	0.23	0.10	91
136	0.13	0.42	0.19	128
137	0.17	0.44	0.24	218
138	0.13	0.30	0.18	243
139	0.10	0.32	0.15	149
140	0.37	0.58	0.45	318
141	0.09	0.21	0.13	159
142	0.29	0.55	0.38	274
143	0.62	0.85	0.71	362
144	0.10	0.35	0.15	118
145	0.15	0.41	0.22	164
146	0.25	0.48	0.33	461
147	0.23	0.55	0.32	159
148	0.13	0.30	0.18	166
149	0.43	0.66	0.53	346
150	0.17	0.29	0.21	350
151	0.19	0.76	0.30	55
152	0.37	0.61	0.46	387
153	0.16	0.31	0.21	150
154	0.12	0.23	0.16	281
155	0.11	0.30	0.16	202
156	0.29	0.68	0.40	130
157	0.12	0.22	0.16	245
158	0.48	0.71	0.57	177
159	0.15	0.49	0.22	130
160	0.18	0.32	0.23	336
161	0.39	0.70	0.50	220
162	0.10	0.24	0.14	229
163	0.33	0.57	0.42	316
164	0.26	0.54	0.36	283

164	0.20	0.54	0.30	285
165	0.18	0.42	0.25	197
166	0.18	0.61	0.28	101
167	0.14	0.34	0.20	231
168	0.20	0.43	0.27	370
169	0.18	0.33	0.23	258
170	0.06	0.19	0.09	101
171	0.08	0.33	0.13	89
172	0.20	0.46	0.28	193
173	0.22	0.44	0.29	309
174	0.08	0.20	0.11	172
175	0.31	0.77	0.44	95
176	0.57	0.74	0.64	346
177	0.45	0.66	0.54	322
178	0.24	0.56	0.33	232
179	0.06	0.17	0.09	125
180	0.16	0.47	0.24	145
181	0.03	0.19	0.06	77
182	0.10	0.28	0.14	182
183	0.25	0.45	0.32	257
184	0.08	0.20	0.12	216
185	0.11	0.29	0.16	242
186	0.11	0.31	0.16	165
187	0.35	0.62	0.44	263
188	0.08	0.22	0.11	174
189	0.34	0.51	0.41	136
190	0.42	0.71	0.53	202
191	0.10	0.30	0.15	134
192	0.24	0.49	0.33	230
193	0.09	0.29	0.13	90
194	0.24	0.54	0.33	185
195	0.04	0.14	0.06	156
196	0.06	0.21	0.09	160
197	0.18	0.30	0.22	266
198	0.13	0.25	0.17	284
199	0.06	0.15	0.09	145
200	0.44	0.83	0.58	212
201	0.21	0.40	0.28	317
202	0.45	0.66	0.53	427
203	0.10	0.24	0.15	232
204	0.17	0.43	0.25	217
205	0.37	0.60	0.46	527
206	0.05	0.14	0.07	124
207	0.19	0.35	0.25	103
208	0.34	0.59	0.43	287
209	0.10	0.25	0.14	193
210	0.23	0.50	0.32	220
211	0.10	0.32	0.15	140
212	0.06	0.19	0.10	161
213	0.21	0.50	0.30	72
214	0.39	0.57	0.46	396
215	0.15	0.48	0.23	134
216	0.20	0.35	0.25	400
217	0.10	0.41	0.16	75
218	0.60	0.81	0.69	219
219	0.23	0.48	0.31	210
220	0.48	0.76	0.59	298
221	0.58	0.77	0.66	266

222	0.34	0.58	0.43	290
223	0.04	0.13	0.06	128
224	0.19	0.48	0.28	159
225	0.17	0.49	0.26	164
226	0.19	0.47	0.27	144
227	0.34	0.52	0.41	276
228	0.04	0.09	0.06	235
229	0.07	0.17	0.10	216
230	0.10	0.29	0.15	228
231	0.17	0.64	0.27	64
232	0.07	0.24	0.10	103
233	0.24	0.50	0.33	216
234	0.14	0.28	0.19	116
235	0.14	0.45	0.21	77
236	0.30	0.78	0.44	67
237	0.11	0.26	0.16	218
238	0.09	0.30	0.14	139
239	0.07	0.19	0.11	94
240	0.12	0.42	0.19	77
241	0.07	0.20	0.10	167
242	0.23	0.50	0.32	86
243	0.05	0.28	0.09	58
244	0.26	0.46	0.33	269
245	0.12	0.29	0.17	112
246	0.65	0.81	0.72	255
247	0.07	0.36	0.12	58
248	0.03	0.17	0.05	81
249	0.04	0.15	0.07	131
250	0.09	0.35	0.14	93
251	0.18	0.40	0.25	154
252	0.04	0.11	0.06	129
253	0.13	0.43	0.20	83
254	0.09	0.20	0.12	191
255	0.09	0.19	0.12	219
256	0.05	0.15	0.08	130
257	0.11	0.42	0.17	93
258	0.31	0.62	0.42	217
259	0.11	0.33	0.16	141
260	0.23	0.43	0.30	143
261	0.10	0.22	0.14	219
262	0.15	0.47	0.22	107
263	0.24	0.39	0.30	236
264	0.09	0.30	0.14	119
265	0.10	0.43	0.16	72
266	0.08	0.26	0.12	70
267	0.09	0.27	0.14	107
268	0.23	0.53	0.32	169
269	0.13	0.31	0.19	129
270	0.32	0.58	0.41	159
271	0.27	0.61	0.37	190
272	0.20	0.40	0.26	248
273	0.68	0.82	0.74	264
274	0.37	0.74	0.49	105
275	0.06	0.21	0.09	104
276	0.03	0.10	0.05	115
277	0.33	0.64	0.43	170
278	0.31	0.62	0.42	145
279	0.44	0.76	0.56	230

280	0.18	0.50	0.26	80
281	0.45	0.61	0.52	217
282	0.34	0.57	0.42	175
283	0.14	0.27	0.19	269
284	0.14	0.51	0.22	74
285	0.27	0.64	0.38	206
286	0.38	0.74	0.51	227
287	0.20	0.62	0.30	130
288	0.07	0.15	0.10	129
289	0.06	0.35	0.11	80
290	0.06	0.23	0.09	99
291	0.23	0.52	0.32	208
292	0.04	0.16	0.06	67
293	0.25	0.64	0.36	109
294	0.14	0.43	0.21	140
295	0.12	0.32	0.18	241
296	0.10	0.35	0.16	72
297	0.07	0.24	0.11	107
298	0.31	0.66	0.42	61
299	0.36	0.64	0.46	77
300	0.09	0.23	0.13	111
301	0.02	0.03	0.02	126
302	0.05	0.18	0.08	73
303	0.25	0.48	0.33	176
304	0.68	0.83	0.75	230
305	0.46	0.78	0.58	156
306	0.21	0.51	0.30	146
307	0.09	0.27	0.13	98
308	0.01	0.04	0.01	78
309	0.09	0.26	0.13	94
310	0.24	0.47	0.32	162
311	0.31	0.64	0.42	116
312	0.10	0.37	0.15	57
313	0.04	0.22	0.07	65
314	0.15	0.41	0.22	138
315	0.21	0.42	0.28	195
316	0.12	0.45	0.18	69
317	0.09	0.25	0.14	134
318	0.21	0.49	0.29	148
319	0.37	0.61	0.46	161
320	0.09	0.28	0.14	104
321	0.32	0.58	0.41	156
322	0.18	0.49	0.26	134
323	0.31	0.53	0.39	232
324	0.08	0.27	0.13	92
325	0.16	0.33	0.22	197
326	0.06	0.20	0.09	126
327	0.04	0.09	0.05	115
328	0.57	0.74	0.64	198
329	0.18	0.42	0.25	125
330	0.15	0.38	0.21	81
331	0.09	0.24	0.13	94
332	0.06	0.21	0.10	56
333	0.09	0.20	0.13	260
334	0.06	0.23	0.10	60
335	0.09	0.30	0.14	110
336	0.20	0.48	0.28	71
337	0.04	0.21	0.07	66

337	0.04	0.21	0.07	66
338	0.18	0.48	0.26	150
339	0.02	0.13	0.04	54
340	0.45	0.69	0.55	195
341	0.35	0.59	0.44	79
342	0.11	0.42	0.17	38
343	0.12	0.49	0.19	43
344	0.20	0.34	0.25	68
345	0.18	0.47	0.26	73
346	0.07	0.22	0.10	116
347	0.16	0.60	0.26	111
348	0.05	0.22	0.08	63
349	0.32	0.69	0.43	104
350	0.13	0.55	0.21	44
351	0.12	0.33	0.17	40
352	0.35	0.63	0.45	136
353	0.08	0.31	0.12	54
354	0.08	0.22	0.12	134
355	0.15	0.43	0.22	120
356	0.29	0.50	0.36	228
357	0.34	0.50	0.40	269
358	0.16	0.49	0.24	80
359	0.25	0.63	0.36	140
360	0.08	0.30	0.13	125
361	0.50	0.81	0.62	169
362	0.04	0.16	0.06	56
363	0.51	0.81	0.63	154
364	0.08	0.22	0.12	58
365	0.08	0.34	0.13	71
366	0.38	0.76	0.51	54
367	0.05	0.17	0.08	116
368	0.05	0.19	0.08	54
369	0.02	0.11	0.03	71
370	0.02	0.08	0.03	61
371	0.04	0.18	0.07	71
372	0.13	0.50	0.20	52
373	0.35	0.61	0.44	150
374	0.13	0.41	0.19	93
375	0.05	0.16	0.07	67
376	0.02	0.05	0.02	76
377	0.16	0.35	0.22	106
378	0.03	0.06	0.04	86
379	0.02	0.36	0.04	14
380	0.25	0.63	0.36	122
381	0.05	0.17	0.08	104
382	0.04	0.23	0.06	66
383	0.18	0.45	0.25	110
384	0.04	0.10	0.06	155
385	0.09	0.46	0.15	50
386	0.05	0.22	0.08	64
387	0.07	0.18	0.11	93
388	0.14	0.45	0.22	102
389	0.04	0.12	0.06	108
390	0.59	0.74	0.66	178
391	0.13	0.31	0.19	115
392	0.21	0.55	0.30	42
393	0.01	0.01	0.01	134
394	0.06	0.20	0.09	112

395	0.15	0.37	0.22	176
396	0.09	0.22	0.13	125
397	0.39	0.61	0.47	224
398	0.27	0.76	0.40	63
399	0.02	0.10	0.03	59
400	0.12	0.40	0.18	63
401	0.10	0.42	0.16	98
402	0.14	0.33	0.20	162
403	0.08	0.29	0.13	83
404	0.25	0.79	0.38	19
405	0.08	0.28	0.12	92
406	0.10	0.51	0.17	41
407	0.22	0.47	0.30	43
408	0.23	0.46	0.31	160
409	0.08	0.24	0.12	50
410	0.03	0.26	0.05	19
411	0.11	0.28	0.16	175
412	0.07	0.22	0.11	72
413	0.06	0.16	0.09	95
414	0.10	0.25	0.15	97
415	0.05	0.19	0.07	48
416	0.23	0.40	0.29	83
417	0.04	0.12	0.06	40
418	0.08	0.27	0.13	91
419	0.16	0.47	0.24	90
420	0.08	0.38	0.13	37
421	0.06	0.21	0.10	66
422	0.12	0.42	0.19	73
423	0.09	0.32	0.15	56
424	0.42	0.91	0.57	33
425	0.02	0.05	0.03	76
426	0.03	0.11	0.05	81
427	0.55	0.76	0.64	150
428	0.26	0.69	0.37	29
429	0.96	0.94	0.95	389
430	0.26	0.50	0.34	167
431	0.06	0.19	0.09	123
432	0.09	0.49	0.15	39
433	0.18	0.45	0.26	82
434	0.38	0.77	0.51	66
435	0.22	0.56	0.31	93
436	0.22	0.51	0.31	87
437	0.07	0.22	0.11	86
438	0.30	0.57	0.40	104
439	0.08	0.27	0.13	100
440	0.07	0.17	0.10	141
441	0.24	0.55	0.34	110
442	0.08	0.25	0.12	123
443	0.10	0.32	0.16	71
444	0.10	0.29	0.15	109
445	0.07	0.31	0.12	48
446	0.13	0.50	0.21	76
447	0.08	0.39	0.13	38
448	0.27	0.69	0.39	81
449	0.20	0.47	0.28	132
450	0.16	0.44	0.24	81
451	0.15	0.42	0.22	76
452	0.07	0.16	0.10	44

453	0.02	0.09	0.03	44
454	0.22	0.64	0.33	70
455	0.12	0.37	0.18	155
456	0.13	0.44	0.20	43
457	0.12	0.38	0.18	72
458	0.03	0.16	0.05	62
459	0.08	0.29	0.12	69
460	0.03	0.10	0.05	119
461	0.30	0.47	0.36	79
462	0.06	0.21	0.09	47
463	0.15	0.43	0.22	104
464	0.27	0.46	0.34	106
465	0.14	0.38	0.20	64
466	0.25	0.44	0.31	173
467	0.21	0.52	0.30	107
468	0.17	0.37	0.23	126
469	0.03	0.11	0.05	114
470	0.70	0.84	0.76	140
471	0.22	0.48	0.30	79
472	0.23	0.38	0.28	143
473	0.33	0.58	0.42	158
474	0.12	0.25	0.16	138
475	0.06	0.25	0.10	59
476	0.17	0.48	0.25	88
477	0.39	0.69	0.50	176
478	0.21	0.92	0.34	24
479	0.08	0.22	0.12	92
480	0.37	0.62	0.46	100
481	0.16	0.45	0.24	103
482	0.08	0.34	0.13	74
483	0.36	0.70	0.48	105
484	0.04	0.14	0.06	83
485	0.03	0.12	0.05	82
486	0.06	0.27	0.10	71
487	0.15	0.32	0.20	120
488	0.06	0.16	0.09	105
489	0.20	0.41	0.27	87
490	0.50	0.84	0.63	32
491	0.01	0.03	0.01	69
492	0.02	0.10	0.04	49
493	0.03	0.09	0.05	117
494	0.07	0.25	0.11	61
495	0.84	0.86	0.85	344
496	0.11	0.27	0.16	52
497	0.21	0.37	0.27	137
498	0.11	0.29	0.16	98
499	0.08	0.32	0.13	79
micro avg	0.29	0.47	0.36	173812
macro avg	0.21	0.41	0.27	173812
weighted avg	0.36	0.47	0.39	173812
samples avg	0.35	0.44	0.35	173812

```
Time taken to run this cell : 4:29:38.463351
```

```
joblib.dump(classifier, 'lr_with_more_title_weight.pkl')
```

```
joblib.dump(classifier, 'LR_with_more_than_100_tags_weights.pkl')
```

```
start = datetime.now()
classifier_2 = OneVsRestClassifier(LogisticRegression(penalty='l1'))
classifier_2.fit(x_train_multilabel, y_train)
predictions_2 = classifier_2.predict(x_test_multilabel)
print("Accuracy :",metrics.accuracy_score(y_test, predictions_2))
print("Hamming loss ",metrics.hamming_loss(y_test,predictions_2))


precision = precision_score(y_test, predictions_2, average='micro')
recall = recall_score(y_test, predictions_2, average='micro')
f1 = f1_score(y_test, predictions_2, 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_2, average='macro')
recall = recall_score(y_test, predictions_2, average='macro')
f1 = f1_score(y_test, predictions_2, average='macro')

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

print (metrics.classification_report(y_test, predictions_2))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Accuracy : 0.21231
Hamming loss  0.00313218
Micro-average quality numbers
Precision: 0.5687, Recall: 0.4097, F1-measure: 0.4763
Macro-average quality numbers
Precision: 0.4512, Recall: 0.3347, F1-measure: 0.3807
              precision    recall  f1-score   support

           0       0.90      0.73      0.81      5519
           1       0.52      0.41      0.46      8190
           2       0.64      0.47      0.54      6529
           3       0.68      0.52      0.59      3231
           4       0.66      0.49      0.56      6430
           5       0.62      0.42      0.50      2879
           6       0.74      0.57      0.64      5086
           7       0.75      0.61      0.68      4533
           8       0.35      0.18      0.24      3000
           9       0.70      0.59      0.64      2765
          10       0.42      0.29      0.35      3051
          11       0.60      0.45      0.51      3009
          12       0.48      0.36      0.41      2630
          13       0.54      0.38      0.44      1426
          14       0.80      0.61      0.69      2548
          15       0.48      0.30      0.37      2371
          16       0.54      0.29      0.38       873
          17       0.79      0.64      0.71      2151
          18       0.44      0.29      0.35      2204
          19       0.55      0.44      0.49       831
          20       0.70      0.49      0.57      1860
          21       0.26      0.18      0.21      2023
          22       0.40      0.30      0.34      1513
          23       0.76      0.58      0.66      1207
          24       0.45      0.34      0.39       506
          25       0.52      0.37      0.43       425
          26       0.57      0.44      0.50       793
          27       0.54      0.41      0.46      1291
          28       0.60      0.41      0.49      1208
          29       0.28      0.17      0.21       406
          30       0.46      0.24      0.31       504
          31       0.20      0.14      0.17       732
          32       0.47      0.32      0.38       441
          33       0.53      0.37      0.43      1645
          34       0.48      0.29      0.36      1058
          35       0.73      0.57      0.64       946
          36       0.48      0.29      0.36       644
          37       0.90      0.70      0.79       136
          38       0.51      0.38      0.43       570
          39       0.63      0.34      0.45       766
          40       0.53      0.43      0.47      1132
          41       0.35      0.31      0.33       174
          42       0.67      0.54      0.60       210
          43       0.64      0.45      0.53       433
          44       0.59      0.48      0.53       626
          45       0.56      0.38      0.46       852
          46       0.63      0.48      0.55       534
          47       0.30      0.24      0.26       350
          48       0.63      0.54      0.58       496
```

49	0.76	0.64	0.69	785
50	0.19	0.12	0.14	475
51	0.26	0.21	0.23	305
52	0.24	0.11	0.15	251
53	0.54	0.41	0.47	914
54	0.39	0.25	0.30	728
55	0.16	0.08	0.11	258
56	0.35	0.29	0.32	821
57	0.35	0.19	0.25	541
58	0.59	0.35	0.44	748
59	0.89	0.70	0.79	724
60	0.35	0.19	0.24	660
61	0.43	0.24	0.31	235
62	0.88	0.72	0.79	718
63	0.78	0.69	0.73	468
64	0.44	0.31	0.36	191
65	0.29	0.18	0.22	429
66	0.22	0.12	0.16	415
67	0.67	0.55	0.61	274
68	0.72	0.54	0.62	510
69	0.60	0.50	0.54	466
70	0.25	0.16	0.20	305
71	0.34	0.22	0.27	247
72	0.71	0.53	0.61	401
73	0.85	0.79	0.82	86
74	0.57	0.43	0.49	120
75	0.82	0.71	0.76	129
76	0.13	0.05	0.07	473
77	0.37	0.30	0.33	143
78	0.67	0.47	0.55	347
79	0.49	0.27	0.35	479
80	0.43	0.37	0.40	279
81	0.51	0.27	0.35	461
82	0.14	0.06	0.09	298
83	0.72	0.52	0.60	396
84	0.39	0.38	0.38	184
85	0.43	0.30	0.35	573
86	0.27	0.12	0.17	325
87	0.50	0.42	0.46	273
88	0.46	0.31	0.37	135
89	0.25	0.17	0.20	232
90	0.49	0.41	0.45	409
91	0.51	0.33	0.40	420
92	0.68	0.57	0.62	408
93	0.55	0.49	0.52	241
94	0.20	0.09	0.13	211
95	0.31	0.17	0.22	277
96	0.21	0.12	0.15	410
97	0.76	0.47	0.58	501
98	0.67	0.62	0.64	136
99	0.45	0.37	0.41	239
100	0.33	0.20	0.25	324
101	0.85	0.73	0.79	277
102	0.89	0.76	0.82	613
103	0.37	0.23	0.28	157
104	0.21	0.12	0.15	295
105	0.65	0.46	0.54	334
106	0.66	0.36	0.47	335

107	0.69	0.58	0.63	389
108	0.51	0.33	0.40	251
109	0.56	0.47	0.51	317
110	0.30	0.11	0.16	187
111	0.45	0.19	0.26	140
112	0.57	0.47	0.52	154
113	0.49	0.28	0.36	332
114	0.43	0.28	0.34	323
115	0.44	0.33	0.38	344
116	0.67	0.54	0.60	370
117	0.44	0.30	0.36	313
118	0.76	0.75	0.75	874
119	0.36	0.26	0.30	293
120	0.13	0.09	0.10	200
121	0.66	0.50	0.57	463
122	0.23	0.12	0.16	119
123	0.15	0.04	0.06	256
124	0.86	0.72	0.78	195
125	0.30	0.18	0.23	138
126	0.72	0.53	0.61	376
127	0.15	0.07	0.09	122
128	0.14	0.06	0.08	252
129	0.43	0.38	0.40	144
130	0.31	0.18	0.23	150
131	0.19	0.09	0.12	210
132	0.51	0.33	0.40	361
133	0.84	0.64	0.73	453
134	0.81	0.77	0.79	124
135	0.15	0.12	0.13	91
136	0.51	0.38	0.43	128
137	0.45	0.39	0.42	218
138	0.34	0.21	0.26	243
139	0.30	0.19	0.23	149
140	0.69	0.54	0.61	318
141	0.19	0.12	0.15	159
142	0.57	0.42	0.49	274
143	0.81	0.83	0.82	362
144	0.43	0.28	0.34	118
145	0.50	0.43	0.46	164
146	0.51	0.39	0.44	461
147	0.69	0.43	0.53	159
148	0.32	0.20	0.25	166
149	0.90	0.59	0.72	346
150	0.49	0.22	0.30	350
151	0.90	0.67	0.77	55
152	0.71	0.52	0.60	387
153	0.37	0.33	0.35	150
154	0.32	0.15	0.20	281
155	0.25	0.19	0.22	202
156	0.74	0.65	0.69	130
157	0.21	0.10	0.13	245
158	0.91	0.70	0.79	177
159	0.46	0.40	0.43	130
160	0.38	0.24	0.29	336
161	0.79	0.65	0.71	220
162	0.19	0.11	0.14	229
163	0.78	0.46	0.58	316
164	0.63	0.42	0.50	283

164	0.65	0.42	0.50	285
165	0.53	0.37	0.43	197
166	0.54	0.53	0.54	101
167	0.37	0.23	0.28	231
168	0.43	0.35	0.39	370
169	0.39	0.24	0.29	258
170	0.27	0.17	0.21	101
171	0.34	0.28	0.31	89
172	0.48	0.38	0.42	193
173	0.46	0.33	0.39	309
174	0.30	0.14	0.19	172
175	0.79	0.74	0.76	95
176	0.85	0.63	0.73	346
177	0.81	0.60	0.69	322
178	0.53	0.47	0.50	232
179	0.21	0.10	0.14	125
180	0.46	0.40	0.43	145
181	0.30	0.21	0.25	77
182	0.18	0.12	0.14	182
183	0.52	0.37	0.43	257
184	0.23	0.13	0.17	216
185	0.31	0.20	0.24	242
186	0.34	0.22	0.27	165
187	0.68	0.57	0.62	263
188	0.19	0.10	0.13	174
189	0.64	0.48	0.55	136
190	0.82	0.59	0.69	202
191	0.29	0.21	0.24	134
192	0.58	0.46	0.51	230
193	0.26	0.19	0.22	90
194	0.55	0.55	0.55	185
195	0.16	0.08	0.10	156
196	0.14	0.09	0.11	160
197	0.29	0.17	0.21	266
198	0.28	0.15	0.20	284
199	0.19	0.08	0.11	145
200	0.86	0.77	0.81	212
201	0.48	0.26	0.33	317
202	0.69	0.63	0.66	427
203	0.21	0.15	0.17	232
204	0.38	0.29	0.33	217
205	0.49	0.49	0.49	527
206	0.15	0.06	0.09	124
207	0.40	0.34	0.37	103
208	0.77	0.55	0.64	287
209	0.20	0.11	0.14	193
210	0.55	0.39	0.45	220
211	0.44	0.20	0.28	140
212	0.15	0.09	0.11	161
213	0.45	0.53	0.49	72
214	0.60	0.42	0.50	396
215	0.67	0.42	0.51	134
216	0.46	0.26	0.34	400
217	0.32	0.25	0.28	75
218	0.93	0.77	0.84	219
219	0.58	0.42	0.49	210
220	0.84	0.67	0.75	298
221	0.89	0.70	0.79	266

222	0.66	0.45	0.53	290
223	0.12	0.05	0.07	128
224	0.69	0.48	0.57	159
225	0.41	0.35	0.38	164
226	0.47	0.34	0.39	144
227	0.54	0.40	0.46	276
228	0.09	0.04	0.06	235
229	0.14	0.06	0.09	216
230	0.32	0.20	0.25	228
231	0.63	0.53	0.58	64
232	0.23	0.16	0.18	103
233	0.61	0.39	0.48	216
234	0.50	0.23	0.32	116
235	0.45	0.32	0.38	77
236	0.88	0.69	0.77	67
237	0.29	0.18	0.22	218
238	0.26	0.18	0.21	139
239	0.22	0.06	0.10	94
240	0.38	0.31	0.34	77
241	0.31	0.13	0.19	167
242	0.63	0.42	0.50	86
243	0.31	0.24	0.27	58
244	0.52	0.40	0.45	269
245	0.12	0.08	0.10	112
246	0.92	0.82	0.86	255
247	0.22	0.22	0.22	58
248	0.13	0.07	0.10	81
249	0.05	0.02	0.03	131
250	0.40	0.26	0.31	93
251	0.57	0.34	0.43	154
252	0.10	0.05	0.06	129
253	0.47	0.34	0.39	83
254	0.24	0.13	0.17	191
255	0.11	0.06	0.08	219
256	0.14	0.08	0.10	130
257	0.38	0.31	0.34	93
258	0.63	0.53	0.57	217
259	0.27	0.18	0.21	141
260	0.67	0.24	0.35	143
261	0.40	0.18	0.25	219
262	0.48	0.36	0.41	107
263	0.37	0.24	0.29	236
264	0.27	0.21	0.23	119
265	0.43	0.28	0.34	72
266	0.10	0.06	0.07	70
267	0.35	0.24	0.29	107
268	0.53	0.47	0.50	169
269	0.29	0.17	0.22	129
270	0.69	0.54	0.61	159
271	0.77	0.53	0.63	190
272	0.46	0.34	0.39	248
273	0.84	0.75	0.79	264
274	0.83	0.67	0.74	105
275	0.20	0.12	0.15	104
276	0.07	0.03	0.05	115
277	0.77	0.61	0.68	170
278	0.71	0.48	0.57	145
279	0.88	0.75	0.81	230

280	0.58	0.40	0.47	80
281	0.65	0.55	0.59	217
282	0.69	0.53	0.60	175
283	0.26	0.17	0.21	269
284	0.53	0.39	0.45	74
285	0.69	0.51	0.59	206
286	0.83	0.70	0.76	227
287	0.65	0.42	0.51	130
288	0.16	0.08	0.11	129
289	0.16	0.11	0.13	80
290	0.19	0.14	0.16	99
291	0.59	0.39	0.47	208
292	0.28	0.15	0.19	67
293	0.76	0.54	0.63	109
294	0.34	0.26	0.30	140
295	0.24	0.17	0.20	241
296	0.23	0.15	0.18	72
297	0.22	0.15	0.18	107
298	0.62	0.57	0.60	61
299	0.73	0.56	0.63	77
300	0.15	0.12	0.13	111
301	0.03	0.01	0.01	126
302	0.16	0.11	0.13	73
303	0.55	0.44	0.49	176
304	0.89	0.81	0.85	230
305	0.83	0.71	0.76	156
306	0.43	0.37	0.40	146
307	0.22	0.11	0.15	98
308	0.04	0.01	0.02	78
309	0.46	0.17	0.25	94
310	0.59	0.38	0.46	162
311	0.71	0.50	0.59	116
312	0.45	0.35	0.40	57
313	0.35	0.11	0.16	65
314	0.41	0.36	0.38	138
315	0.50	0.29	0.37	195
316	0.39	0.32	0.35	69
317	0.26	0.21	0.23	134
318	0.54	0.41	0.47	148
319	0.81	0.56	0.66	161
320	0.18	0.18	0.18	104
321	0.69	0.62	0.65	156
322	0.55	0.46	0.50	134
323	0.53	0.45	0.49	232
324	0.21	0.16	0.18	92
325	0.37	0.23	0.29	197
326	0.10	0.07	0.08	126
327	0.17	0.08	0.11	115
328	0.94	0.71	0.81	198
329	0.43	0.30	0.35	125
330	0.53	0.26	0.35	81
331	0.33	0.15	0.21	94
332	0.29	0.20	0.23	56
333	0.14	0.08	0.10	260
334	0.16	0.12	0.13	60
335	0.25	0.12	0.16	110
336	0.59	0.54	0.56	71
337	0.13	0.09	0.10	66

337	0.12	0.09	0.10	66
338	0.39	0.45	0.42	150
339	0.05	0.04	0.04	54
340	0.79	0.57	0.66	195
341	0.68	0.51	0.58	79
342	0.37	0.50	0.43	38
343	0.55	0.40	0.46	43
344	0.33	0.28	0.30	68
345	0.64	0.34	0.45	73
346	0.10	0.07	0.08	116
347	0.61	0.49	0.54	111
348	0.24	0.19	0.21	63
349	0.83	0.71	0.77	104
350	0.57	0.57	0.57	44
351	0.25	0.28	0.26	40
352	0.76	0.62	0.68	136
353	0.40	0.22	0.29	54
354	0.24	0.12	0.16	134
355	0.51	0.42	0.46	120
356	0.43	0.31	0.36	228
357	0.55	0.42	0.48	269
358	0.56	0.36	0.44	80
359	0.75	0.63	0.68	140
360	0.30	0.19	0.23	125
361	0.87	0.73	0.80	169
362	0.17	0.12	0.14	56
363	0.83	0.77	0.80	154
364	0.19	0.19	0.19	58
365	0.22	0.15	0.18	71
366	0.90	0.67	0.77	54
367	0.14	0.10	0.12	116
368	0.26	0.19	0.22	54
369	0.09	0.06	0.07	71
370	0.23	0.11	0.15	61
371	0.29	0.10	0.15	71
372	0.52	0.44	0.48	52
373	0.59	0.47	0.52	150
374	0.26	0.22	0.23	93
375	0.12	0.09	0.10	67
376	0.07	0.03	0.04	76
377	0.45	0.37	0.41	106
378	0.07	0.02	0.04	86
379	0.23	0.21	0.22	14
380	0.76	0.56	0.64	122
381	0.10	0.06	0.07	104
382	0.25	0.15	0.19	66
383	0.49	0.41	0.45	110
384	0.16	0.07	0.10	155
385	0.40	0.36	0.38	50
386	0.21	0.12	0.16	64
387	0.26	0.12	0.16	93
388	0.48	0.33	0.39	102
389	0.13	0.06	0.08	108
390	0.90	0.68	0.78	178
391	0.36	0.21	0.26	115
392	0.66	0.45	0.54	42
393	0.04	0.01	0.01	134
394	0.29	0.16	0.21	112

395	0.37	0.30	0.33	176
396	0.30	0.17	0.22	125
397	0.64	0.45	0.52	224
398	0.75	0.67	0.71	63
399	0.11	0.07	0.09	59
400	0.44	0.41	0.43	63
401	0.42	0.32	0.36	98
402	0.45	0.24	0.31	162
403	0.24	0.18	0.21	83
404	0.64	0.84	0.73	19
405	0.21	0.15	0.18	92
406	0.48	0.39	0.43	41
407	0.50	0.33	0.39	43
408	0.69	0.49	0.57	160
409	0.12	0.08	0.10	50
410	0.00	0.00	0.00	19
411	0.30	0.22	0.25	175
412	0.22	0.15	0.18	72
413	0.21	0.09	0.13	95
414	0.24	0.15	0.19	97
415	0.15	0.10	0.12	48
416	0.43	0.34	0.38	83
417	0.23	0.15	0.18	40
418	0.24	0.13	0.17	91
419	0.52	0.42	0.47	90
420	0.28	0.24	0.26	37
421	0.07	0.05	0.06	66
422	0.46	0.40	0.43	73
423	0.37	0.29	0.32	56
424	0.85	0.88	0.87	33
425	0.10	0.04	0.06	76
426	0.06	0.02	0.04	81
427	0.92	0.73	0.81	150
428	1.00	0.76	0.86	29
429	0.98	0.95	0.97	389
430	0.56	0.44	0.49	167
431	0.45	0.15	0.22	123
432	0.26	0.18	0.21	39
433	0.31	0.28	0.29	82
434	0.90	0.71	0.80	66
435	0.56	0.47	0.51	93
436	0.49	0.38	0.43	87
437	0.16	0.09	0.12	86
438	0.64	0.49	0.55	104
439	0.46	0.21	0.29	100
440	0.17	0.07	0.10	141
441	0.40	0.43	0.41	110
442	0.24	0.20	0.22	123
443	0.29	0.21	0.25	71
444	0.22	0.12	0.15	109
445	0.42	0.35	0.39	48
446	0.41	0.28	0.33	76
447	0.23	0.26	0.24	38
448	0.59	0.56	0.57	81
449	0.44	0.28	0.34	132
450	0.41	0.33	0.37	81
451	0.67	0.38	0.49	76
452	0.11	0.07	0.08	44

453	0.00	0.00	0.00	44
454	0.75	0.54	0.63	70
455	0.29	0.25	0.27	155
456	0.31	0.26	0.28	43
457	0.37	0.31	0.33	72
458	0.20	0.18	0.19	62
459	0.42	0.32	0.36	69
460	0.14	0.08	0.10	119
461	0.60	0.34	0.44	79
462	0.30	0.26	0.28	47
463	0.34	0.26	0.29	104
464	0.56	0.42	0.48	106
465	0.35	0.28	0.31	64
466	0.44	0.33	0.38	173
467	0.60	0.44	0.51	107
468	0.42	0.29	0.35	126
469	0.17	0.06	0.09	114
470	0.93	0.81	0.87	140
471	0.58	0.38	0.46	79
472	0.41	0.41	0.41	143
473	0.64	0.39	0.49	158
474	0.28	0.12	0.16	138
475	0.20	0.15	0.17	59
476	0.63	0.45	0.53	88
477	0.73	0.65	0.69	176
478	0.90	0.79	0.84	24
479	0.28	0.17	0.21	92
480	0.68	0.58	0.63	100
481	0.37	0.36	0.36	103
482	0.26	0.15	0.19	74
483	0.71	0.59	0.65	105
484	0.18	0.07	0.10	83
485	0.05	0.04	0.04	82
486	0.30	0.18	0.23	71
487	0.38	0.23	0.28	120
488	0.23	0.10	0.13	105
489	0.54	0.39	0.45	87
490	0.90	0.84	0.87	32
491	0.05	0.03	0.04	69
492	0.14	0.06	0.09	49
493	0.06	0.04	0.05	117
494	0.49	0.38	0.43	61
495	0.95	0.80	0.87	344
496	0.19	0.12	0.14	52
497	0.49	0.34	0.40	137
498	0.33	0.15	0.21	98
499	0.31	0.23	0.26	79
micro avg	0.57	0.41	0.48	173812
macro avg	0.45	0.33	0.38	173812
weighted avg	0.55	0.41	0.47	173812
samples avg	0.44	0.39	0.38	173812

```
Time taken to run this cell : 4:36:05.316346
```

5. Assignments

Perform hyperparam tuning on alpha (or lambda) for Logistic r performance using GridSearch

For this code as been taking long we are going to reduce dimension size to .1 M, n_gram to (1,2) and r

```
#https://stackoverflow.com/questions/12632992/gridsearch-for-an-estimator-inside-a-onevsre
#https://stackoverflow.com/questions/50290273/gridsearchcv-representation-of-each-class-ir

#https://datascience.stackexchange.com/questions/41680/how-to-implement-gridsearchcv-for-c



from sklearn.multiclass import OneVsRestClassifier
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import f1_score

classifier_3 = OneVsRestClassifier(LogisticRegression(penalty='l1',class_weight = 'balance

parameters = {
    "estimator__C": [.01,.1,1,10]
#    "C": [.001,.01]

}


model_tunning = GridSearchCV(classifier_3, param_grid=parameters,scoring='f1_micro',refit=


model_tunning.fit(x_train_multilabel, y_train)
```

```
GridSearchCV(cv='warn', error_score='raise-deprecating',
             estimator=OneVsRestClassifier(estimator=LogisticRegression(C=1.0,
                                                                        class_weight=
                                                                        dual=False,
                                                                        fit_intercept
                                                                        intercept_sca
                                                                        l1_ratio=None
                                                                        max_iter=100,
                                                                        multi_class='
                                                                        n_jobs=None,
                                                                        penalty='l1',
                                                                        random_state=
                                                                        solver='warn'
                                                                        tol=0.0001,
                                                                        verbose=0,
                                                                        warm_start=Fa
                                                   n_jobs=None),
             iid='warn', n_jobs=None,
             param_grid={'estimator__C': [0.01, 0.1, 1, 10]},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring='f1_micro', verbose=0)
```

```
print (model_tunning.best_estimator_)
print (model_tunning.best_params_)
```

```
OneVsRestClassifier(estimator=LogisticRegression(C=1, class_weight='balanced',
                                                 dual=False, fit_intercept=True,
                                                 intercept_scaling=1,
                                                 l1_ratio=None, max_iter=100,
                                                 multi_class='warn',
                                                 n_jobs=None, penalty='l1',
                                                 random_state=None,
                                                 solver='warn', tol=0.0001,
                                                 verbose=0, warm_start=False),
                    n_jobs=None)
{'estimator__C': 1}
```

For we see best estimator__C value is 1 so now we can train with that

```
classifier_final = OneVsRestClassifier(LogisticRegression(penalty='l1',class_weight = 'bal
```

```
classifier_final.fit(x_train_multilabel, y_train)
predictions_final = classifier_final.predict(x_test_multilabel)
```

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

```
precision = precision_score(y_test, predictions_final, average='micro')
recall = recall_score(y_test, predictions_final, average='micro')
f1 = f1_score(y_test, predictions_final, 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_final, average='macro')
recall = recall_score(y_test, predictions_final, average='macro')
f1 = f1_score(y_test, predictions_final, average='macro')

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

print (metrics.classification_report(y_test, predictions_final))
```

```
Micro-average quality numbers
Precision: 0.3319, Recall: 0.4992, F1-measure: 0.3987
Macro-average quality numbers
Precision: 0.2709, Recall: 0.4168, F1-measure: 0.3133
              precision    recall  f1-score   support

           0       0.49      0.49      0.49       820
           1       0.36      0.52      0.43      1931
           2       0.15      0.28      0.19       544
           3       0.23      0.29      0.25       222
           4       0.48      0.66      0.56      1311
           5       0.49      0.61      0.55      1014
           6       0.43      0.56      0.49      1374
           7       0.43      0.70      0.53       702
           8       0.66      0.72      0.69      1424
           9       0.71      0.81      0.76      1037
          10       0.50      0.70      0.58       797
          11       0.27      0.51      0.35       156
          12       0.27      0.53      0.36        36
          13       0.42      0.57      0.48       610
          14       0.29      0.41      0.34       405
          15       0.25      0.32      0.28       144
          16       0.27      0.45      0.34       425
          17       0.32      0.48      0.39       485
          18       0.46      0.78      0.58       269
          19       0.58      0.70      0.63       518
          20       0.38      0.52      0.44       529
          21       0.60      0.65      0.62       294
          22       0.62      0.55      0.58       520
          23       0.25      0.48      0.33       246
          24       0.26      0.49      0.34       312
          25       0.31      0.46      0.37       314
          26       0.18      0.39      0.24       190
          27       0.16      0.26      0.20       342
          28       0.17      0.31      0.22        96
          29       0.10      0.25      0.15        32
          30       0.39      0.65      0.48       747
          31       0.09      0.29      0.14        14
          32       0.42      0.72      0.53       166
          33       0.35      0.47      0.40       171
          34       0.37      0.46      0.41       256
          35       0.55      0.65      0.60       199
          36       0.06      0.18      0.10        60
          37       0.18      0.40      0.25       203
          38       0.48      0.59      0.53       201
          39       0.27      0.46      0.34       208
          40       0.04      0.15      0.06        13
          41       0.16      0.31      0.21       154
          42       0.21      0.43      0.28        69
          43       0.28      0.54      0.37       426
          44       0.22      0.49      0.31        77
          45       0.32      0.54      0.40       223
          46       0.27      0.50      0.35       144
          47       0.54      0.58      0.56       245
          48       0.19      0.30      0.23        91
          49       0.35      0.46      0.40       157
          50       0.72      0.73      0.73       132
```

51	0.68	0.78	0.73	41
52	0.38	0.58	0.46	124
53	0.13	0.32	0.19	96
54	0.09	0.29	0.14	128
55	0.26	0.43	0.32	46
56	0.53	0.72	0.61	151
57	0.13	0.14	0.13	80
58	0.16	0.29	0.21	65
59	0.25	0.30	0.27	182
60	0.59	0.73	0.65	148
61	0.29	0.28	0.28	196
62	0.12	0.31	0.17	58
63	0.28	0.35	0.31	43
64	0.30	0.36	0.32	197
65	0.39	0.55	0.46	82
66	0.36	0.66	0.47	50
67	0.33	0.66	0.44	105
68	0.13	0.17	0.15	98
69	0.21	0.21	0.21	238
70	0.09	0.09	0.09	35
71	0.33	0.67	0.44	54
72	0.16	0.24	0.19	25
73	0.20	0.52	0.29	29
74	0.06	0.24	0.10	29
75	0.12	0.33	0.18	40
76	0.42	0.65	0.51	105
77	0.38	0.68	0.49	28
78	0.15	0.25	0.19	202
79	0.40	0.70	0.51	37
80	0.15	0.33	0.20	15
81	0.28	0.50	0.36	52
82	0.17	0.36	0.23	50
83	0.06	0.11	0.08	56
84	0.56	0.63	0.59	54
85	0.39	0.68	0.49	34
86	0.09	0.20	0.13	30
87	0.35	0.59	0.44	29
88	0.34	0.88	0.49	24
89	0.53	0.88	0.66	117
90	0.11	0.26	0.15	66
91	0.16	0.34	0.21	68
92	0.22	0.39	0.28	67
93	0.31	0.57	0.40	28
94	0.38	0.47	0.42	17
95	0.42	0.55	0.47	51
96	0.34	0.57	0.42	53
97	0.11	0.16	0.13	61
98	0.05	0.13	0.07	79
99	0.23	0.44	0.30	18
100	0.14	0.18	0.16	11
101	0.44	0.65	0.52	207
102	0.00	0.00	0.00	6
103	0.09	0.10	0.10	30
104	0.14	0.24	0.18	54
105	0.42	0.54	0.47	39
106	0.18	0.29	0.22	70
107	0.09	0.21	0.12	14
108	0.23	0.32	0.27	66

109	0.27	0.42	0.33	50
110	0.16	0.33	0.21	87
111	0.37	0.55	0.44	51
112	0.42	0.03	0.05	291
113	0.80	0.82	0.81	49
114	0.27	0.30	0.28	110
115	0.03	0.04	0.03	28
116	0.05	0.20	0.07	5
117	0.10	0.23	0.14	56
118	0.57	0.54	0.55	125
119	0.47	0.64	0.54	44
120	0.30	0.45	0.36	42
121	0.24	0.27	0.25	55
122	0.61	0.63	0.62	68
123	0.06	0.16	0.09	82
124	0.00	0.00	0.00	0
125	0.19	0.71	0.29	7
126	0.09	0.22	0.13	18
127	0.16	0.26	0.20	31
128	0.58	0.54	0.56	13
129	0.48	0.60	0.53	50
130	0.13	0.25	0.17	91
131	0.49	0.77	0.60	35
132	0.08	0.23	0.12	26
133	0.11	0.19	0.14	32
134	0.34	0.49	0.40	35
135	0.56	0.78	0.65	37
136	0.00	0.00	0.00	55
137	0.19	0.63	0.29	41
138	0.10	0.40	0.16	15
139	0.18	0.38	0.25	99
140	0.61	0.71	0.66	86
141	0.31	0.49	0.38	53
142	0.25	0.25	0.25	36
143	0.41	0.68	0.51	66
144	0.34	0.61	0.44	64
145	0.11	0.24	0.15	25
146	0.14	0.31	0.20	125
147	0.12	0.53	0.20	15
148	0.48	0.56	0.52	48
149	0.19	0.45	0.27	65
150	0.16	0.27	0.20	11
151	0.31	0.67	0.43	15
152	0.11	0.25	0.15	52
153	0.21	0.44	0.29	18
154	0.50	0.50	0.50	16
155	0.07	0.30	0.12	20
156	0.31	0.36	0.34	121
157	0.44	0.60	0.50	107
158	0.07	0.20	0.11	15
159	0.45	0.54	0.49	105
160	0.29	0.46	0.36	69
161	0.34	0.50	0.41	56
162	0.14	0.23	0.17	47
163	0.09	0.13	0.10	121
164	0.31	0.51	0.39	41
165	0.48	0.06	0.10	229
166	0.22	0.19	0.20	98

166	0.22	0.19	0.20	98
167	0.16	0.30	0.21	33
168	0.28	0.30	0.29	44
169	0.53	0.51	0.52	45
170	0.86	0.49	0.62	51
171	0.00	0.00	0.00	18
172	0.33	0.60	0.43	48
173	0.14	0.50	0.21	12
174	0.20	0.42	0.27	62
175	0.69	0.80	0.74	44
176	0.65	0.87	0.74	30
177	0.29	0.47	0.36	30
178	0.00	0.00	0.00	0
179	0.25	1.00	0.40	1
180	0.40	0.42	0.41	40
181	0.09	0.16	0.11	44
182	0.33	1.00	0.50	2
183	0.34	0.57	0.43	75
184	0.06	0.50	0.11	4
185	0.40	0.42	0.41	64
186	0.19	0.58	0.29	12
187	0.79	0.69	0.74	55
188	0.51	0.70	0.59	64
189	0.24	0.39	0.30	96
190	0.03	0.09	0.05	22
191	0.51	0.28	0.36	76
192	0.19	0.64	0.29	45
193	0.33	0.57	0.42	14
194	0.36	0.66	0.46	50
195	0.32	0.55	0.41	20
196	0.52	0.66	0.58	35
197	0.35	0.63	0.45	94
198	0.05	0.29	0.09	14
199	0.05	0.08	0.06	25
200	0.20	0.20	0.20	54
201	0.11	0.18	0.14	22
202	0.12	0.28	0.17	43
203	0.09	0.16	0.11	43
204	0.85	0.73	0.78	62
205	0.02	0.33	0.04	3
206	0.16	0.42	0.23	43
207	0.12	0.29	0.17	7
208	0.03	0.12	0.05	8
209	0.19	0.24	0.21	42
210	0.18	0.50	0.26	10
211	0.13	0.35	0.19	40
212	0.36	0.57	0.44	23
213	0.00	0.00	0.00	6
214	0.39	0.62	0.48	47
215	0.14	0.16	0.15	62
216	0.43	0.51	0.46	77
217	0.21	0.18	0.20	22
218	0.04	0.33	0.07	3
219	0.03	0.07	0.04	28
220	0.27	0.21	0.24	81
221	0.16	0.45	0.24	31
222	0.14	0.15	0.14	34
223	0.60	0.45	0.51	60

224	0.26	0.70	0.38	10
225	0.43	0.60	0.50	10
226	0.64	0.90	0.75	92
227	0.37	0.77	0.50	13
228	0.10	0.23	0.14	13
229	0.67	0.84	0.74	43
230	0.23	0.34	0.27	35
231	0.05	0.25	0.09	4
232	0.28	0.35	0.31	20
233	0.36	0.63	0.46	145
234	0.55	0.62	0.58	55
235	0.00	0.00	0.00	2
236	0.17	0.24	0.20	37
237	0.62	0.54	0.58	90
238	0.23	0.21	0.22	58
239	0.24	0.45	0.32	20
240	0.74	0.74	0.74	61
241	0.66	0.74	0.70	42
242	0.52	0.90	0.66	30
243	0.64	0.67	0.65	66
244	0.50	0.36	0.42	42
245	0.04	0.06	0.05	31
246	0.67	0.67	0.67	6
247	0.13	0.22	0.16	18
248	0.78	0.63	0.70	51
249	0.13	0.41	0.20	17
250	0.34	0.68	0.45	22
251	0.46	0.60	0.52	52
252	0.32	0.28	0.30	29
253	0.06	0.18	0.10	28
254	0.04	0.10	0.06	10
255	0.04	0.40	0.07	5
256	0.20	0.67	0.31	3
257	0.32	0.51	0.40	41
258	0.19	0.20	0.19	30
259	0.20	0.67	0.31	3
260	0.02	0.03	0.02	38
261	0.00	0.00	0.00	1
262	0.30	0.42	0.35	19
263	0.04	0.14	0.07	14
264	0.06	0.14	0.09	37
265	0.03	0.11	0.05	9
266	0.14	0.44	0.21	45
267	0.51	0.61	0.56	33
268	0.33	0.88	0.48	16
269	0.37	0.66	0.47	35
270	0.14	0.27	0.19	11
271	0.02	0.03	0.03	30
272	0.15	0.62	0.24	8
273	0.14	0.38	0.20	21
274	0.27	0.48	0.34	123
275	0.28	0.49	0.35	67
276	0.56	0.90	0.69	20
277	0.00	0.00	0.00	14
278	0.12	0.26	0.16	19
279	0.48	0.83	0.61	12
280	0.13	0.20	0.16	15
281	0.71	0.71	0.71	17

282	0.67	0.85	0.75	41
283	0.29	0.60	0.39	15
284	0.34	0.42	0.38	74
285	0.28	0.32	0.30	38
286	0.04	0.12	0.06	16
287	0.03	0.13	0.05	30
288	0.54	0.79	0.64	28
289	0.07	0.19	0.10	21
290	0.82	0.66	0.73	41
291	0.12	0.33	0.17	12
292	0.26	0.46	0.33	24
293	0.32	0.60	0.41	20
294	0.10	0.39	0.16	23
295	0.07	0.14	0.09	29
296	0.12	0.43	0.19	28
297	0.10	0.36	0.15	42
298	0.09	0.17	0.12	53
299	0.10	0.22	0.14	36
300	0.34	0.34	0.34	41
301	0.33	0.73	0.46	37
302	0.55	0.65	0.60	26
303	0.17	0.45	0.25	11
304	0.16	0.35	0.22	31
305	0.15	0.35	0.21	17
306	0.10	0.22	0.14	9
307	0.11	0.33	0.16	6
308	0.02	0.06	0.03	34
309	0.49	0.51	0.50	43
310	0.05	0.10	0.07	30
311	0.30	0.36	0.33	50
312	0.05	0.17	0.08	24
313	0.14	0.29	0.19	42
314	0.21	0.55	0.30	22
315	0.18	0.16	0.17	58
316	0.25	0.10	0.14	10
317	0.28	0.47	0.36	57
318	0.28	0.50	0.36	10
319	0.02	0.09	0.03	11
320	0.09	0.45	0.15	11
321	0.19	0.50	0.28	8
322	0.35	0.41	0.38	22
323	0.65	0.71	0.68	28
324	0.50	0.60	0.55	50
325	0.10	0.22	0.14	18
326	0.15	0.27	0.19	33
327	0.03	0.24	0.06	17
328	0.14	0.24	0.18	29
329	0.33	0.57	0.42	7
330	0.23	0.50	0.31	10
331	0.18	0.52	0.27	25
332	1.00	1.00	1.00	2
333	0.35	0.73	0.47	11
334	0.04	0.04	0.04	24
335	0.21	0.60	0.32	5
336	0.10	0.15	0.12	33
337	0.37	0.37	0.37	30
338	0.77	0.86	0.81	42
339	0.13	0.23	0.17	26

339	0.13	0.25	0.17	26
340	0.35	0.61	0.44	36
341	0.58	0.54	0.56	13
342	0.28	0.73	0.40	11
343	0.30	0.60	0.40	10
344	0.17	0.19	0.18	21
345	0.00	0.00	0.00	0
346	0.00	0.00	0.00	6
347	0.09	0.25	0.14	12
348	0.03	0.08	0.04	13
349	0.36	0.33	0.35	24
350	0.28	0.52	0.36	27
351	0.27	0.40	0.32	43
352	0.00	0.00	0.00	30
353	0.23	0.55	0.32	22
354	0.21	0.32	0.26	31
355	0.17	0.80	0.28	10
356	0.27	0.35	0.30	20
357	0.43	0.75	0.55	20
358	0.34	0.50	0.41	28
359	0.42	0.71	0.53	21
360	0.12	0.24	0.16	25
361	0.33	0.54	0.41	35
362	0.70	0.83	0.76	36
363	0.07	0.47	0.11	17
364	0.47	0.54	0.50	13
365	0.27	0.43	0.33	21
366	0.24	0.33	0.28	18
367	0.32	0.12	0.18	97
368	0.37	0.55	0.44	29
369	0.37	0.83	0.51	12
370	0.15	0.54	0.23	13
371	0.06	0.22	0.09	18
372	0.10	0.33	0.15	6
373	0.19	0.50	0.27	6
374	0.22	0.27	0.24	30
375	0.12	0.30	0.17	27
376	0.10	0.14	0.12	28
377	0.06	0.50	0.11	2
378	0.11	0.50	0.18	4
379	0.09	0.05	0.07	19
380	0.29	0.80	0.42	5
381	0.28	0.56	0.37	18
382	0.44	0.68	0.54	22
383	0.02	0.06	0.03	16
384	0.36	0.62	0.46	13
385	0.21	0.28	0.24	18
386	0.53	0.91	0.67	11
387	0.38	0.74	0.50	88
388	0.03	0.15	0.04	13
389	0.40	0.33	0.36	6
390	0.00	0.00	0.00	6
391	0.84	0.80	0.82	51
392	0.11	0.15	0.13	13
393	0.51	0.49	0.50	37
394	0.00	0.00	0.00	6
395	0.00	0.00	0.00	9
396	0.00	0.00	0.00	13

397	0.50	0.50	0.50	6
398	0.40	0.66	0.50	29
399	0.77	0.73	0.75	33
400	0.14	0.13	0.13	31
401	0.26	0.20	0.22	50
402	0.87	0.72	0.79	18
403	0.04	0.14	0.06	7
404	0.38	0.65	0.48	26
405	0.70	0.86	0.77	56
406	0.57	1.00	0.73	4
407	0.18	0.29	0.22	17
408	0.21	0.55	0.31	11
409	0.04	0.11	0.05	18
410	0.18	0.50	0.26	10
411	0.26	0.27	0.26	45
412	0.59	0.50	0.54	20
413	0.33	0.28	0.30	25
414	0.13	0.35	0.19	20
415	0.00	0.00	0.00	6
416	0.11	0.15	0.13	26
417	0.20	0.40	0.27	10
418	0.01	0.06	0.02	18
419	0.29	0.83	0.43	6
420	0.33	0.53	0.41	17
421	1.00	1.00	1.00	1
422	0.00	0.00	0.00	6
423	0.00	0.00	0.00	12
424	0.23	0.75	0.35	4
425	0.17	0.36	0.24	11
426	0.05	0.09	0.06	11
427	0.35	0.75	0.48	8
428	0.38	0.50	0.43	26
429	0.34	0.72	0.46	40
430	0.00	0.00	0.00	2
431	0.04	0.09	0.06	35
432	0.17	0.33	0.22	15
433	0.00	0.00	0.00	18
434	0.00	0.00	0.00	0
435	0.00	0.00	0.00	0
436	0.15	0.29	0.20	28
437	0.39	0.42	0.41	33
438	0.61	0.55	0.58	20
439	0.10	0.14	0.11	36
440	0.15	0.33	0.21	18
441	0.29	0.61	0.39	18
442	0.57	0.75	0.65	16
443	0.09	0.09	0.09	22
444	0.08	0.17	0.11	6
445	0.36	0.71	0.48	21
446	0.80	0.70	0.74	46
447	0.15	0.23	0.18	69
448	0.29	0.29	0.29	7
449	0.09	0.33	0.14	3
450	0.17	0.21	0.19	52
451	0.05	0.19	0.08	16
452	0.71	1.00	0.83	17
453	0.00	0.00	0.00	13
454	0.25	0.27	0.26	11

455	0.05	0.17	0.07	12
456	0.08	0.33	0.13	6
457	0.12	0.17	0.14	18
458	0.03	0.07	0.04	15
459	0.79	0.54	0.64	28
460	0.04	0.06	0.04	18
461	0.22	0.50	0.30	10
462	0.23	0.25	0.24	24
463	0.24	0.39	0.30	18
464	0.89	0.62	0.73	39
465	0.27	0.73	0.39	11
466	0.11	0.14	0.12	35
467	0.09	0.14	0.11	21
468	0.18	0.11	0.14	37
469	0.10	0.40	0.15	5
470	0.10	0.25	0.14	8
471	0.45	0.54	0.49	37
472	0.03	0.02	0.02	47
473	0.21	0.57	0.31	14
474	0.63	0.74	0.68	23
475	0.57	0.86	0.69	66
476	0.05	0.67	0.09	3
477	0.39	0.58	0.47	19
478	0.00	0.00	0.00	1
479	0.07	0.22	0.11	23
480	0.08	0.13	0.10	60
481	0.12	0.23	0.16	26
482	0.07	0.50	0.12	4
483	0.38	0.38	0.38	8
484	0.68	0.57	0.62	23
485	0.40	0.44	0.42	18
486	0.42	0.67	0.52	12
487	0.61	0.48	0.54	29
488	0.08	1.00	0.15	1
489	0.33	0.50	0.40	6
490	0.11	0.29	0.16	7
491	0.00	0.00	0.00	3
492	0.17	0.50	0.26	10
493	0.33	0.53	0.41	19
494	0.12	0.29	0.17	7
495	0.46	0.75	0.57	8
496	0.26	0.50	0.34	18
497	0.09	0.07	0.08	72
498	0.17	0.38	0.23	8
499	0.30	0.50	0.38	32
micro avg	0.33	0.50	0.40	37472
macro avg	0.27	0.42	0.31	37472
weighted avg	0.37	0.50	0.42	37472
samples avg	0.35	0.48	0.37	37472

Try OneVsRestClassifier with Linear-SVM (SGDClassifier with l

```
start = datetime.now()
classifier_4 = OneVsRestClassifier(SGDClassifier(loss='hinge', alpha=0.001, penalty='l1'))
classifier_4.fit(x_train_multilabel, y_train)
predictions_4 = classifier_4.predict (x_test_multilabel)


print("Accuracy :",metrics.accuracy_score(y_test, predictions_4))
print("Hamming loss ",metrics.hamming_loss(y_test,predictions_4))


precision_4 = precision_score(y_test, predictions_4, average='micro')
recall_4 = recall_score(y_test, predictions_4, average='micro')
f1_4 = f1_score(y_test, predictions_4, average='micro')

print("Micro-average quality numbers")
print("Precision: {:.4f}, Recall: {:.4f}, F1-measure: {:.4f}".format(precision_4, recall_4

precision_4_macro = precision_score(y_test, predictions_4, average='macro')
recall_4_macro = recall_score(y_test, predictions_4, average='macro')
f1_4_macro = f1_score(y_test, predictions_4, average='macro')

print("Macro-average quality numbers")
print("Precision: {:.4f}, Recall: {:.4f}, F1-measure: {:.4f}".format(precision_4_macro, re

print (metrics.classification_report(y_test, predictions_4))
print("Time taken to run this cell :", datetime.now() - start)
```

```
Accuracy : 0.19613
Hamming loss  0.00310292
Micro-average quality numbers
Precision: 0.6098, Recall: 0.2981, F1-measure: 0.4005
Macro-average quality numbers
Precision: 0.3210, Recall: 0.2243, F1-measure: 0.2458
              precision    recall  f1-score   support

           0       0.84      0.65      0.73      5519
           1       0.52      0.17      0.26      8190
           2       0.82      0.29      0.43      6529
           3       0.70      0.46      0.55      3231
           4       0.74      0.41      0.53      6430
           5       0.65      0.41      0.50      2879
           6       0.87      0.43      0.58      5086
           7       0.79      0.60      0.68      4533
           8       0.52      0.16      0.25      3000
           9       0.72      0.54      0.62      2765
          10       0.10      0.00      0.00      3051
          11       0.72      0.34      0.46      3009
          12       0.60      0.27      0.37      2630
          13       0.47      0.20      0.28      1426
          14       0.83      0.58      0.68      2548
          15       0.80      0.10      0.18      2371
          16       0.47      0.18      0.26       873
          17       0.74      0.69      0.71      2151
          18       0.62      0.19      0.29      2204
          19       0.45      0.49      0.47       831
          20       0.69      0.50      0.58      1860
          21       0.00      0.00      0.00      2023
          22       0.00      0.00      0.00      1513
          23       0.80      0.57      0.67      1207
          24       0.13      0.02      0.03       506
          25       0.59      0.35      0.44       425
          26       0.54      0.35      0.42       793
          27       0.58      0.31      0.41      1291
          28       0.64      0.40      0.49      1208
          29       0.00      0.00      0.00       406
          30       0.71      0.20      0.32       504
          31       0.05      0.00      0.01       732
          32       0.60      0.17      0.26       441
          33       0.16      0.00      0.01      1645
          34       0.60      0.24      0.35      1058
          35       0.74      0.51      0.60       946
          36       0.57      0.24      0.34       644
          37       0.61      0.83      0.70       136
          38       0.61      0.31      0.41       570
          39       0.75      0.36      0.49       766
          40       0.73      0.01      0.02      1132
          41       0.33      0.26      0.29       174
          42       0.56      0.50      0.53       210
          43       0.56      0.54      0.55       433
          44       0.49      0.53      0.50       626
          45       0.54      0.27      0.36       852
          46       0.50      0.42      0.46       534
          47       0.12      0.01      0.02       350
          48       0.50      0.49      0.49       496
```

49	0.77	0.55	0.64	785
50	0.04	0.01	0.01	475
51	0.00	0.00	0.00	305
52	0.02	0.00	0.01	251
53	0.46	0.44	0.45	914
54	0.32	0.18	0.23	728
55	0.00	0.00	0.00	258
56	0.00	0.00	0.00	821
57	0.20	0.02	0.03	541
58	0.66	0.30	0.42	748
59	0.86	0.71	0.78	724
60	0.04	0.01	0.01	660
61	0.75	0.26	0.39	235
62	0.80	0.81	0.80	718
63	0.72	0.65	0.68	468
64	0.47	0.43	0.45	191
65	0.00	0.00	0.00	429
66	0.00	0.00	0.00	415
67	0.66	0.64	0.65	274
68	0.68	0.65	0.67	510
69	0.48	0.61	0.54	466
70	0.12	0.14	0.13	305
71	0.00	0.00	0.00	247
72	0.51	0.52	0.51	401
73	0.75	0.88	0.81	86
74	0.50	0.42	0.45	120
75	0.77	0.78	0.77	129
76	0.00	0.00	0.00	473
77	0.18	0.37	0.24	143
78	0.71	0.51	0.59	347
79	0.65	0.24	0.35	479
80	0.25	0.43	0.32	279
81	0.67	0.19	0.30	461
82	0.00	0.00	0.00	298
83	0.64	0.47	0.54	396
84	0.00	0.00	0.00	184
85	0.63	0.22	0.33	573
86	0.64	0.02	0.04	325
87	0.09	0.06	0.07	273
88	0.00	0.00	0.00	135
89	0.12	0.20	0.15	232
90	0.48	0.22	0.30	409
91	0.64	0.14	0.23	420
92	0.67	0.58	0.62	408
93	0.44	0.50	0.47	241
94	0.00	0.00	0.00	211
95	0.00	0.00	0.00	277
96	0.17	0.01	0.02	410
97	0.77	0.25	0.38	501
98	0.54	0.67	0.60	136
99	0.50	0.26	0.34	239
100	0.00	0.00	0.00	324
101	0.92	0.63	0.75	277
102	0.85	0.74	0.79	613
103	0.00	0.00	0.00	157
104	0.00	0.00	0.00	295
105	0.59	0.40	0.48	334
106	0.00	0.00	0.00	335

107	0.71	0.41	0.52	389
108	0.00	0.00	0.00	251
109	0.44	0.48	0.46	317
110	0.00	0.00	0.00	187
111	0.42	0.06	0.10	140
112	0.61	0.18	0.28	154
113	0.41	0.33	0.37	332
114	0.00	0.00	0.00	323
115	0.41	0.05	0.09	344
116	0.51	0.49	0.50	370
117	0.35	0.33	0.34	313
118	0.64	0.63	0.64	874
119	0.35	0.27	0.30	293
120	0.00	0.00	0.00	200
121	0.60	0.59	0.60	463
122	0.00	0.00	0.00	119
123	0.00	0.00	0.00	256
124	0.82	0.85	0.83	195
125	0.14	0.22	0.18	138
126	0.72	0.49	0.58	376
127	0.00	0.00	0.00	122
128	0.00	0.00	0.00	252
129	0.12	0.05	0.07	144
130	0.59	0.13	0.21	150
131	0.00	0.00	0.00	210
132	0.88	0.02	0.04	361
133	0.70	0.59	0.64	453
134	0.67	0.89	0.76	124
135	0.00	0.00	0.00	91
136	0.67	0.14	0.23	128
137	0.40	0.35	0.37	218
138	0.00	0.00	0.00	243
139	0.00	0.00	0.00	149
140	0.63	0.47	0.54	318
141	0.00	0.00	0.00	159
142	0.56	0.34	0.42	274
143	0.62	0.70	0.66	362
144	0.26	0.31	0.28	118
145	0.38	0.41	0.40	164
146	0.42	0.35	0.39	461
147	0.57	0.60	0.59	159
148	0.18	0.18	0.18	166
149	0.94	0.57	0.71	346
150	0.00	0.00	0.00	350
151	0.37	0.60	0.46	55
152	0.50	0.57	0.53	387
153	0.00	0.00	0.00	150
154	0.50	0.05	0.09	281
155	0.00	0.00	0.00	202
156	0.61	0.66	0.63	130
157	0.00	0.00	0.00	245
158	0.61	0.69	0.65	177
159	0.37	0.40	0.39	130
160	0.00	0.00	0.00	336
161	0.68	0.52	0.59	220
162	0.00	0.00	0.00	229
163	0.82	0.49	0.62	316
164	0.68	0.24	0.36	283

164	0.08	0.24	0.36	283
165	0.00	0.00	0.00	197
166	0.16	0.16	0.16	101
167	0.19	0.25	0.22	231
168	0.00	0.00	0.00	370
169	0.00	0.00	0.00	258
170	0.00	0.00	0.00	101
171	0.45	0.44	0.44	89
172	0.00	0.00	0.00	193
173	0.37	0.27	0.31	309
174	0.00	0.00	0.00	172
175	0.71	0.80	0.75	95
176	0.74	0.66	0.70	346
177	0.92	0.30	0.46	322
178	0.49	0.50	0.50	232
179	0.00	0.00	0.00	125
180	0.40	0.34	0.37	145
181	0.08	0.19	0.11	77
182	0.12	0.03	0.04	182
183	0.49	0.29	0.37	257
184	0.00	0.00	0.00	216
185	0.00	0.00	0.00	242
186	0.00	0.00	0.00	165
187	0.52	0.62	0.57	263
188	0.00	0.00	0.00	174
189	0.00	0.00	0.00	136
190	0.63	0.59	0.61	202
191	0.17	0.01	0.01	134
192	0.55	0.51	0.53	230
193	0.00	0.00	0.00	90
194	0.47	0.49	0.48	185
195	0.00	0.00	0.00	156
196	0.00	0.00	0.00	160
197	0.00	0.00	0.00	266
198	0.00	0.00	0.00	284
199	0.00	0.00	0.00	145
200	0.81	0.70	0.75	212
201	0.36	0.26	0.31	317
202	0.52	0.58	0.55	427
203	0.00	0.00	0.00	232
204	1.00	0.00	0.01	217
205	0.41	0.52	0.46	527
206	0.00	0.00	0.00	124
207	0.05	0.01	0.02	103
208	0.81	0.41	0.55	287
209	0.00	0.00	0.00	193
210	0.51	0.20	0.29	220
211	0.82	0.10	0.18	140
212	0.06	0.01	0.01	161
213	0.33	0.14	0.20	72
214	0.52	0.45	0.48	396
215	0.77	0.28	0.41	134
216	0.46	0.04	0.08	400
217	0.39	0.36	0.37	75
218	0.87	0.78	0.82	219
219	0.50	0.50	0.50	210
220	0.76	0.45	0.56	298
221	0.83	0.68	0.75	266

222	0.81	0.31	0.45	290
223	0.13	0.06	0.09	128
224	0.51	0.45	0.48	159
225	0.49	0.20	0.29	164
226	0.31	0.41	0.35	144
227	0.32	0.42	0.36	276
228	0.00	0.00	0.00	235
229	0.00	0.00	0.00	216
230	0.00	0.00	0.00	228
231	0.58	0.61	0.60	64
232	0.11	0.13	0.12	103
233	0.66	0.39	0.49	216
234	0.00	0.00	0.00	116
235	0.47	0.58	0.52	77
236	0.87	0.70	0.78	67
237	0.00	0.00	0.00	218
238	0.00	0.00	0.00	139
239	0.00	0.00	0.00	94
240	0.50	0.30	0.37	77
241	0.48	0.07	0.12	167
242	0.71	0.34	0.46	86
243	0.17	0.14	0.15	58
244	0.00	0.00	0.00	269
245	0.00	0.00	0.00	112
246	0.92	0.67	0.77	255
247	0.31	0.21	0.25	58
248	0.00	0.00	0.00	81
249	0.00	0.00	0.00	131
250	0.20	0.15	0.17	93
251	0.26	0.32	0.29	154
252	0.00	0.00	0.00	129
253	0.35	0.28	0.31	83
254	0.00	0.00	0.00	191
255	0.08	0.01	0.02	219
256	0.00	0.00	0.00	130
257	0.00	0.00	0.00	93
258	0.59	0.62	0.60	217
259	0.00	0.00	0.00	141
260	0.71	0.22	0.34	143
261	0.00	0.00	0.00	219
262	0.54	0.12	0.20	107
263	0.00	0.00	0.00	236
264	0.00	0.00	0.00	119
265	0.28	0.32	0.30	72
266	0.00	0.00	0.00	70
267	0.29	0.07	0.11	107
268	0.50	0.59	0.54	169
269	0.00	0.00	0.00	129
270	0.50	0.55	0.52	159
271	0.33	0.15	0.21	190
272	0.26	0.15	0.19	248
273	0.75	0.59	0.66	264
274	0.65	0.70	0.68	105
275	0.01	0.01	0.01	104
276	0.00	0.00	0.00	115
277	0.65	0.60	0.62	170
278	0.50	0.36	0.42	145
279	0.81	0.52	0.63	230

280	0.38	0.36	0.37	80
281	0.56	0.74	0.64	217
282	0.66	0.50	0.57	175
283	0.22	0.06	0.09	269
284	0.40	0.34	0.37	74
285	0.71	0.51	0.60	206
286	0.82	0.67	0.74	227
287	0.48	0.25	0.32	130
288	0.22	0.09	0.13	129
289	0.00	0.00	0.00	80
290	0.00	0.00	0.00	99
291	0.49	0.44	0.46	208
292	0.00	0.00	0.00	67
293	0.45	0.28	0.35	109
294	0.15	0.34	0.21	140
295	0.00	0.00	0.00	241
296	0.15	0.24	0.18	72
297	0.00	0.00	0.00	107
298	0.75	0.15	0.25	61
299	0.93	0.17	0.29	77
300	0.00	0.00	0.00	111
301	0.00	0.00	0.00	126
302	0.00	0.00	0.00	73
303	0.35	0.54	0.42	176
304	0.77	0.56	0.65	230
305	0.90	0.68	0.77	156
306	0.28	0.34	0.31	146
307	0.00	0.00	0.00	98
308	0.00	0.00	0.00	78
309	0.43	0.17	0.24	94
310	0.32	0.31	0.32	162
311	0.61	0.54	0.58	116
312	0.37	0.40	0.38	57
313	0.00	0.00	0.00	65
314	0.37	0.26	0.31	138
315	0.39	0.23	0.29	195
316	0.53	0.28	0.36	69
317	0.31	0.09	0.14	134
318	0.39	0.19	0.25	148
319	0.79	0.31	0.45	161
320	0.17	0.26	0.21	104
321	0.56	0.64	0.60	156
322	0.28	0.20	0.23	134
323	0.41	0.33	0.37	232
324	0.14	0.22	0.17	92
325	0.16	0.05	0.08	197
326	0.00	0.00	0.00	126
327	0.00	0.00	0.00	115
328	0.93	0.51	0.66	198
329	0.32	0.37	0.34	125
330	0.69	0.11	0.19	81
331	0.00	0.00	0.00	94
332	0.00	0.00	0.00	56
333	0.00	0.00	0.00	260
334	0.00	0.00	0.00	60
335	0.00	0.00	0.00	110
336	0.35	0.44	0.39	71
337	0.00	0.00	0.00	66

337	0.00	0.00	0.00	66
338	0.33	0.31	0.32	150
339	0.00	0.00	0.00	54
340	0.57	0.53	0.55	195
341	0.00	0.00	0.00	79
342	0.11	0.29	0.16	38
343	0.67	0.23	0.34	43
344	0.00	0.00	0.00	68
345	0.45	0.40	0.42	73
346	0.00	0.00	0.00	116
347	0.84	0.37	0.51	111
348	0.00	0.00	0.00	63
349	0.60	0.70	0.65	104
350	0.15	0.52	0.24	44
351	0.00	0.00	0.00	40
352	0.78	0.21	0.33	136
353	0.32	0.17	0.22	54
354	0.05	0.01	0.01	134
355	0.22	0.13	0.17	120
356	0.00	0.00	0.00	228
357	0.50	0.12	0.19	269
358	0.00	0.00	0.00	80
359	0.73	0.31	0.44	140
360	0.00	0.00	0.00	125
361	0.87	0.52	0.65	169
362	0.00	0.00	0.00	56
363	0.72	0.78	0.75	154
364	0.00	0.00	0.00	58
365	0.00	0.00	0.00	71
366	0.90	0.48	0.63	54
367	0.00	0.00	0.00	116
368	0.00	0.00	0.00	54
369	0.00	0.00	0.00	71
370	0.00	0.00	0.00	61
371	0.00	0.00	0.00	71
372	0.54	0.40	0.46	52
373	0.64	0.37	0.47	150
374	0.27	0.22	0.24	93
375	0.05	0.01	0.02	67
376	0.00	0.00	0.00	76
377	0.00	0.00	0.00	106
378	0.00	0.00	0.00	86
379	0.00	0.00	0.00	14
380	1.00	0.07	0.12	122
381	0.00	0.00	0.00	104
382	0.00	0.00	0.00	66
383	0.75	0.11	0.19	110
384	0.00	0.00	0.00	155
385	0.00	0.00	0.00	50
386	0.22	0.20	0.21	64
387	0.00	0.00	0.00	93
388	0.36	0.25	0.30	102
389	0.00	0.00	0.00	108
390	0.87	0.52	0.65	178
391	0.41	0.20	0.27	115
392	0.83	0.36	0.50	42
393	0.00	0.00	0.00	134
394	0.00	0.00	0.00	112

395	0.00	0.00	0.00	176
396	0.00	0.00	0.00	125
397	0.51	0.21	0.30	224
398	0.66	0.62	0.64	63
399	0.00	0.00	0.00	59
400	0.32	0.29	0.30	63
401	0.00	0.00	0.00	98
402	0.00	0.00	0.00	162
403	0.00	0.00	0.00	83
404	0.57	0.84	0.68	19
405	0.00	0.00	0.00	92
406	0.17	0.20	0.18	41
407	0.30	0.30	0.30	43
408	0.00	0.00	0.00	160
409	0.00	0.00	0.00	50
410	0.00	0.00	0.00	19
411	0.00	0.00	0.00	175
412	0.00	0.00	0.00	72
413	0.00	0.00	0.00	95
414	0.00	0.00	0.00	97
415	0.00	0.00	0.00	48
416	0.32	0.16	0.21	83
417	0.08	0.03	0.04	40
418	0.00	0.00	0.00	91
419	0.30	0.27	0.28	90
420	0.24	0.16	0.19	37
421	0.00	0.00	0.00	66
422	0.43	0.32	0.37	73
423	0.17	0.27	0.21	56
424	0.89	0.76	0.82	33
425	0.00	0.00	0.00	76
426	0.00	0.00	0.00	81
427	0.95	0.63	0.76	150
428	1.00	0.59	0.74	29
429	0.00	0.00	0.00	389
430	0.35	0.46	0.40	167
431	0.00	0.00	0.00	123
432	0.30	0.36	0.33	39
433	0.33	0.23	0.27	82
434	1.00	0.58	0.73	66
435	0.47	0.41	0.44	93
436	0.00	0.00	0.00	87
437	0.43	0.03	0.06	86
438	0.36	0.40	0.38	104
439	1.00	0.01	0.02	100
440	0.00	0.00	0.00	141
441	0.00	0.00	0.00	110
442	0.15	0.14	0.14	123
443	0.00	0.00	0.00	71
444	0.26	0.08	0.12	109
445	0.00	0.00	0.00	48
446	0.00	0.00	0.00	76
447	0.19	0.18	0.19	38
448	0.55	0.51	0.53	81
449	0.00	0.00	0.00	132
450	0.35	0.35	0.35	81
451	0.62	0.43	0.51	76
452	0.00	0.00	0.00	44

453	0.00	0.00	0.00	44
454	0.42	0.51	0.46	70
455	0.00	0.00	0.00	155
456	0.00	0.00	0.00	43
457	0.24	0.28	0.26	72
458	0.00	0.00	0.00	62
459	0.00	0.00	0.00	69
460	0.00	0.00	0.00	119
461	0.45	0.13	0.20	79
462	0.00	0.00	0.00	47
463	0.40	0.02	0.04	104
464	0.36	0.34	0.35	106
465	0.00	0.00	0.00	64
466	0.51	0.22	0.31	173
467	0.48	0.54	0.51	107
468	0.00	0.00	0.00	126
469	0.00	0.00	0.00	114
470	0.85	0.62	0.72	140
471	0.00	0.00	0.00	79
472	0.34	0.42	0.38	143
473	0.50	0.05	0.09	158
474	0.00	0.00	0.00	138
475	0.00	0.00	0.00	59
476	0.17	0.33	0.22	88
477	0.52	0.55	0.53	176
478	0.95	0.75	0.84	24
479	0.00	0.00	0.00	92
480	0.64	0.55	0.59	100
481	0.27	0.15	0.19	103
482	0.00	0.00	0.00	74
483	0.65	0.56	0.60	105
484	0.00	0.00	0.00	83
485	0.00	0.00	0.00	82
486	0.00	0.00	0.00	71
487	0.22	0.23	0.22	120
488	0.00	0.00	0.00	105
489	0.43	0.33	0.37	87
490	0.97	0.88	0.92	32
491	0.00	0.00	0.00	69
492	0.00	0.00	0.00	49
493	0.00	0.00	0.00	117
494	0.59	0.16	0.26	61
495	0.00	0.00	0.00	344
496	0.15	0.10	0.12	52
497	0.00	0.00	0.00	137
498	0.00	0.00	0.00	98
499	0.00	0.00	0.00	79
micro avg	0.61	0.30	0.40	173812
macro avg	0.32	0.22	0.25	173812
weighted avg	0.50	0.30	0.35	173812
samples avg	0.38	0.28	0.30	173812

```
Time taken to run this cell : 0:27:14.563231
```

```
from prettytable import PrettyTable
```

```
from prettytable import PrettyTable

x1 = PrettyTable()
x1.field_names = ["Algorithm","Hyper-Parameter", "Accuracy","Hammimg Loss"]
x1.add_row(["Logisitic regression using SGD", "alpha = .00001", ".09885",".00580972"])
x1.add_row(["Logisitic regression( no SGD,direct sklearn)", "", ".21231",".0031321"])
x1.add_row(["Logisitic regression", "C=1", " 0.0764",".0056422"])
x1.add_row(["Linear SVM ", "alpha =.001", " 0.19613",".00310292"])
x2=PrettyTable()
x2.field_names = ["Algorithm","Precision(Micro)","Recall(micro)","F1(micro)"]
x2.add_row(["Logisitic regression using SGD", ".2915",".4694","0.3597"])
x2.add_row(["Logisitic regression( no SGD,direct sklearn)",".2915",".5687",".4763"])
x2.add_row(["Logisitic regression", "0.3319",".4992",".3987"])
x2.add_row(["Linear SVM ", "0.6098","0.2981","0.4005"])
x3=PrettyTable()
x3.field_names = ["Algorithm","Precision(Macro)","Recall(Macro)","F1(Macro)"]
x3.add_row(["Logisitic regression using SGD",".2078",".4114",".2675"])
x3.add_row(["Logisitic regression( no SGD,direct sklearn)",".4512",".3347",".3807"])
x3.add_row(["Logisitic regression",".2709",".4168",".3133"])
x3.add_row(["Linear SVM ","0.3210","0.2243","0.2458"])
print (x1)
print (x2)
print (x3)
```

```
+-----------------------------------------------+-----------------+----------+--------
|                  Algorithm                    | Hyper-Parameter | Accuracy | Hammimg
+-----------------------------------------------+-----------------+----------+--------
|        Logisitic regression using SGD         |  alpha = .00001 |  .09885  |  .00580
| Logisitic regression( no SGD,direct sklearn)  |                 |  .21231  |   .0031
|             Logisitic regression              |       C=1       |  0.0764  |   .0056
|                 Linear SVM                    |   alpha =.001   |  0.19613 |  .00310
+-----------------------------------------------+-----------------+----------+--------
+-----------------------------------------------+------------------+---------------+--
|                  Algorithm                    | Precision(Micro) | Recall(micro) | F
+-----------------------------------------------+------------------+---------------+--
|        Logisitic regression using SGD         |      .2915       |     .4694     |
| Logisitic regression( no SGD,direct sklearn)  |      .2915       |     .5687     |
|             Logisitic regression              |      0.3319      |     .4992     |
|                 Linear SVM                    |      0.6098      |     0.2981    |
+-----------------------------------------------+------------------+---------------+--
+-----------------------------------------------+------------------+---------------+--
|                  Algorithm                    | Precision(Macro) | Recall(Macro) | F
+-----------------------------------------------+------------------+---------------+--
|        Logisitic regression using SGD         |      .2078       |     .4114     |
| Logisitic regression( no SGD,direct sklearn)  |      .4512       |     .3347     |
|             Logisitic regression              |      .2709       |     .4168     |
|                 Linear SVM                    |      0.3210      |     0.2243    |
+-----------------------------------------------+------------------+---------------+--
```

As per the assignment we are to consider Micro-F1 score as parameter to decide on the best algori

1)Logistic Regression seems to be the best

2)When performing hyperparameter tuning i used n_grams as bigrams (3rd column) and other first an larger number of datapoints (.5 million) so we clearly see it outperforms Logistic regression(bigram wi first two columns of logistic regression has F1 Micro as .3597 and .4763 where as in 3rd column we h