Quora Question Pairs

1. Business Problem

1.1 Description

Quora is a place to gain and share knowledge—about anything. It’s a platform to ask questions and connect with people who contribute unique insights and quality answers. This empowers people to learn from each other and to better understand the world.

Over 100 million people visit Quora every month, so it's no surprise that many people ask similarly worded questions. Multiple questions with the same intent can cause seekers to spend more time finding the best answer to their question, and make writers feel they need to answer multiple versions of the same question. Quora values canonical questions because they provide a better experience to active seekers and writers, and offer more value to both of these groups in the long term.

> Credits: Kaggle

Problem Statement

- Identify which questions asked on Quora are duplicates of questions that have already been asked.
- This could be useful to instantly provide answers to questions that have already been answered.
- We are tasked with predicting whether a pair of questions are duplicates or not.

1.2 Sources/Useful Links

- Source : https://www.kaggle.com/c/quora-question-pairs (https://www.kaggle.com/c/quora-question-pairs)

Useful Links

- Discussions : https://www.kaggle.com/anokas/data-analysis-xgboost-starter-0-35460-lb/comments (https://www.kaggle.com/anokas/data-analysis-xgboost-starter-0-35460-lb/comments)
- Kaggle Winning Solution and other approaches: https://www.dropbox.com/sh/93968nfnrzh8bp5/AACZdtsApc1QSTQc7X0H3QZ5a?dl=0 (https://www.dropbox.com/sh/93968nfnrzh8bp5/AACZdtsApc1QSTQc7X0H3QZ5a?dl=0)
- Blog 1 : https://engineering.quora.com/Semantic-Question-Matching-with-Deep-Learning (https://engineering.quora.com/Semantic-Question-Matching-with-Deep-Learning)
- Blog 2 : https://towardsdatascience.com/identifying-duplicate-questions-on-quora-top-12-on-kaggle-4c1cf93f1c30 (https://towardsdatascience.com/identifying-duplicate-questions-on-quora-top-12-on-kaggle-4c1cf93f1c30)

1.3 Real world/Business Objectives and Constraints

1. The cost of a mis-classification can be very high.
2. You would want a probability of a pair of questions to be duplicates so that you can choose any threshold of choice.
3. No strict latency concerns.
4. Interpretability is partially important.

2. Machine Learning Probelm

2.1 Data

2.1.1 Data Overview

- Data will be in a file Train.csv
- Train.csv contains 5 columns : qid1, qid2, question1, question2, is_duplicate
- Size of Train.csv - 60MB
- Number of rows in Train.csv = 404,290

2.1.2 Example Data point

```
"id","qid1","qid2","question1","question2","is_duplicate"
"0","1","2","What is the step by step guide to invest in share market in india?","W
hat is the step by step guide to invest in share market?","0"
"1","3","4","What is the story of Kohinoor (Koh-i-Noor) Diamond?","What would happe
n if the Indian government stole the Kohinoor (Koh-i-Noor) diamond back?","0"
"7","15","16","How can I be a good geologist?","What should I do to be a great geol
ogist?","1"
"11","23","24","How do I read and find my YouTube comments?","How can I see all my
Youtube comments?","1"
```

2.2 Mapping the real world problem to an ML problem

2.2.1 Type of Machine Leaning Problem

It is a binary classification problem, for a given pair of questions we need to predict if they are duplicate or not.

2.2.2 Performance Metric

Source: https://www.kaggle.com/c/quora-question-pairs#evaluation (https://www.kaggle.com/c/quora-question-pairs#evaluation)

Metric(s):

- log-loss : https://www.kaggle.com/wiki/LogarithmicLoss (https://www.kaggle.com/wiki/LogarithmicLoss)
- Binary Confusion Matrix

2.3 Train and Test Construction

We build train and test by randomly splitting in the ratio of 70:30 or 80:20 whatever we choose as we have sufficient points to work with.

3. Exploratory Data Analysis

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from subprocess import check_output
%matplotlib inline
import plotly.offline as py
py.init_notebook_mode(connected=True)
import plotly.graph_objs as go
import plotly.tools as tls
import os
import gc

import re
from nltk.corpus import stopwords
import distance
from nltk.stem import PorterStemmer
from bs4 import BeautifulSoup
```

3.1 Reading data and basic stats

In [2]:
```
df = pd.read_csv("train.csv")

print("Number of data points:",df.shape[0])
```

```
Number of data points: 404290
```

In [3]:
```
df.head()
```

Out[3]:

	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24}[/math] i...	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 404290 entries, 0 to 404289
Data columns (total 6 columns):
id              404290 non-null int64
qid1            404290 non-null int64
qid2            404290 non-null int64
question1       404289 non-null object
question2       404288 non-null object
is_duplicate    404290 non-null int64
dtypes: int64(4), object(2)
memory usage: 18.5+ MB
```

We are given a minimal number of data fields here, consisting of:

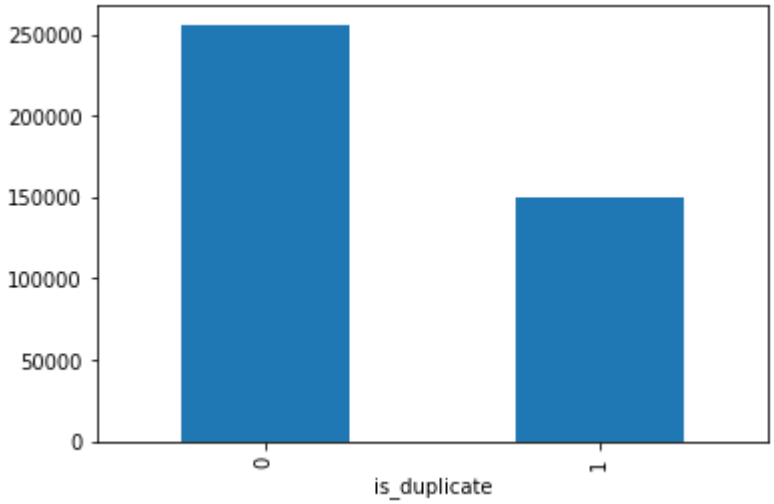
- id: Looks like a simple rowID
- qid{1, 2}: The unique ID of each question in the pair
- question{1, 2}: The actual textual contents of the questions.
- is_duplicate: The label that we are trying to predict - whether the two questions are duplicates of each other.

3.2.1 Distribution of data points among output classes

- Number of duplicate(smilar) and non-duplicate(non similar) questions

In [5]:
```
df.groupby("is_duplicate")['id'].count().plot.bar()
```

Out[5]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2804226d240>
```



In [6]:
```
print('~> Total number of question pairs for training:\n   {}'.format(len(df
)))
```

```
~> Total number of question pairs for training:
   404290
```

In [7]:
```
print('~> Question pairs are not Similar (is_duplicate = 0):\n   {}%'.format(1
00 - round(df['is_duplicate'].mean()*100, 2)))
print('\n~> Question pairs are Similar (is_duplicate = 1):\n   {}%'.format(rou
nd(df['is_duplicate'].mean()*100, 2)))
```

```
~> Question pairs are not Similar (is_duplicate = 0):
   63.08%

~> Question pairs are Similar (is_duplicate = 1):
   36.92%
```

In [8]:
```
df['is_duplicate'].mean()
```

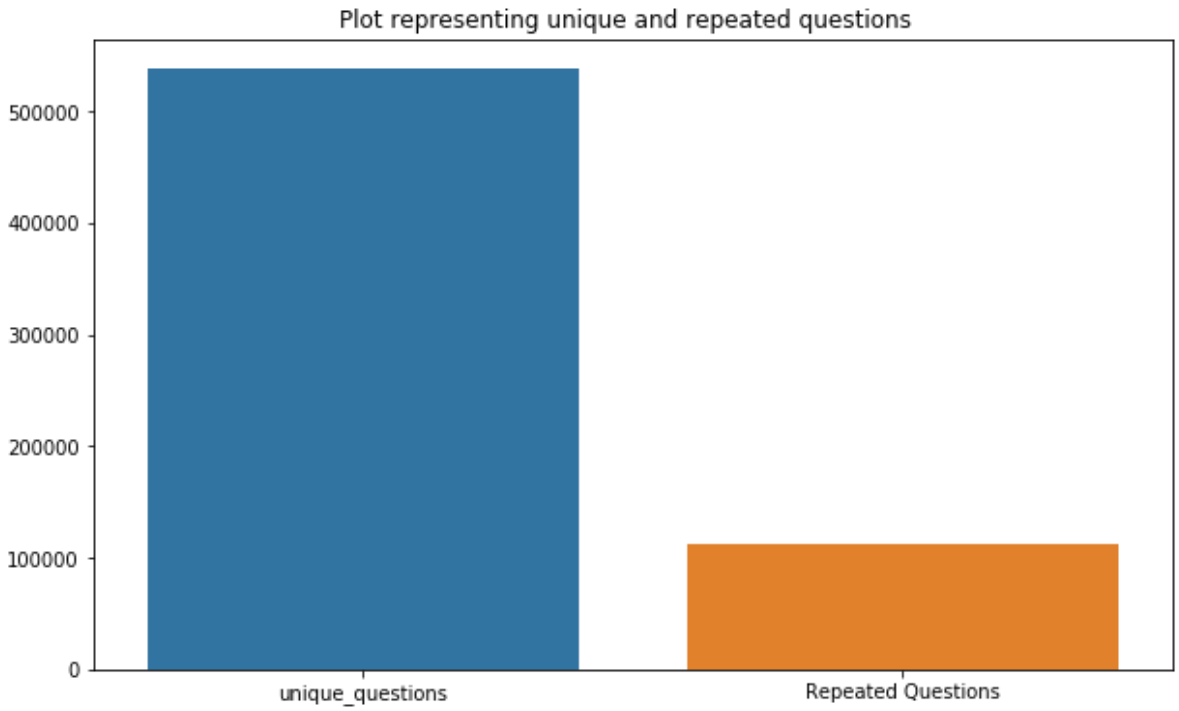
Out[8]: 0.369197853026293

3.2.2 Number of unique questions

In [9]:
```
qids = pd.Series(df['qid1'].tolist() + df['qid2'].tolist())
unique_qs = len(np.unique(qids))
qs_morethan_onetime = np.sum(qids.value_counts() > 1)
print ('Total number of  Unique Questions are: {}\n'.format(unique_qs))
#print len(np.unique(qids))

print ('Number of unique questions that appear more than one time: {} ({}%)\n'
.format(qs_morethan_onetime,qs_morethan_onetime/unique_qs*100))

print ('Max number of times a single question is repeated: {}\n'.format(max(qi
ds.value_counts())))

q_vals=qids.value_counts()

q_vals=q_vals.values
```

```
Total number of  Unique Questions are: 537933

Number of unique questions that appear more than one time: 111780 (20.7795394
5937505%)

Max number of times a single question is repeated: 157
```

In [10]:
```
x = ["unique_questions" , "Repeated Questions"]
y =  [unique_qs , qs_morethan_onetime]

plt.figure(figsize=(10, 6))
plt.title ("Plot representing unique and repeated questions  ")
sns.barplot(x,y)
plt.show()
```



3.2.3 Checking for Duplicates

In [11]:
```
#checking whether there are any repeated pair of questions

pair_duplicates = df[['qid1','qid2','is_duplicate']].groupby(['qid1','qid2']).
count().reset_index()

print ("Number of duplicate questions",(pair_duplicates).shape[0] - df.shape[0
])
```

```
Number of duplicate questions 0
```

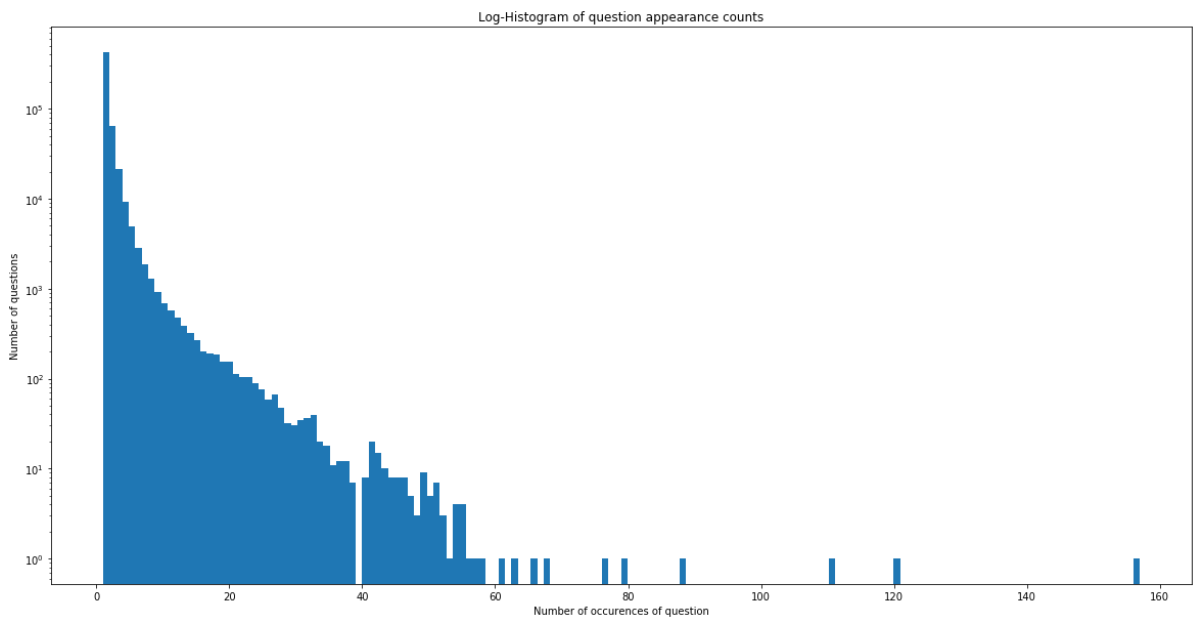
3.2.4 Number of occurrences of each question

In [12]:
```
qids.head()
```

Out[12]:
```
0    1
1    3
2    5
3    7
4    9
dtype: int64
```

In [13]:
```
plt.figure(figsize=(20, 10))

plt.hist(qids.value_counts(), bins=160)

plt.yscale('log', nonposy='clip')

plt.title('Log-Histogram of question appearance counts')

plt.xlabel('Number of occurences of question')

plt.ylabel('Number of questions')

print ('Maximum number of times a single question is repeated: {}\n'.format(ma
x(qids.value_counts())))
```

```
Maximum number of times a single question is repeated: 157
```



3.2.5 Checking for NULL values

In [14]:
```
df[df.isnull().any(1)]
```

Out[14]:

	id	qid1	qid2	question1	question2	is_duplicate
105780	105780	174363	174364	How can I develop android app?	NaN	0
201841	201841	303951	174364	How can I create an Android app?	NaN	0
363362	363362	493340	493341	NaN	My Chinese name is Haichao Yu. What English na...	0

In [15]:
```
#Checking whether there are any rows with null values
nan_rows = df[df.isnull().any(1)]
print (nan_rows)
```

```
            id    qid1    qid2                         question1  \
105780  105780  174363  174364    How can I develop android app?
201841  201841  303951  174364  How can I create an Android app?
363362  363362  493340  493341                               NaN

                                                question2  is_duplicate
105780                                                NaN             0
201841                                                NaN             0
363362  My Chinese name is Haichao Yu. What English na...             0
```

In [16]:
```
# Filling the null values with ' '
df = df.fillna('')
nan_rows = df[df.isnull().any(1)]
print (nan_rows)
```

```
Empty DataFrame
Columns: [id, qid1, qid2, question1, question2, is_duplicate]
Index: []
```

3.3 Basic Feature Extraction (before cleaning)

Let us now construct a few features like:

- **freq_qid1** = Frequency of qid1's
- **freq_qid2** = Frequency of qid2's
- **q1len** = Length of q1
- **q2len** = Length of q2
- **q1_n_words** = Number of words in Question 1
- **q2_n_words** = Number of words in Question 2
- **word_Common** = (Number of common unique words in Question 1 and Question 2)
- **word_Total** =(Total num of words in Question 1 + Total num of words in Question 2)
- **word_share** = (word_common)/(word_Total)
- **freq_q1+freq_q2** = sum total of frequency of qid1 and qid2
- **freq_q1-freq_q2** = absolute difference of frequency of qid1 and qid2

In [17]:
```
if os.path.isfile('df_fe_without_preprocessing_train.csv'):
    df = pd.read_csv("df_fe_without_preprocessing_train.csv",encoding='latin-
1')
else:
    df['freq_qid1'] = df.groupby('qid1')['qid1'].transform('count')
    df['freq_qid2'] = df.groupby('qid2')['qid2'].transform('count')
    df['q1len'] = df['question1'].str.len()
    df['q2len'] = df['question2'].str.len()
    df['q1_n_words'] = df['question1'].apply(lambda row: len(row.split(" ")))
    df['q2_n_words'] = df['question2'].apply(lambda row: len(row.split(" ")))

    def normalized_word_Common(row):
        w1 = set(map(lambda word: word.lower().strip(), row['question1'].split
(" ")))
        w2 = set(map(lambda word: word.lower().strip(), row['question2'].split
(" ")))
        return 1.0 * len(w1 & w2)
    df['word_Common'] = df.apply(normalized_word_Common, axis=1)

    def normalized_word_Total(row):
        w1 = set(map(lambda word: word.lower().strip(), row['question1'].split
(" ")))
        w2 = set(map(lambda word: word.lower().strip(), row['question2'].split
(" ")))
        return 1.0 * (len(w1) + len(w2))
    df['word_Total'] = df.apply(normalized_word_Total, axis=1)

    def normalized_word_share(row):
        w1 = set(map(lambda word: word.lower().strip(), row['question1'].split
(" ")))
        w2 = set(map(lambda word: word.lower().strip(), row['question2'].split
(" ")))
        return 1.0 * len(w1 & w2)/(len(w1) + len(w2))
    df['word_share'] = df.apply(normalized_word_share, axis=1)

    df['freq_q1+q2'] = df['freq_qid1']+df['freq_qid2']
    df['freq_q1-q2'] = abs(df['freq_qid1']-df['freq_qid2'])

    df.to_csv("df_fe_without_preprocessing_train.csv", index=False)

df.head()
```

Out[17]:

	id	qid1	qid2	question1	question2	is_duplicate	freq_qid1	freq_qid2	q1len	q2len	q1_
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0	1	1	66	57	
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0	4	1	51	88	
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0	1	1	73	59	
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24} [/math] i...	0	1	1	50	65	
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0	3	1	76	39	

3.3.1 Analysis of some of the extracted features

- Here are some questions have only one single words.

In [18]:
```
print ("Minimum length of the questions in question1 : " , min(df['q1_n_words'
]))

print ("Minimum length of the questions in question2 : " , min(df['q2_n_words'
]))

print ("Number of Questions with minimum length [question1] :", df[df['q1_n_wo
rds']== 1].shape[0])
print ("Number of Questions with minimum length [question2] :", df[df['q2_n_wo
rds']== 1].shape[0])
```

```
Minimum length of the questions in question1 :  1
Minimum length of the questions in question2 :  1
Number of Questions with minimum length [question1] : 67
Number of Questions with minimum length [question2] : 24
```

3.3.1.1 Feature: word_share

In [19]:
```
df[0:]
```

Out[19]:

	id	qid1	qid2	question1	question2	is_duplicate	freq_qid1
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0	1
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0	4
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0	1
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24} [/math] i...	0	1
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0	3
5	5	11	12	Astrology: I am a Capricorn Sun Cap moon and c...	I'm a triple Capricorn (Sun, Moon and ascendan...	1	1
6	6	13	14	Should I buy tiago?	What keeps childern active and far from phone ...	0	1
7	7	15	16	How can I be a good geologist?	What should I do to be a great geologist?	1	1
8	8	17	18	When do you use ã□· instead of ã□□?	When do you use "&" instead of "and"?	0	1
9	9	19	20	Motorola (company): Can I hack my Charter Moto...	How do I hack Motorola DCX3400 for free internet?	0	1
10	10	21	22	Method to find separation of slits using fresn...	What are some of the things technicians can te...	0	1
11	11	23	24	How do I read and find my YouTube comments?	How can I see all my Youtube comments?	1	1
12	12	25	26	What can make Physics easy to learn?	How can you make physics easy to learn?	1	1
13	13	27	28	What was your first sexual experience like?	What was your first sexual experience?	1	2
14	14	29	30	What are the laws to change your status from a...	What are the laws to change your status from a...	0	5

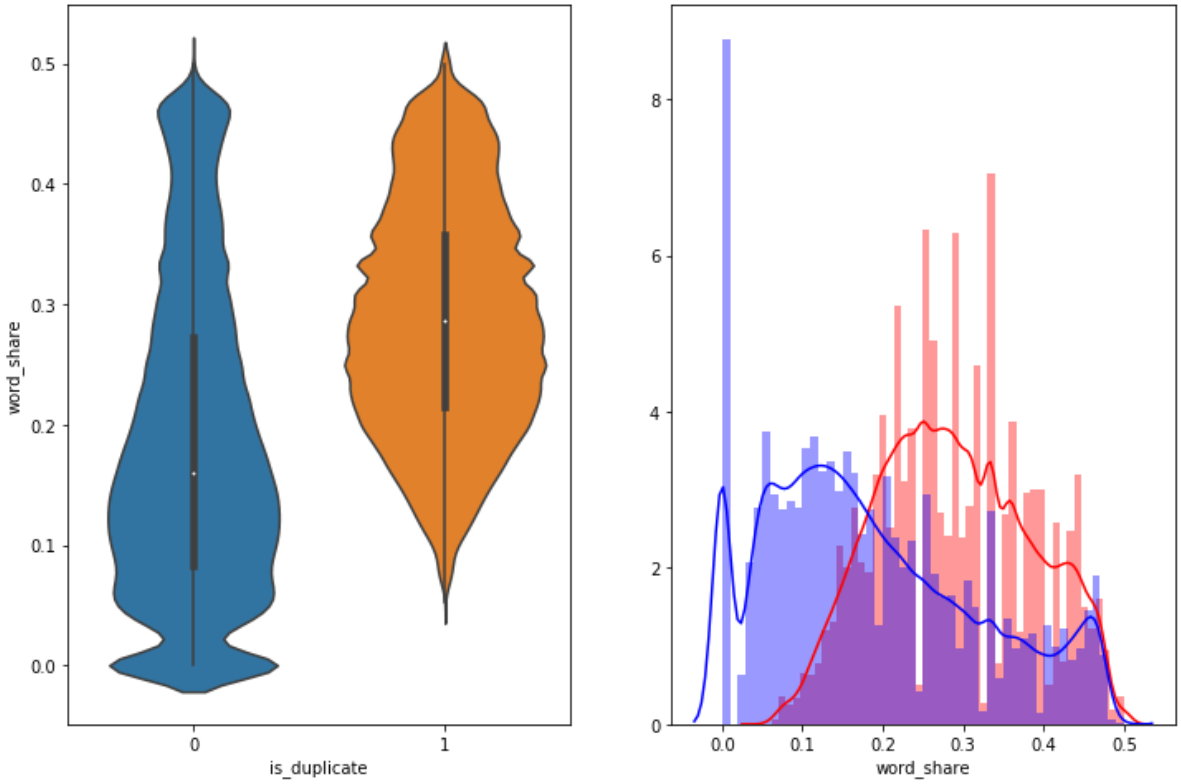
	id	qid1	qid2	question1	question2	is_duplicate	freq_qid1
15	15	31	32	What would a Trump presidency mean for current...	How will a Trump presidency affect the student...	1	7
16	16	33	34	What does manipulation mean?	What does manipulation means?	1	1
17	17	35	36	Why do girls want to be friends with the guy t...	How do guys feel after rejecting a girl?	0	1
18	18	37	38	Why are so many Quora users posting questions ...	Why do people ask Quora questions which can be...	1	18
19	19	39	40	Which is the best digital marketing institutio...	Which is the best digital marketing institute ...	0	1
20	20	41	42	Why do rockets look white?	Why are rockets and boosters painted white?	1	1
21	21	43	44	What's causing someone to be jealous?	What can I do to avoid being jealous of someone?	0	1
22	22	45	46	What are the questions should not ask on Quora?	Which question should I ask on Quora?	0	4
23	23	47	48	How much is 30 kV in HP?	Where can I find a conversion chart for CC to ...	0	1
24	24	49	50	What does it mean that every time I look at th...	How many times a day do a clockâ□□s hands over...	0	3
25	25	51	52	What are some tips on making it through the jo...	What are some tips on making it through the jo...	0	4
26	26	53	54	What is web application?	What is the web application framework?	0	3
27	27	55	56	Does society place too much importance on sports?	How do sports contribute to the society?	0	1
28	28	57	58	What is best way to make money online?	What is best way to ask for money online?	0	20
29	29	59	60	How should I prepare for CA final law?	How one should know that he/she completely pre...	1	10
...	...	...	...	...	...	...	...
404260	404260	182494	691	Which phone is best under 12000?	What is the best phone to buy below 15k?	0	2

	id	qid1	qid2	question1	question2	is_duplicate	freq_qid1
404261	404261	281150	124172	Who is the overall most popular Game of Throne...	Who is the most popular character in the Game ...	1	1
404262	404262	537905	466328	How do you troubleshoot a Toshiba laptop?	How do I reset a Toshiba laptop?	0	1
404263	404263	375195	537906	How does the burning of fossil fuels contribut...	Why does CO2 contribute more to global warming...	0	1
404264	404264	537907	537908	Is it safe to store an external battery power ...	How do I make a safe and cheap power bank?	0	1
404265	404265	25994	16064	How can I gain weight on my body?	What should I eat to gain weight?	1	19
404266	404266	155813	146284	What is the green dot next to the phone icon o...	My boyfriend says he deleted his Facebook Mess...	0	4
404267	404267	20171	290649	What are the causes of the fall of the Roman E...	What were the most important causes and effect...	1	4
404268	404268	537909	537910	Why don't we still do great music like in the ...	Should I raise my young child on 80's music?	0	1
404269	404269	537911	349794	How do you diagnose antisocial personality dis...	What Does It Feel Like to have antisocial pers...	0	1
404270	404270	537912	35364	What is the difference between who and how?	What is the difference between "&" and "and"?	0	1
404271	404271	537913	537914	Does Stalin have any grandchildren that are st...	What was Joseph Stalin's 5 year plan? How did ...	0	1
404272	404272	128018	14005	What are the best new car products or inventio...	What are some mind-blowing vehicles tools that...	1	6
404273	404273	537915	537916	What happens if you put milk in a coffee maker?	What would happen if I put milk instead of wat...	1	1
404274	404274	178643	87385	Will the next generation of parenting change o...	What kind of parents will the next generation ...	1	4
404275	404275	97922	537917	In accounting, why do we debit expenses and cr...	What is a utilities expense in accounting? How...	0	3
404276	404276	24305	308365	What is copilotsearch.com?	What is ContenVania.com?	0	7
404277	404277	355668	537918	What does analytics do?	What are analytical people like?	0	1

	id	qid1	qid2	question1	question2	is_duplicate	freq_qid1
404278	404278	537919	169786	How did you prepare for AIIMS/NEET/AIPMT?	How did you prepare for the AIIMS UG entrance ...	0	1
404279	404279	537920	537921	What is the minimum time required to build a f...	What is a cheaper and quicker way to build an ...	0	1
404280	404280	537922	537923	What are some outfit ideas to wear to a frat p...	What are some outfit ideas wear to a frat them...	1	1
404281	404281	99131	81495	Why is Manaphy childish in PokÃ©mon Ranger and...	Why is Manaphy annoying in Pokemon ranger and ...	1	7
404282	404282	1931	16773	How does a long distance relationship work?	How are long distance relationships maintained?	1	26
404283	404283	537924	537925	What do you think of the removal of the MagSaf...	What will the CPU upgrade to the 2016 Apple Ma...	0	1
404284	404284	537926	537927	What does Jainism say about homosexuality?	What does Jainism say about Gays and Homosexua...	1	1
404285	404285	433578	379845	How many keywords are there in the Racket prog...	How many keywords are there in PERL Programmin...	0	2
404286	404286	18840	155606	Do you believe there is life after death?	Is it true that there is life after death?	1	12
404287	404287	537928	537929	What is one coin?	What's this coin?	0	1
404288	404288	537930	537931	What is the approx annual cost of living while...	I am having little hairfall problem but I want...	0	1
404289	404289	537932	537933	What is like to have sex with cousin?	What is it like to have sex with your cousin?	0	1

404290 rows × 17 columns

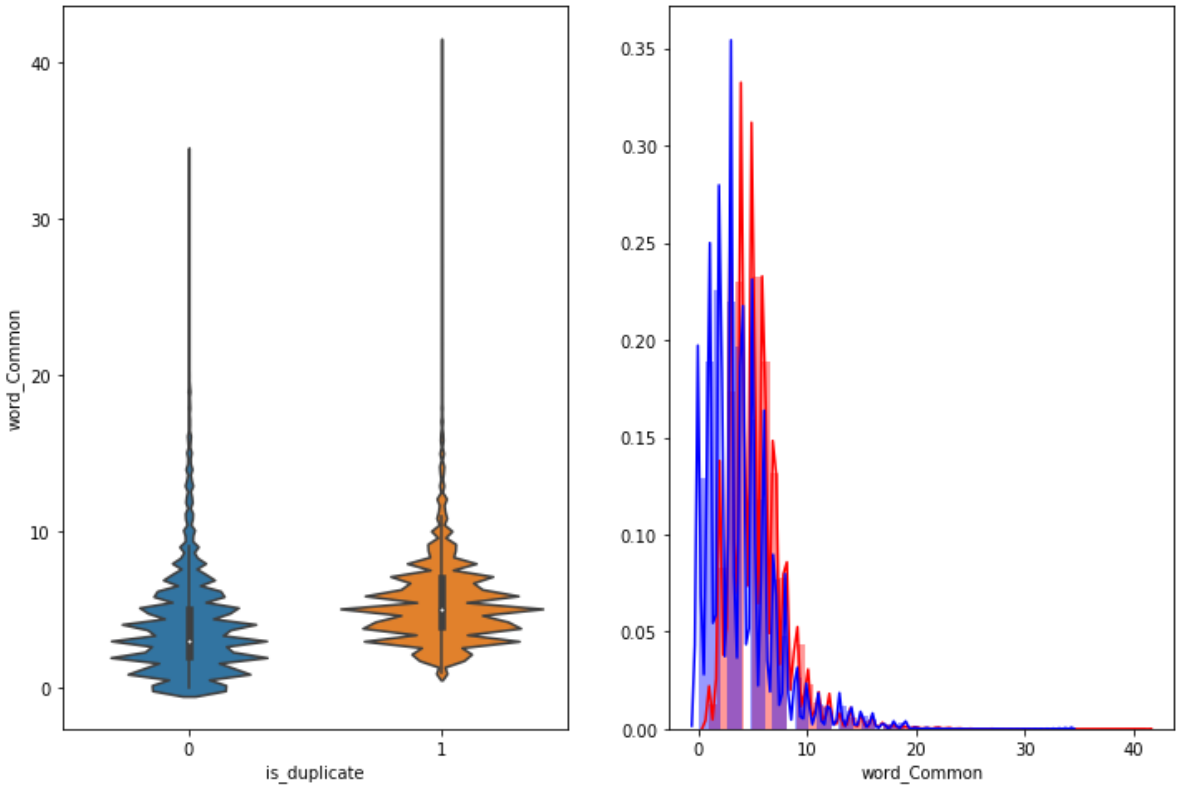
In [20]:
```
plt.figure(figsize=(12, 8))

plt.subplot(1,2,1)
sns.violinplot(x = 'is_duplicate', y = 'word_share', data = df[0:])

plt.subplot(1,2,2)
sns.distplot(df[df['is_duplicate'] == 1.0]['word_share'][0:] , label = "1", co
lor = 'red')
sns.distplot(df[df['is_duplicate'] == 0.0]['word_share'][0:] , label = "0" , c
olor = 'blue' )
plt.show()
```



- The distributions for normalized word_share have some overlap on the far right-hand side, i.e., there are quite a lot of questions with high word similarity
- The average word share and Common no. of words of qid1 and qid2 is more when they are duplicate(Similar)

3.3.1.2 Feature: word_Common

In [21]:
```
plt.figure(figsize=(12, 8))

plt.subplot(1,2,1)
sns.violinplot(x = 'is_duplicate', y = 'word_Common', data = df[0:])

plt.subplot(1,2,2)
sns.distplot(df[df['is_duplicate'] == 1.0]['word_Common'][0:] , label = "1", c
olor = 'red')
sns.distplot(df[df['is_duplicate'] == 0.0]['word_Common'][0:] , label = "0" ,
color = 'blue' )
plt.show()
```



The distributions of the word_Common feature in similar and non-similar questions are highly overlapping

In [ ]:

2nd Notebook:Quora_Preprocessing.ipynb

1.2.1 : EDA: Advanced Feature Extraction.

In [22]:
```
import warnings
warnings.filterwarnings("ignore")
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from subprocess import check_output
%matplotlib inline
import plotly.offline as py
py.init_notebook_mode(connected=True)
import plotly.graph_objs as go
import plotly.tools as tls
import os
import gc

import re
from nltk.corpus import stopwords
import distance
from nltk.stem import PorterStemmer
from bs4 import BeautifulSoup
import re
from nltk.corpus import stopwords
# This package is used for finding longest common subsequence between two stri
ngs
# you can write your own dp code for this
import distance
from nltk.stem import PorterStemmer
from bs4 import BeautifulSoup
from fuzzywuzzy import fuzz
from sklearn.manifold import TSNE
# Import the Required lib packages for WORD-Cloud generation
# https://stackoverflow.com/questions/45625434/how-to-install-wordcloud-in-pyt
hon3-6
from wordcloud import WordCloud, STOPWORDS
from os import path
from PIL import Image
import nltk
# nltk.download()
```

In [23]:
```
#https://stackoverflow.com/questions/12468179/unicodedecodeerror-utf8-codec-ca
nt-decode-byte-0x9c
if os.path.isfile('df_fe_without_preprocessing_train.csv'):
    df = pd.read_csv("df_fe_without_preprocessing_train.csv",encoding='latin-
1')
    df = df.fillna('')
    df.head()
else:
    print("get df_fe_without_preprocessing_train.csv from drive or run the pre
vious notebook")
```

In [24]:
```
df.head(2)
```

Out[24]:

	id	qid1	qid2	question1	question2	is_duplicate	freq_qid1	freq_qid2	q1len	q2len	q1_n_
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0	1	1	66	57	
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0	4	1	51	88	

3.4 Preprocessing of Text

- Preprocessing:
  - Removing html tags
  - Removing Punctuations
  - Performing stemming
  - Removing Stopwords
  - Expanding contractions etc.

In [25]:
```
# To get the results in 4 decemal points
SAFE_DIV = 0.0001

STOP_WORDS = stopwords.words("english")


def preprocess(x):
    x = str(x).lower()
    x = x.replace(",000,000", "m").replace(",000", "k").replace("’", "'").repl
ace("’", "'")\
                           .replace("won't", "will not").replace("cannot", "ca
n not").replace("can't", "can not")\
                           .replace("n't", " not").replace("what's", "what is"
).replace("it's", "it is")\
                           .replace("'ve", " have").replace("i'm", "i am").rep
lace("'re", " are")\
                           .replace("he's", "he is").replace("she's", "she is"
).replace("'s", " own")\
                           .replace("%", " percent ").replace("₹", " rupee "). 
replace("$", " dollar ")\
                           .replace("€", " euro ").replace("'ll", " will")
    x = re.sub(r"([0-9]+)000000", r"\1m", x)
    x = re.sub(r"([0-9]+)000", r"\1k", x)


    porter = PorterStemmer()
    pattern = re.compile('\W')

    if type(x) == type(''):
        x = re.sub(pattern, ' ', x)


    if type(x) == type(''):
        x = porter.stem(x)
        example1 = BeautifulSoup(x)
        x = example1.get_text()


    return x
```

- Function to Compute and get the features : With 2 parameters of Question 1 and Question 2

3.5 Advanced Feature Extraction (NLP and Fuzzy Features)

Definition:

- **Token**: You get a token by splitting sentence a space
- **Stop_Word** : stop words as per NLTK.
- **Word** : A token that is not a stop_word

Features:

- **cwc_min** : Ratio of common_word_count to min lenghth of word count of Q1 and Q2
  cwc_min = common_word_count / (min(len(q1_words), len(q2_words))

- **cwc_max** : Ratio of common_word_count to max lenghth of word count of Q1 and Q2
  cwc_max = common_word_count / (max(len(q1_words), len(q2_words))

- **csc_min** : Ratio of common_stop_count to min lenghth of stop count of Q1 and Q2
  csc_min = common_stop_count / (min(len(q1_stops), len(q2_stops))

- **csc_max** : Ratio of common_stop_count to max lenghth of stop count of Q1 and Q2
  csc_max = common_stop_count / (max(len(q1_stops), len(q2_stops))

- **ctc_min** : Ratio of common_token_count to min lenghth of token count of Q1 and Q2
  ctc_min = common_token_count / (min(len(q1_tokens), len(q2_tokens))

- **ctc_max** : Ratio of common_token_count to max lenghth of token count of Q1 and Q2
  ctc_max = common_token_count / (max(len(q1_tokens), len(q2_tokens))

- **last_word_eq** : Check if First word of both questions is equal or not
  last_word_eq = int(q1_tokens[-1] == q2_tokens[-1])

- **first_word_eq** : Check if First word of both questions is equal or not
  first_word_eq = int(q1_tokens[0] == q2_tokens[0])

- **abs_len_diff** : Abs. length difference
  abs_len_diff = abs(len(q1_tokens) - len(q2_tokens))

- **mean_len** : Average Token Length of both Questions
  mean_len = (len(q1_tokens) + len(q2_tokens))/2

- **fuzz_ratio** : https://github.com/seatgeek/fuzzywuzzy#usage (https://github.com/seatgeek/fuzzywuzzy#usage) http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/ (http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/)

- **fuzz_partial_ratio** : https://github.com/seatgeek/fuzzywuzzy#usage (https://github.com/seatgeek/fuzzywuzzy#usage) http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/ (http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/)

- **token_sort_ratio** : https://github.com/seatgeek/fuzzywuzzy#usage (https://github.com/seatgeek/fuzzywuzzy#usage) http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/ (http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/)

- **token_set_ratio** : https://github.com/seatgeek/fuzzywuzzy#usage (https://github.com/seatgeek/fuzzywuzzy#usage) http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/ (http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/)

- **longest_substr_ratio** : Ratio of length longest common substring to min lenghth of token count of Q1 and Q2
  longest_substr_ratio = len(longest common substring) / (min(len(q1_tokens), len(q2_tokens))

In [26]:
```
def get_token_features(q1, q2):
    token_features = [0.0]*10

    # Converting the Sentence into Tokens:
    q1_tokens = q1.split()
    q2_tokens = q2.split()

    if len(q1_tokens) == 0 or len(q2_tokens) == 0:
        return token_features
    # Get the non-stopwords in Questions
    q1_words = set([word for word in q1_tokens if word not in STOP_WORDS])
    q2_words = set([word for word in q2_tokens if word not in STOP_WORDS])

    #Get the stopwords in Questions
    q1_stops = set([word for word in q1_tokens if word in STOP_WORDS])
    q2_stops = set([word for word in q2_tokens if word in STOP_WORDS])

    # Get the common non-stopwords from Question pair
    common_word_count = len(q1_words.intersection(q2_words))

    # Get the common stopwords from Question pair
    common_stop_count = len(q1_stops.intersection(q2_stops))

    # Get the common Tokens from Question pair
    common_token_count = len(set(q1_tokens).intersection(set(q2_tokens)))


    token_features[0] = common_word_count / (min(len(q1_words), len(q2_words))
+ SAFE_DIV)
    token_features[1] = common_word_count / (max(len(q1_words), len(q2_words))
+ SAFE_DIV)
    token_features[2] = common_stop_count / (min(len(q1_stops), len(q2_stops))
+ SAFE_DIV)
    token_features[3] = common_stop_count / (max(len(q1_stops), len(q2_stops))
+ SAFE_DIV)
    token_features[4] = common_token_count / (min(len(q1_tokens), len(q2_token
s)) + SAFE_DIV)
    token_features[5] = common_token_count / (max(len(q1_tokens), len(q2_token
s)) + SAFE_DIV)

    # Last word of both question is same or not
    token_features[6] = int(q1_tokens[-1] == q2_tokens[-1])

    # First word of both question is same or not
    token_features[7] = int(q1_tokens[0] == q2_tokens[0])

    token_features[8] = abs(len(q1_tokens) - len(q2_tokens))

    #Average Token Length of both Questions
    token_features[9] = (len(q1_tokens) + len(q2_tokens))/2
    return token_features

# get the Longest Common sub string

def get_longest_substr_ratio(a, b):
    strs = list(distance.lcsubstrings(a, b))
```

```
    if len(strs) == 0:
        return 0
    else:
        return len(strs[0]) / (min(len(a), len(b)) + 1)

def extract_features(df):
    # preprocessing each question
    df["question1"] = df["question1"].fillna("").apply(preprocess)
    df["question2"] = df["question2"].fillna("").apply(preprocess)

    print("token features...")

    # Merging Features with dataset

    token_features = df.apply(lambda x: get_token_features(x["question1"], x[
"question2"]), axis=1)

    df["cwc_min"]       = list(map(lambda x: x[0], token_features))
    df["cwc_max"]       = list(map(lambda x: x[1], token_features))
    df["csc_min"]       = list(map(lambda x: x[2], token_features))
    df["csc_max"]       = list(map(lambda x: x[3], token_features))
    df["ctc_min"]       = list(map(lambda x: x[4], token_features))
    df["ctc_max"]       = list(map(lambda x: x[5], token_features))
    df["last_word_eq"]  = list(map(lambda x: x[6], token_features))
    df["first_word_eq"] = list(map(lambda x: x[7], token_features))
    df["abs_len_diff"]  = list(map(lambda x: x[8], token_features))
    df["mean_len"]      = list(map(lambda x: x[9], token_features))

    #Computing Fuzzy Features and Merging with Dataset

    # do read this blog: http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string
-matching-in-python/
    # https://stackoverflow.com/questions/31806695/when-to-use-which-fuzz-func
tion-to-compare-2-strings
    # https://github.com/seatgeek/fuzzywuzzy
    print("fuzzy features..")

    df["token_set_ratio"]       = df.apply(lambda x: fuzz.token_set_ratio(x["q
uestion1"], x["question2"]), axis=1)
    # The token sort approach involves tokenizing the string in question, sort
ing the tokens alphabetically, and
    # then joining them back into a string We then compare the transformed str
ings with a simple ratio().
    df["token_sort_ratio"]      = df.apply(lambda x: fuzz.token_sort_ratio(x[
"question1"], x["question2"]), axis=1)
    df["fuzz_ratio"]            = df.apply(lambda x: fuzz.QRatio(x["question1"
], x["question2"]), axis=1)
    df["fuzz_partial_ratio"]    = df.apply(lambda x: fuzz.partial_ratio(x["que
stion1"], x["question2"]), axis=1)
    df["longest_substr_ratio"]  = df.apply(lambda x: get_longest_substr_ratio(
x["question1"], x["question2"]), axis=1)
    return df
```

In [27]:
```
if os.path.isfile('nlp_features_train.csv'):
    df = pd.read_csv("nlp_features_train.csv",encoding='latin-1')
    df.fillna('')
else:
    print("Extracting features for train:")
    df = pd.read_csv("train.csv")
    df = extract_features(df)
    df.to_csv("nlp_features_train.csv", index=False)
df.head(2)
```

Out[27]:

	id	qid1	qid2	question1	question2	is_duplicate	cwc_min	cwc_max	csc_min	csc_max
0	0	1	2	what is the step by step guide to invest in sh...	what is the step by step guide to invest in sh...	0	0.999980	0.833319	0.999983	0.999983
1	1	3	4	what is the story of kohinoor koh i noor dia...	what would happen if the indian government sto...	0	0.799984	0.399996	0.749981	0.599988

2 rows × 21 columns

3.5.1 Analysis of extracted features

3.5.1.1 Plotting Word clouds

- Creating Word Cloud of Duplicates and Non-Duplicates Question pairs
- We can observe the most frequent occuring words

In [28]:
```
df_duplicate = df[df['is_duplicate'] == 1]
dfp_nonduplicate = df[df['is_duplicate'] == 0]

# Converting 2d array of q1 and q2 and flatten the array: like {{1,2},{3,4}} t
o {1,2,3,4}
p = np.dstack([df_duplicate["question1"], df_duplicate["question2"]]).flatten
()
n = np.dstack([dfp_nonduplicate["question1"], dfp_nonduplicate["question2"]]).
flatten()

print ("Number of data points in class 1 (duplicate pairs) :",len(p))
print ("Number of data points in class 0 (non duplicate pairs) :",len(n))

#Saving the np array into a text file
np.savetxt('train_p.txt', p, delimiter=' ', fmt='%s',encoding="utf-8")
np.savetxt('train_n.txt', n, delimiter=' ', fmt='%s',encoding="utf-8")
```

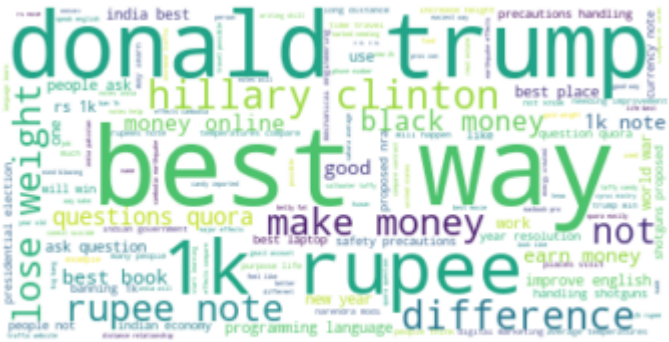
```
Number of data points in class 1 (duplicate pairs) : 298526
Number of data points in class 0 (non duplicate pairs) : 510054
```

In [29]:
```
# reading the text files and removing the Stop Words:
d = path.dirname('.')

textp_w = open(path.join(d, 'train_p.txt')).read()
textn_w = open(path.join(d, 'train_n.txt')).read()
stopwords = set(STOPWORDS)
stopwords.add("said")
stopwords.add("br")
stopwords.add(" ")
stopwords.remove("not")

stopwords.remove("no")
#stopwords.remove("good")
#stopwords.remove("love")
stopwords.remove("like")
#stopwords.remove("best")
#stopwords.remove("!")
print ("Total number of words in duplicate pair questions :",len(textp_w))
print ("Total number of words in non duplicate pair questions :",len(textn_w))
```

```
Total number of words in duplicate pair questions : 16110303
Total number of words in non duplicate pair questions : 33194892
```

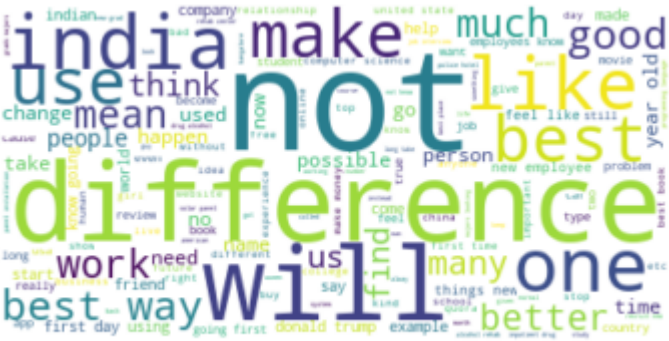
In [30]:
```
wc = WordCloud(background_color="white", max_words=len(textp_w), stopwords=sto
pwords)
wc.generate(textp_w)
print ("Word Cloud for Duplicate Question pairs")
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
plt.show()
```

```
Word Cloud for Duplicate Question pairs
```



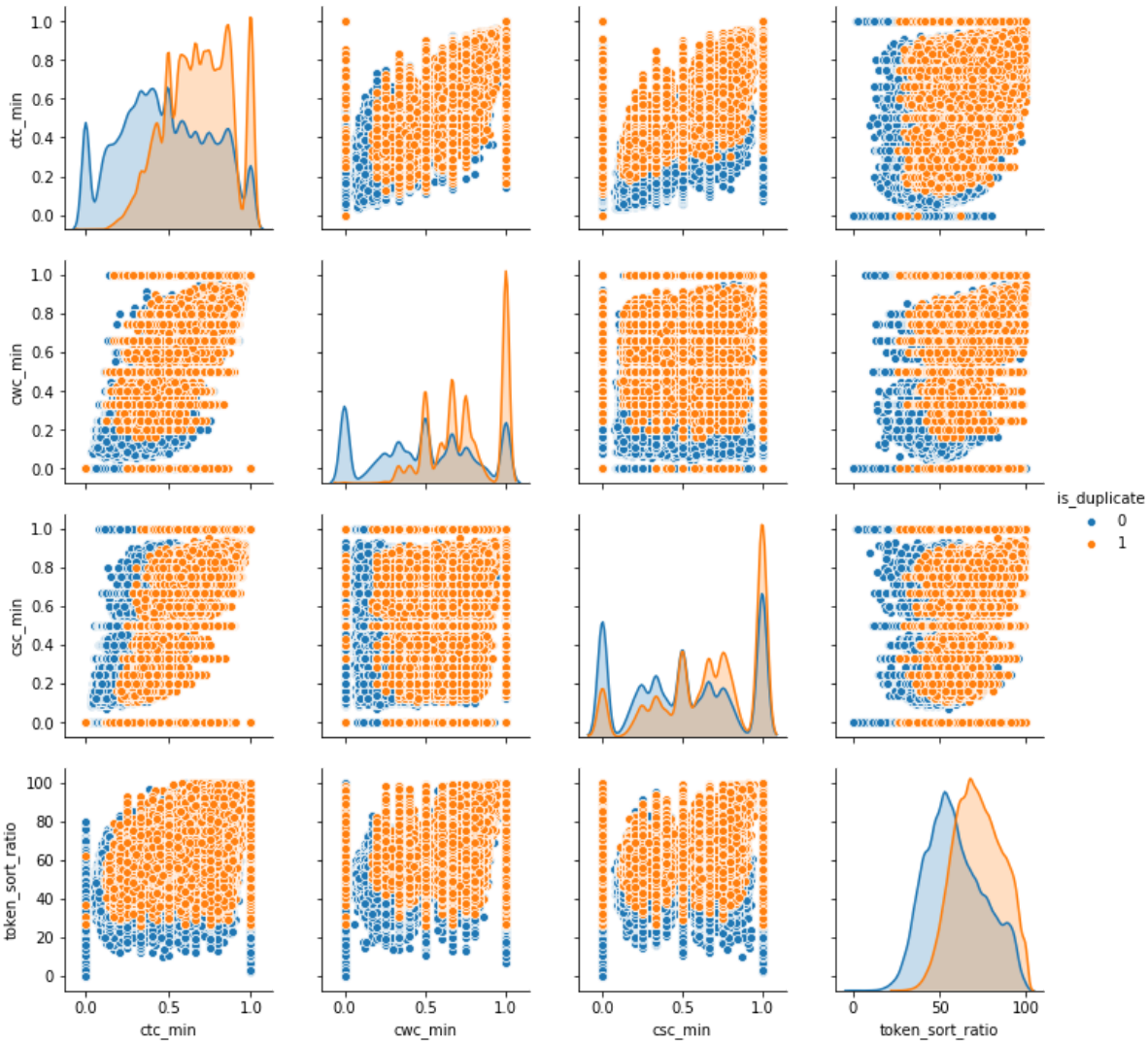
In [31]:
```
wc = WordCloud(background_color="white", max_words=len(textn_w),stopwords=stop
words)
# generate word cloud
wc.generate(textn_w)
print ("Word Cloud for non-Duplicate Question pairs:")
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
plt.show()
```

```
Word Cloud for non-Duplicate Question pairs:
```

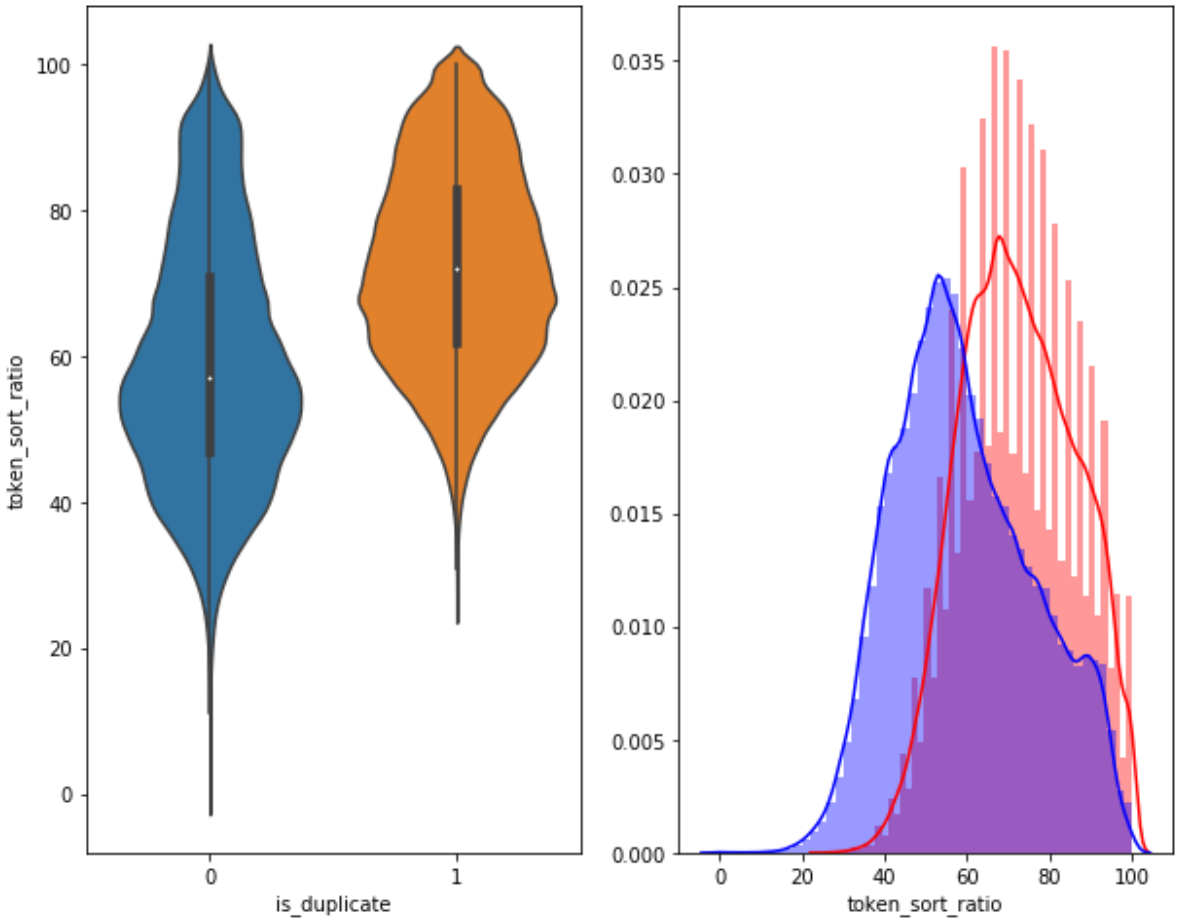


3.5.1.2 Pair plot of features ['ctc_min', 'cwc_min', 'csc_min', 'token_sort_ratio']

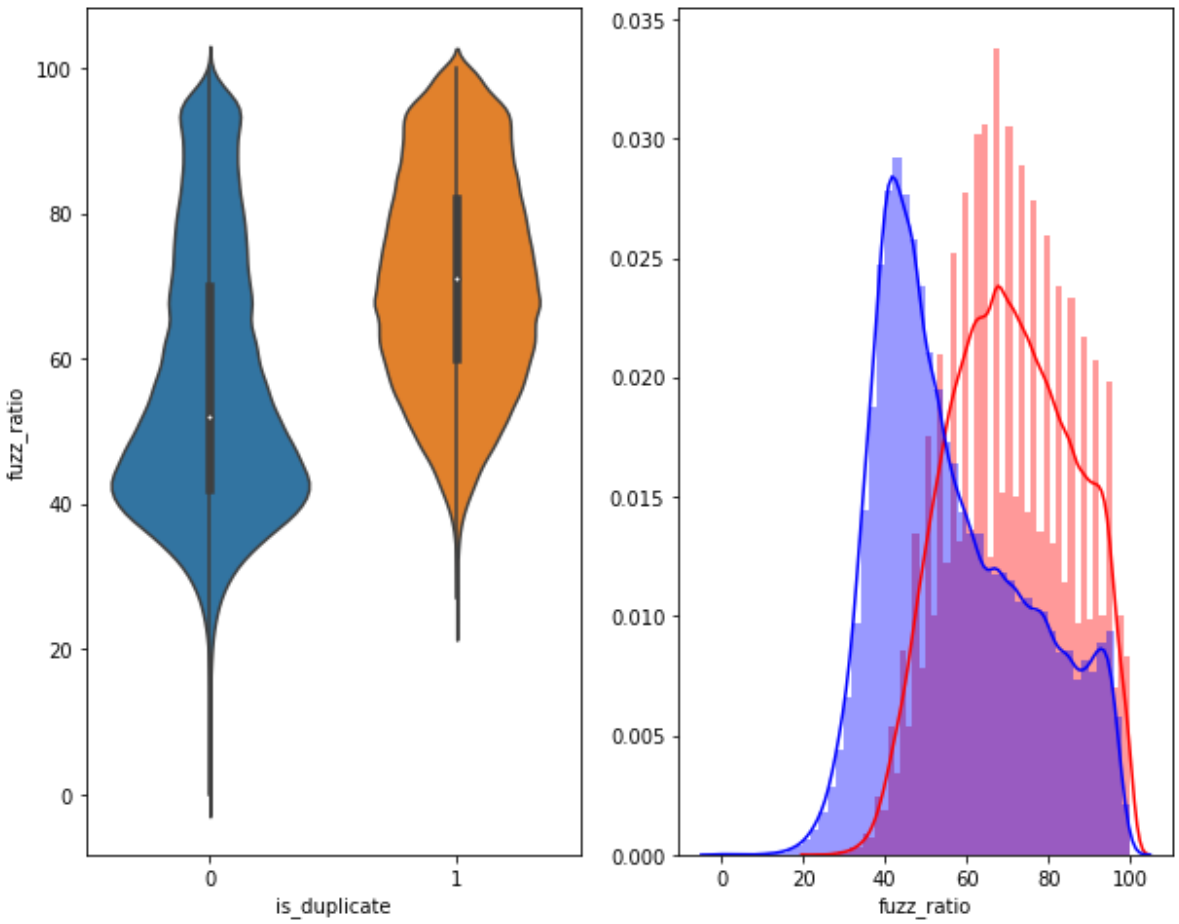
In [32]:
```
n = df.shape[0]
sns.pairplot(df[['ctc_min', 'cwc_min', 'csc_min', 'token_sort_ratio', 'is_dupl
icate']][0:n], hue='is_duplicate', vars=['ctc_min', 'cwc_min', 'csc_min', 'tok
en_sort_ratio'])
plt.show()
```



In [33]:
```
# Distribution of the token_sort_ratio
plt.figure(figsize=(10, 8))

plt.subplot(1,2,1)
sns.violinplot(x = 'is_duplicate', y = 'token_sort_ratio', data = df[0:] , )

plt.subplot(1,2,2)
sns.distplot(df[df['is_duplicate'] == 1.0]['token_sort_ratio'][0:] , label =
"1", color = 'red')
sns.distplot(df[df['is_duplicate'] == 0.0]['token_sort_ratio'][0:] , label =
"0" , color = 'blue' )
plt.show()
```



In [34]:
```
plt.figure(figsize=(10, 8))

plt.subplot(1,2,1)
sns.violinplot(x = 'is_duplicate', y = 'fuzz_ratio', data = df[0:] , )

plt.subplot(1,2,2)
sns.distplot(df[df['is_duplicate'] == 1.0]['fuzz_ratio'][0:] , label = "1", co
lor = 'red')
sns.distplot(df[df['is_duplicate'] == 0.0]['fuzz_ratio'][0:] , label = "0" , c
olor = 'blue' )
plt.show()
```



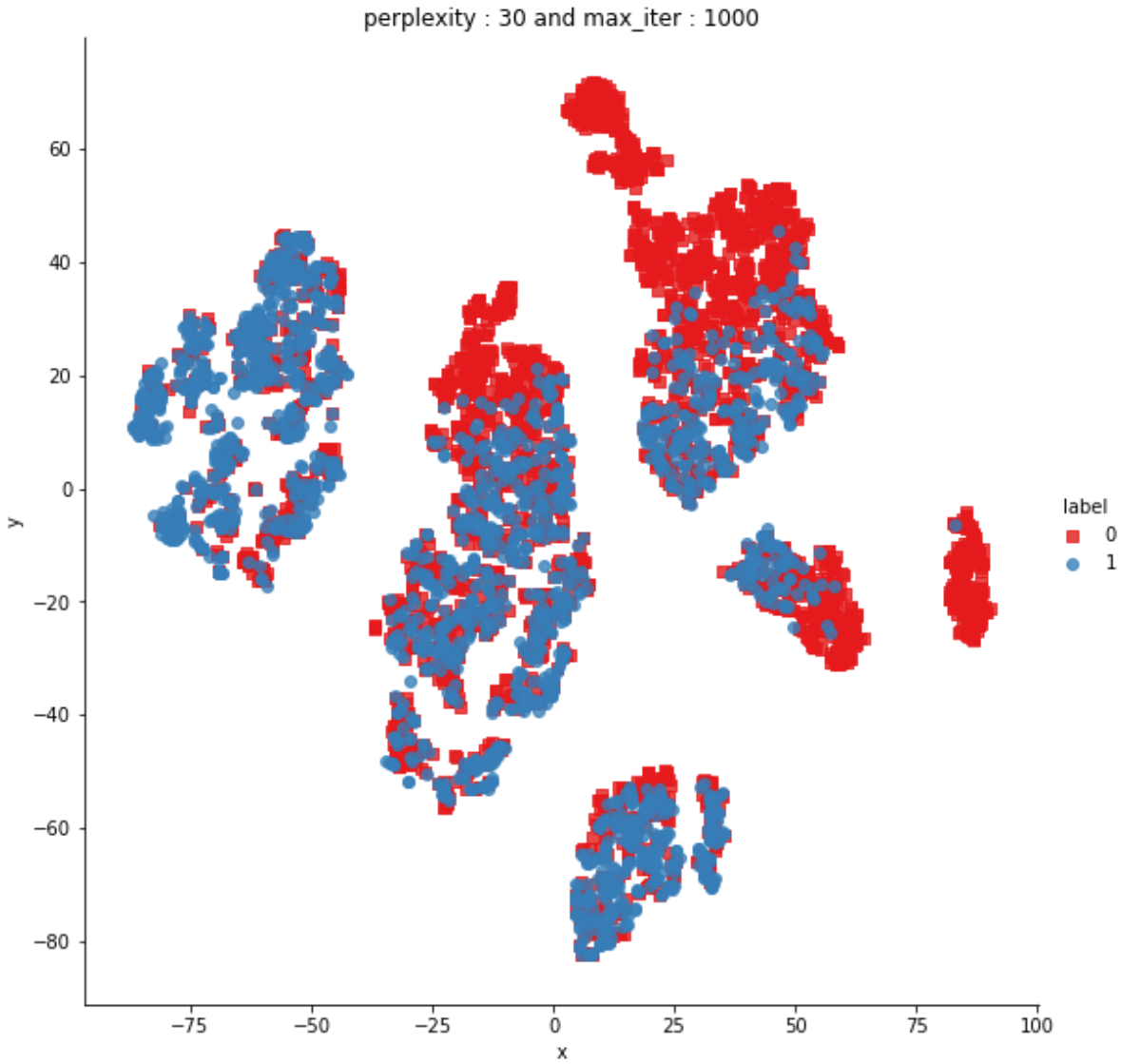
3.5.2 Visualization

In [35]:
```
# Using TSNE for Dimentionality reduction for 15 Features(Generated after clea
ning the data) to 3 dimention

from sklearn.preprocessing import MinMaxScaler

dfp_subsampled = df[0:5000]
X = MinMaxScaler().fit_transform(dfp_subsampled[['cwc_min', 'cwc_max', 'csc_mi
n', 'csc_max' , 'ctc_min' , 'ctc_max' , 'last_word_eq', 'first_word_eq' , 'abs
_len_diff' , 'mean_len' , 'token_set_ratio' , 'token_sort_ratio' ,  'fuzz_rati
o' , 'fuzz_partial_ratio' , 'longest_substr_ratio']])
y = dfp_subsampled['is_duplicate'].values
```

In [36]:
```
tsne2d = TSNE(
    n_components=2,
    init='random', # pca
    random_state=101,
    method='barnes_hut',
    n_iter=1000,
    verbose=2,
    angle=0.5
).fit_transform(X)
```

```
[t-SNE] Computing 91 nearest neighbors...
[t-SNE] Indexed 5000 samples in 0.016s...
[t-SNE] Computed neighbors for 5000 samples in 0.371s...
[t-SNE] Computed conditional probabilities for sample 1000 / 5000
[t-SNE] Computed conditional probabilities for sample 2000 / 5000
[t-SNE] Computed conditional probabilities for sample 3000 / 5000
[t-SNE] Computed conditional probabilities for sample 4000 / 5000
[t-SNE] Computed conditional probabilities for sample 5000 / 5000
[t-SNE] Mean sigma: 0.116557
[t-SNE] Computed conditional probabilities in 0.296s
[t-SNE] Iteration 50: error = 80.9162369, gradient norm = 0.0427600 (50 itera
tions in 2.785s)
[t-SNE] Iteration 100: error = 70.3915100, gradient norm = 0.0108003 (50 iter
ations in 1.969s)
[t-SNE] Iteration 150: error = 68.6126938, gradient norm = 0.0054721 (50 iter
ations in 1.849s)
[t-SNE] Iteration 200: error = 67.7680206, gradient norm = 0.0042246 (50 iter
ations in 1.886s)
[t-SNE] Iteration 250: error = 67.2733459, gradient norm = 0.0037275 (50 iter
ations in 2.016s)
[t-SNE] KL divergence after 250 iterations with early exaggeration: 67.273346
[t-SNE] Iteration 300: error = 1.7734827, gradient norm = 0.0011933 (50 itera
tions in 2.041s)
[t-SNE] Iteration 350: error = 1.3717980, gradient norm = 0.0004826 (50 itera
tions in 1.858s)
[t-SNE] Iteration 400: error = 1.2037998, gradient norm = 0.0002772 (50 itera
tions in 1.850s)
[t-SNE] Iteration 450: error = 1.1133003, gradient norm = 0.0001877 (50 itera
tions in 1.868s)
[t-SNE] Iteration 500: error = 1.0579894, gradient norm = 0.0001429 (50 itera
tions in 1.863s)
[t-SNE] Iteration 550: error = 1.0220573, gradient norm = 0.0001178 (50 itera
tions in 1.936s)
[t-SNE] Iteration 600: error = 0.9990303, gradient norm = 0.0001036 (50 itera
tions in 1.889s)
[t-SNE] Iteration 650: error = 0.9836842, gradient norm = 0.0000951 (50 itera
tions in 1.929s)
[t-SNE] Iteration 700: error = 0.9732341, gradient norm = 0.0000860 (50 itera
tions in 2.056s)
[t-SNE] Iteration 750: error = 0.9649901, gradient norm = 0.0000789 (50 itera
tions in 2.227s)
[t-SNE] Iteration 800: error = 0.9582695, gradient norm = 0.0000745 (50 itera
tions in 2.058s)
[t-SNE] Iteration 850: error = 0.9525222, gradient norm = 0.0000732 (50 itera
tions in 1.957s)
[t-SNE] Iteration 900: error = 0.9479918, gradient norm = 0.0000689 (50 itera
tions in 1.894s)
[t-SNE] Iteration 950: error = 0.9442031, gradient norm = 0.0000651 (50 itera
tions in 1.910s)
[t-SNE] Iteration 1000: error = 0.9408465, gradient norm = 0.0000590 (50 iter
ations in 1.906s)
[t-SNE] KL divergence after 1000 iterations: 0.940847
```

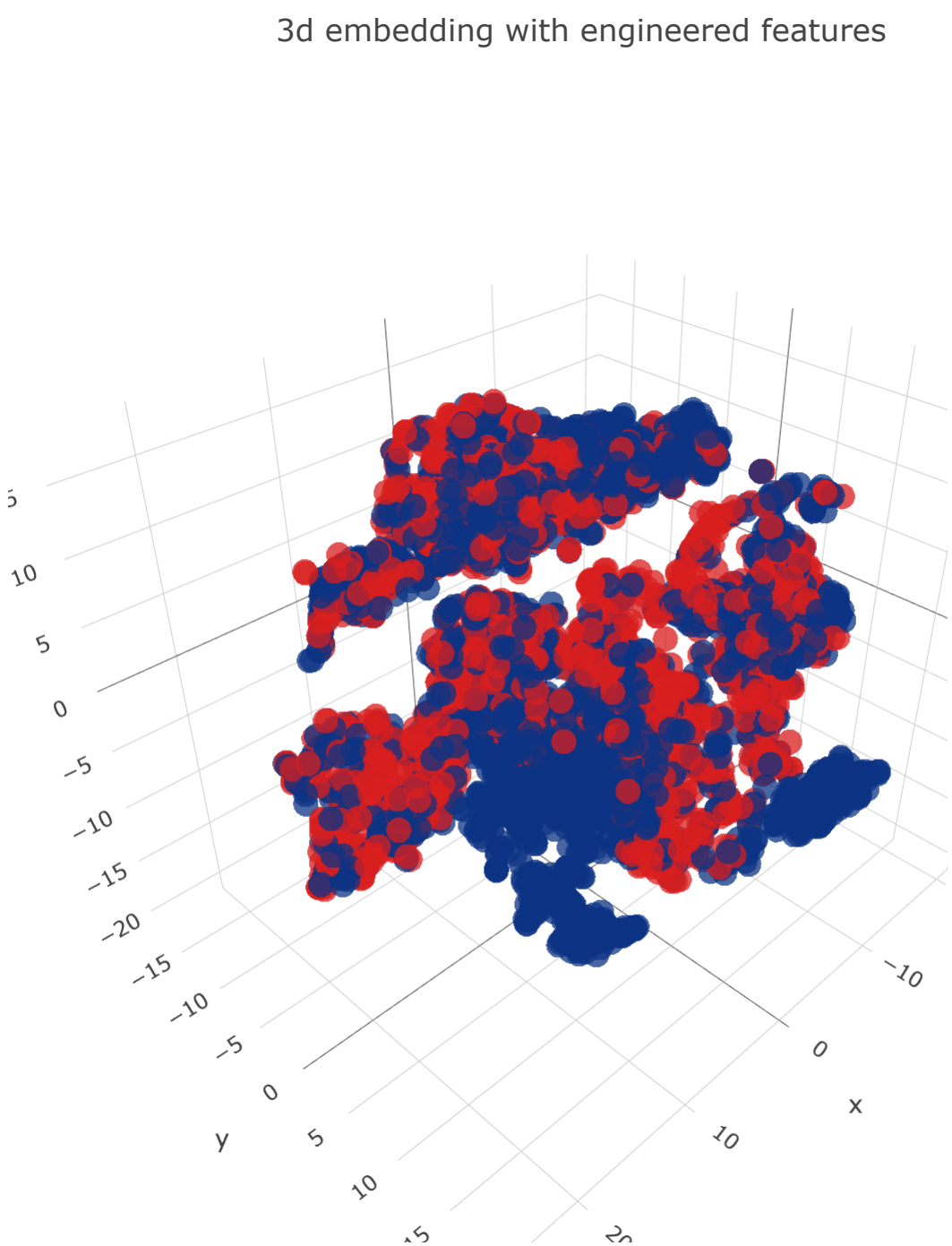
In [37]:
```
df = pd.DataFrame({'x':tsne2d[:,0], 'y':tsne2d[:,1] ,'label':y})

# draw the plot in appropriate place in the grid
sns.lmplot(data=df, x='x', y='y', hue='label', fit_reg=False, size=8,palette=
"Set1",markers=['s','o'])
plt.title("perplexity : {} and max_iter : {}".format(30, 1000))
plt.show()
```



In [38]:
```
from sklearn.manifold import TSNE
tsne3d = TSNE(
    n_components=3,
    init='random', # pca
    random_state=101,
    method='barnes_hut',
    n_iter=1000,
    verbose=2,
    angle=0.5
).fit_transform(X)
```

```
[t-SNE] Computing 91 nearest neighbors...
[t-SNE] Indexed 5000 samples in 0.009s...
[t-SNE] Computed neighbors for 5000 samples in 0.361s...
[t-SNE] Computed conditional probabilities for sample 1000 / 5000
[t-SNE] Computed conditional probabilities for sample 2000 / 5000
[t-SNE] Computed conditional probabilities for sample 3000 / 5000
[t-SNE] Computed conditional probabilities for sample 4000 / 5000
[t-SNE] Computed conditional probabilities for sample 5000 / 5000
[t-SNE] Mean sigma: 0.116557
[t-SNE] Computed conditional probabilities in 0.217s
[t-SNE] Iteration 50: error = 80.3552017, gradient norm = 0.0329941 (50 itera
tions in 9.078s)
[t-SNE] Iteration 100: error = 69.1100388, gradient norm = 0.0034323 (50 iter
ations in 5.171s)
[t-SNE] Iteration 150: error = 67.6163483, gradient norm = 0.0017810 (50 iter
ations in 4.635s)
[t-SNE] Iteration 200: error = 67.0578613, gradient norm = 0.0011246 (50 iter
ations in 4.552s)
[t-SNE] Iteration 250: error = 66.7297821, gradient norm = 0.0009272 (50 iter
ations in 4.234s)
[t-SNE] KL divergence after 250 iterations with early exaggeration: 66.729782
[t-SNE] Iteration 300: error = 1.4978341, gradient norm = 0.0006938 (50 itera
tions in 5.701s)
[t-SNE] Iteration 350: error = 1.1559117, gradient norm = 0.0001985 (50 itera
tions in 7.915s)
[t-SNE] Iteration 400: error = 1.0108488, gradient norm = 0.0000976 (50 itera
tions in 7.737s)
[t-SNE] Iteration 450: error = 0.9391674, gradient norm = 0.0000627 (50 itera
tions in 7.581s)
[t-SNE] Iteration 500: error = 0.9015961, gradient norm = 0.0000508 (50 itera
tions in 7.479s)
[t-SNE] Iteration 550: error = 0.8815936, gradient norm = 0.0000433 (50 itera
tions in 6.754s)
[t-SNE] Iteration 600: error = 0.8682337, gradient norm = 0.0000373 (50 itera
tions in 7.258s)
[t-SNE] Iteration 650: error = 0.8589998, gradient norm = 0.0000360 (50 itera
tions in 7.123s)
[t-SNE] Iteration 700: error = 0.8518325, gradient norm = 0.0000281 (50 itera
tions in 7.311s)
[t-SNE] Iteration 750: error = 0.8455728, gradient norm = 0.0000284 (50 itera
tions in 7.377s)
[t-SNE] Iteration 800: error = 0.8401663, gradient norm = 0.0000264 (50 itera
tions in 7.109s)
[t-SNE] Iteration 850: error = 0.8351609, gradient norm = 0.0000265 (50 itera
tions in 7.232s)
[t-SNE] Iteration 900: error = 0.8312420, gradient norm = 0.0000225 (50 itera
tions in 7.233s)
[t-SNE] Iteration 950: error = 0.8273517, gradient norm = 0.0000231 (50 itera
tions in 7.036s)
[t-SNE] Iteration 1000: error = 0.8240154, gradient norm = 0.0000213 (50 iter
ations in 7.307s)
[t-SNE] KL divergence after 1000 iterations: 0.824015
```

In [39]:
```
trace1 = go.Scatter3d(
    x=tsne3d[:,0],
    y=tsne3d[:,1],
    z=tsne3d[:,2],
    mode='markers',
    marker=dict(
        sizemode='diameter',
        color = y,
        colorscale = 'Portland',
        colorbar = dict(title = 'duplicate'),
        line=dict(color='rgb(255, 255, 255)'),
        opacity=0.75
    )
)

data=[trace1]
layout=dict(height=800, width=800, title='3d embedding with engineered feature
s')
fig=dict(data=data, layout=layout)
py.iplot(fig, filename='3DBubble')
```


3d embedding with engineered features
10
5
0
−5
−10
−15
−20
−15
−10
−5
0
y
5
10
−10
0
x
10

In [ ]:

In [ ]:

3rd NotebookQ_Mean_W2V.ipynb

3.6 Featurizing text data with tfidf weighted word-vectors

In [40]:
```
import pandas as pd
import matplotlib.pyplot as plt
import re
import time
import warnings
import numpy as np
from nltk.corpus import stopwords
from sklearn.preprocessing import normalize
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
warnings.filterwarnings("ignore")
import sys
import os
import pandas as pd
from scipy.sparse import hstack
import numpy as np
from tqdm import tqdm
from tqdm import tqdm_notebook as tqdm1

# exctract word2vec vectors
# https://github.com/explosion/spaCy/issues/1721
# http://landinghub.visualstudio.com/visual-cpp-build-tools
import spacy
from sklearn.model_selection import train_test_split
```

In [41]:
```
# avoid decoding problems
df = pd.read_csv("train.csv")

# encode questions to unicode
# https://stackoverflow.com/a/6812069
# ---------------- python 2 --------------------
# df['question1'] = df['question1'].apply(lambda x: unicode(str(x),"utf-8"))
# df['question2'] = df['question2'].apply(lambda x: unicode(str(x),"utf-8"))
# ---------------- python 3 --------------------
df['question1'] = df['question1'].apply(lambda x: str(x))
df['question2'] = df['question2'].apply(lambda x: str(x))
```

In [42]:
```
df.head()
```

Out[42]:

	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24}[/math] i...	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0

In [43]:
```
#prepro_features_train.csv (Simple Preprocessing Feartures)
#nlp_features_train.csv (NLP Features)
if os.path.isfile('nlp_features_train.csv'):
    dfnlp = pd.read_csv("nlp_features_train.csv",encoding='latin-1')
else:
    print("download nlp_features_train.csv from drive or run previous noteboo
k")

if os.path.isfile('df_fe_without_preprocessing_train.csv'):
    dfppro = pd.read_csv("df_fe_without_preprocessing_train.csv",encoding='lat
in-1')
else:
    print("download df_fe_without_preprocessing_train.csv from drive or run pr
evious notebook")
```

In [44]:
```
dfnlp.head()
```

Out[44]:

	id	qid1	qid2	question1	question2	is_duplicate	cwc_min	cwc_max	csc_min	csc_max
0	0	1	2	what is the step by step guide to invest in sh...	what is the step by step guide to invest in sh...	0	0.999980	0.833319	0.999983	0.999983
1	1	3	4	what is the story of kohinoor koh i noor dia...	what would happen if the indian government sto...	0	0.799984	0.399996	0.749981	0.599988
2	2	5	6	how can i increase the speed of my internet co...	how can internet speed be increased by hacking...	0	0.399992	0.333328	0.399992	0.249997
3	3	7	8	why am i mentally very lonely how can i solve...	find the remainder when math 23 24 math i...	0	0.000000	0.000000	0.000000	0.000000
4	4	9	10	which one dissolve in water quikly sugar salt...	which fish would survive in salt water	0	0.399992	0.199998	0.999950	0.666644

5 rows × 21 columns

In [45]:
```
df1  = dfnlp.merge(dfppro, on='id',how='left')
```

In [46]:
```
df1 = df1.drop(['qid1_x','qid2_x'],axis=1)
# df2 = dfppro.drop(['qid1','qid2','question1','question2','is_duplicate'],axis=1)
df3 = df.drop(['qid1','qid2','question1','question2','is_duplicate'],axis=1)
```

In [47]:
```
df1.shape
```

Out[47]:
```
(404290, 35)
```

In [48]:
```
df4  = df1.merge(df3, on='id',how='left')
```

In [49]:
```
df4 = df4.sample(n=50000)
```

In [50]:
```
df4.shape
```

Out[50]:
```
(50000, 35)
```

In [51]:
```
y_true = df4['is_duplicate_x']
```

In [52]:
```
y_true
```

Out[52]:
```
157111    1
256127    0
275758    0
339967    0
293853    1
203160    1
221034    1
391435    1
122148    1
22932     0
135107    1
321550    1
390562    1
321641    0
221386    1
91708     1
140841    0
40380     0
168337    0
352779    1
174179    1
250050    1
28832     0
192799    1
61976     0
243373    1
178623    1
114748    1
351965    0
168599    1
         ..
347967    0
279093    0
340051    1
232581    1
220107    1
115954    1
23675     0
334251    0
173395    0
368775    0
286823    0
202282    0
110915    1
55110     0
134729    0
134633    1
378291    1
213499    0
386758    0
153304    1
224912    1
402646    0
103678    1
289407    0
110161    0
278486    1
```

```
338644    0
349720    0
252671    0
255335    1
Name: is_duplicate_x, Length: 50000, dtype: int64
```

In [53]:
```
df4 = df4.drop(['is_duplicate_x'],axis=1)
```

In [54]:
```
df4.shape
```

Out[54]:
```
(50000, 34)
```

In [55]:
```
X_train,X_test, y_train, y_test = train_test_split(df4, y_true, stratify=y_tru
e, test_size=0.3)
```

In [56]:
```
X_train.shape
```

Out[56]:
```
(35000, 34)
```

In [57]:
```
X_test.shape
```

Out[57]:
```
(15000, 34)
```

In [58]:
```
X_train.shape[0] + X_test.shape[0]
```

Out[58]:
```
50000
```

In [59]:
```
# Filling the null values with ' '
X_train = X_train.fillna(' ')
nan_rows1 = X_train[X_train.isnull().any(1)]
print (nan_rows1)

# Filling the null values with ' '
X_test = X_test.fillna(' ')
nan_rows2 = X_test[X_test.isnull().any(1)]
print (nan_rows2)
```

```
Empty DataFrame
Columns: [id, question1_x, question2_x, cwc_min, cwc_max, csc_min, csc_max, c
tc_min, ctc_max, last_word_eq, first_word_eq, abs_len_diff, mean_len, token_s
et_ratio, token_sort_ratio, fuzz_ratio, fuzz_partial_ratio, longest_substr_ra
tio, qid1_y, qid2_y, question1_y, question2_y, is_duplicate_y, freq_qid1, fre
q_qid2, q1len, q2len, q1_n_words, q2_n_words, word_Common, word_Total, word_s
hare, freq_q1+q2, freq_q1-q2]
Index: []

[0 rows x 34 columns]
Empty DataFrame
Columns: [id, question1_x, question2_x, cwc_min, cwc_max, csc_min, csc_max, c
tc_min, ctc_max, last_word_eq, first_word_eq, abs_len_diff, mean_len, token_s
et_ratio, token_sort_ratio, fuzz_ratio, fuzz_partial_ratio, longest_substr_ra
tio, qid1_y, qid2_y, question1_y, question2_y, is_duplicate_y, freq_qid1, fre
q_qid2, q1len, q2len, q1_n_words, q2_n_words, word_Common, word_Total, word_s
hare, freq_q1+q2, freq_q1-q2]
Index: []

[0 rows x 34 columns]
```

TFIDFW2V Vectorization on train data:

In [60]:
```
X_train['question1_x'].isnull().values.any()
```

Out[60]:
```
False
```

In [61]:
```
X_train[X_train.isnull().any(1)]
```

Out[61]:

id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min	ctc_max	last_

0 rows × 34 columns

In [62]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
# merge texts
questions_train = list(X_train['question1_x']) + list(X_train['question2_x'])

tfidf_train = TfidfVectorizer(lowercase=False, )
tfidf_train.fit_transform(questions_train)

# # dict key:word and value:tf-idf score
word2tfidf_train = dict(zip(tfidf_train.get_feature_names(), tfidf_train.idf_
))
```

- After we find TF-IDF scores, we convert each question to a weighted average of word2vec vectors by these scores.
- here we use a pre-trained GLOVE model which comes free with "Spacy". https://spacy.io/usage/vectors-similarity (https://spacy.io/usage/vectors-similarity)
- It is trained on Wikipedia and therefore, it is stronger in terms of word semantics.

In [63]:
```
(X_train['question1_x'][12980:12995])
```

Out[63]:
```
118211    what are some reviews of mpix film processing ...
187661    what will be the next step taken by modi to co...
286830    i am a third year civil engineering student  h...
375761    would hillary clinton be a good president for us
778       for what use cases do you think deep learning ...
287448    how do i travel the world without spending my ...
184093               has india ever held a surgical strike
109676           how can i participate in group discussion
344099         which is the best wordpress security plugin
100522           what are direct and inverse relationships
268084    who supplied the san bernardino shooters with ...
10941     two cards were drawn without a replacement fro...
138356       how do i stop mucus from going down my throat
255570    what would hapeen if there are no physical dis...
353098    as a supporter of bernie sanders  why should i...
Name: question1_x, dtype: object
```

In [64]:
```
# en_vectors_web_lg, which includes over 1 million unique vectors.
nlp = spacy.load('en_core_web_sm')

vecs1 = []
# https://github.com/noamraph/tqdm
# tqdm is used to print the progress bar
for qu1 in tqdm1(list(X_train['question1_x'])):
    doc1 = nlp(qu1)
    # 384 is the number of dimensions of vectors
    mean_vec1 = np.zeros([len(doc1), len(doc1[0].vector)])
    for word1 in doc1:
        # word2vec
        vec1 = word1.vector
        # fetch df score
        try:
            idf = word2tfidf_train[str(word1)]
        except:
            idf = 0
        # compute final vec
        mean_vec1 += vec1 * idf
    mean_vec1 = mean_vec1.mean(axis=0)
    vecs1.append(mean_vec1)
X_train['q1_feats_m'] = list(vecs1)
```

In [65]:
```
vecs2 = []
for qu2 in tqdm1(list(X_train['question2_x'])):
    doc2 = nlp(qu2)
    mean_vec2 = np.zeros([len(doc2), len(doc2[0].vector)])
    for word2 in doc2:
        # word2vec
        vec2 = word2.vector
        # fetch df score
        try:
            idf = word2tfidf_train[str(word2)]
        except:
            #print word
            idf = 0
        # compute final vec
        mean_vec2 += vec2 * idf
    mean_vec2 = mean_vec2.mean(axis=0)
    vecs2.append(mean_vec2)
X_train['q2_feats_m'] = list(vecs2)
```

In [66]:
```
# # storing the final features to csv file
# if not os.path.isfile('X_train.csv'):
#     result1  = X_train
#     result1.to_csv('X_train.csv')
# # storing the final features to csv file
# if not os.path.isfile('y_train.csv'):
#     result2  = y_train
#     result2.to_csv('y_train.csv')
```

In [67]:
```
X_train_q1 = pd.DataFrame(X_train.q1_feats_m.values.tolist(), index= X_train.i
ndex)
X_train_q2 = pd.DataFrame(X_train.q2_feats_m.values.tolist(), index= X_train.i
ndex)
```

In [68]:
```
X_train.drop(['q1_feats_m','q2_feats_m'],axis=1)
```

Out[68]:

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
225515	225515	is there a long range wifi antenna that i can ...	what is the best omnidirectional wifi antenna ...	0.599988	0.428565	0.399992	0.181817	0.499995
378666	378666	what is the difference between tortional strai...	what are the differences between stress and st...	0.333322	0.249994	0.799984	0.799984	0.624992
250427	250427	what is an open circuit in electronics	why are resistors required in electronic circu...	0.000000	0.000000	0.333322	0.249994	0.142855
324317	324317	what is blogging and how can i start my own blog	how can we start our own blogs	0.499975	0.333322	0.599988	0.374995	0.571420
274675	274675	can i use a macbook pro charger for my macbook...	will charging a macbook pro using a macbook ai...	0.799984	0.499994	0.333322	0.199996	0.454541
184624	184624	what will be the effect of banning 500 and 1k ...	what will the real estate look like now after ...	0.571420	0.444440	0.666656	0.571420	0.571424
218728	218728	why has quora banned my real name account as f...	why does quora believe my name is fake	0.749981	0.499992	0.499988	0.499988	0.624992
121999	121999	what is the best source of news in india	what are the best sources of fmcg news in india	0.749981	0.599988	0.799984	0.799984	0.777769
110888	110888	can someone help me to find free online course...	what are some useful free online courses for a...	0.714276	0.624992	0.249994	0.199996	0.499996
69717	69717	can an aircraft carrier be sunk	can a modern aircraft carrier be sunk with mod...	0.999967	0.599988	0.666644	0.499988	0.833319
192088	192088	where are the best coworking office space in b...	what is the nicest looking coworking space in ...	0.599988	0.599988	0.499988	0.499988	0.555549
91512	91512	how much of someone own youtube video can i us...	how much of someone own video can i use in my ...	0.999980	0.999980	0.999986	0.999986	0.923070

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
30399	30399	how good is symbiosis institute of design for ...	which degree has more value scope bachelor o...	0.599988	0.214284	0.249994	0.166664	0.444440
137433	137433	what has modi achieved in 2016	what are some great achievements drawbacks o...	0.333322	0.099999	0.333322	0.142855	0.333328
244169	244169	how is proprietary software developed	what is proprietary software	0.999950	0.666644	0.499975	0.499975	0.749981
28950	28950	how do i earn from qoura	how can i earn money on quora	0.499975	0.333322	0.499988	0.499988	0.499992
263118	263118	what does low melting point means	what is the melting point of molecular compounds	0.499988	0.499988	0.499975	0.249994	0.499992
174173	174173	what is the possibility of planet earth runnin...	will drinking water ever run out on planet earth	0.666656	0.666656	0.333322	0.199996	0.555549
369215	369215	what happens if i take a lot of ibuprofen	does my mum hate me for dropping out of high s...	0.000000	0.000000	0.199996	0.166664	0.111110
256407	256407	how can i suppress hunger	how do you suppress hunger	0.999950	0.999950	0.333322	0.333322	0.599988
300264	300264	what is relation between linear velocity and a...	what is the relation between linear and angula...	0.749981	0.749981	0.999975	0.799984	0.777769
142474	142474	find the missing number 2 6 14 30 62 254	can you guys find the missing number	0.749981	0.333330	0.999900	0.333322	0.571420
347532	347532	what are the best reference books for learning...	what is the best book for learning java	0.749981	0.599988	0.749981	0.749981	0.749991
154933	154933	what are your new year own resolutions for 2017	what is your new year resolution for 2017 or g...	0.749981	0.599988	0.599988	0.599988	0.666659
255911	255911	how was steve jobs	was steve jobs himself	0.999950	0.999950	0.499975	0.499975	0.749981
398162	398162	what is a density of water at room temp	what is the density of water	0.999950	0.499988	0.749981	0.599988	0.833319

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
321309	321309	can a silicon product transmit heat	can we transmit a physical object from one pla...	0.249994	0.124998	0.999950	0.249997	0.499992
265335	265335	how india will be benifited by donald trump ow...	what does a donald trump presidency mean for t...	0.571420	0.499994	0.000000	0.000000	0.333331
244423	244423	can 2k rupees notes will be helpful for stop b...	how will the introduction of new 2k rupee note...	0.499994	0.444440	0.249994	0.199996	0.416663
209874	209874	how do i transfer idea balance to my sim	how do i transfer my idea sim balance to my id...	0.999975	0.799984	0.999980	0.999980	0.999989
...	...	...	...	...	...	...	...	...
66702	66702	ia a ca without graduation eligible for cfa	is a ca without graduation eligible to do cfa	0.999980	0.833319	0.499975	0.249994	0.749991
72107	72107	how you joined in tata	i am at manager designation and my company wan...	0.000000	0.000000	0.333322	0.083333	0.199996
108593	108593	is claritin safe for pet allergies	what temperature can kill rabies virus	0.000000	0.000000	0.000000	0.000000	0.000000
323919	323919	my golf group has 8 players how can foursomes...	being a badminton squash player my golf swin...	0.099999	0.083333	0.428565	0.272725	0.235293
346995	346995	what do you know about new world order	do you believe in the new world order	0.749981	0.749981	0.499988	0.499988	0.624992
284882	284882	what is the easiest and most painless suicide ...	what is a painless suicide method	0.999967	0.749981	0.666644	0.399992	0.833319
182664	182664	how to use a camera on an asus laptop	how do use a camera on a laptop what are some...	0.749981	0.749981	0.599988	0.428565	0.666659
73822	73822	what are the criteria for borderline personali...	why did not the dsm 5 rename borderline person...	0.749981	0.499992	0.249994	0.249994	0.499994

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
57912	57912	what are the reasons why american tourists buy...	which is the tv show with more drama and actio...	0.000000	0.000000	0.199996	0.142855	0.083333
98394	98394	what feature of your smartphone do you like th...	what feature in your smartphone do you love th...	0.666644	0.666644	0.857131	0.857131	0.799992
74945	74945	is there any way to send whatsapp messages to ...	is it possible to send a message to whatsapp c...	0.499992	0.428565	0.499988	0.285710	0.416663
29529	29529	i want to study in usa i have completed bcom a...	why does this guy acts like this around me	0.000000	0.000000	0.000000	0.000000	0.000000
352279	352279	what were the best questions asked in an inter...	what are the most interesting questions asked ...	0.749981	0.749981	0.599988	0.499992	0.666659
194841	194841	what is the average sized penis	how long is a normal penis	0.333322	0.333322	0.333322	0.333322	0.333328
114085	114085	how do i hack my husband devices	how can i hack my husband whatsapp	0.666644	0.666644	0.749981	0.749981	0.714276
165629	165629	what is the best tank in world of tanks and why	what is the best tank in wot	0.666644	0.499988	0.999975	0.571420	0.857131
296942	296942	survivalism if you were left stranded in an i...	if you were stranded on a island which celebri...	0.599988	0.333330	0.714276	0.454541	0.571424
279282	279282	what is the meaning of rs 5200 20200 grade p...	what does pay grade g 4 mean	0.399992	0.222220	0.499975	0.199996	0.428565
306955	306955	which is the best laptop to buy in the range o...	which laptop is best in the range of 30k to 40...	0.833319	0.833319	0.999983	0.999983	0.916659
214001	214001	what is the worst corporate merger or acquisit...	what is the most interesting merger in corpora...	0.749981	0.499992	0.799984	0.799984	0.777769
186884	186884	how is leaky gut caused	what is leaky gut syndrome	0.666644	0.666644	0.499975	0.499975	0.599988

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
228506	228506	housekeeping services in chandigarh mohali and...	who is the best doctors in chandigarh mohali ...	0.599988	0.599988	0.999950	0.399992	0.714276
288610	288610	what are your regular simple habits that has l...	what are your regular simple habits that has l...	0.833319	0.833319	0.999986	0.999986	0.857137
373423	373423	by what framework may i take the application l...	how would i take the application lock on iphone	0.799984	0.666656	0.749981	0.599988	0.777769
146596	146596	why do people drink beer whisky even though it...	why do people drink beer when it tastes so bad	0.799984	0.499994	0.749981	0.599988	0.699993
303295	303295	what licenses are required to sell protein sup...	do i need any business licenses to sell hair o...	0.499992	0.428565	0.499988	0.399992	0.499995
74752	74752	can the macbook air run cs go	how do traditional clothes in australia differ...	0.000000	0.000000	0.000000	0.000000	0.000000
73531	73531	would you rather be wile e coyote or road run...	who own more likeable the road runner or wile...	0.833319	0.714276	0.199996	0.199996	0.545450
128560	128560	what are examples of isotopes of the same elem...	why isotopes of the same element are not distr...	0.666644	0.499988	0.799984	0.666656	0.666659
2591	2591	what do you mean by entrepreneur marketing	what do you mean by entrepreneur	0.999950	0.666644	0.999975	0.999975	0.999983

35000 rows × 34 columns

In [69]:
```
if (1):
    X_train_q1['id']=X_train['id']
    X_train_q2['id']=X_train['id']
    df1  = X_train_q1.merge(X_train_q2, on='id',how='left')
    X_train  = X_train.merge(df1, on='id',how='left')
```

In [70]:
```
# from scipy.sparse import hstack
# X_tr = hstack((X_train,X_train_q1,X_train_q2)).tocsr()
# print(X_tr.shape, y_train.shape)
# print("="*100)
```

TFIDFW2V Vectorization on test data:

In [71]:
```
X_test['question1_x'].isnull().values.any()
```

Out[71]:
```
False
```

In [72]:
```
X_test[X_test.isnull().any(1)]
```

Out[72]:

id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min	ctc_max	last_

0 rows × 34 columns

In [73]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
# merge texts
questions_test = list(X_test['question1_x']) + list(X_test['question2_x'])

# tfidf_train = TfidfVectorizer(lowercase=False, )
tfidf_train.transform(questions_test)

# # dict key:word and value:tf-idf score
word2tfidf_test = dict(zip(tfidf_train.get_feature_names(), tfidf_train.idf_))
```

- After we find TF-IDF scores, we convert each question to a weighted average of word2vec vectors by these scores.
- here we use a pre-trained GLOVE model which comes free with "Spacy". https://spacy.io/usage/vectors-similarity (https://spacy.io/usage/vectors-similarity)
- It is trained on Wikipedia and therefore, it is stronger in terms of word semantics.

In [74]:
```
(X_test['question1_x'][12980:12995])
```

Out[74]:
```
234700    how is the word  premonition  used in a sentence
105349                       why is not my phone charging
256855    how do you multiply some cells by a constant i...
160120    which course is better for civil engg student ...
244750    is it illegal to use a fake  edu email to get ...
216089                     how do i get over a good breakup
171608    what are some of the best love proposal lines ...
78271     i am 17 now  how can i  earn  my first house o...
295483    what has been your most embarrassing moment fr...
349432    i have learnt nothing during my ca articleship...
244878    how would you compare ncat and mat exams and i...
203296    what are some ways i can make money as a 13 ye...
32907        what is the latest google update for seo 2016
157379                    how could i start my own business
353797                    what is this on my shoulder blade
Name: question1_x, dtype: object
```

In [75]:
```
# en_vectors_web_lg, which includes over 1 million unique vectors.
nlp = spacy.load('en_core_web_sm')

vecs1 = []
# https://github.com/noamraph/tqdm
# tqdm is used to print the progress bar
for qu1 in tqdm1(list(X_test['question1_x'])):
    doc1 = nlp(qu1)
    # 384 is the number of dimensions of vectors
    mean_vec1 = np.zeros([len(doc1), len(doc1[0].vector)])
    for word1 in doc1:
        # word2vec
        vec1 = word1.vector
        # fetch df score
        try:
            idf = word2tfidf_test[str(word1)]
        except:
            idf = 0
        # compute final vec
        mean_vec1 += vec1 * idf
    mean_vec1 = mean_vec1.mean(axis=0)
    vecs1.append(mean_vec1)
X_test['q1_feats_m'] = list(vecs1)
```

In [76]:
```
vecs2 = []
for qu2 in tqdm1(list(X_test['question2_x'])):
    doc2 = nlp(qu2)
    mean_vec2 = np.zeros([len(doc2), len(doc2[0].vector)])
    for word2 in doc2:
        # word2vec
        vec2 = word2.vector
        # fetch df score
        try:
            idf = word2tfidf_test[str(word2)]
        except:
            #print word
            idf = 0
        # compute final vec
        mean_vec2 += vec2 * idf
    mean_vec2 = mean_vec2.mean(axis=0)
    vecs2.append(mean_vec2)
X_test['q2_feats_m'] = list(vecs2)
```

In [77]:
```
# if not os.path.isfile('X_test.csv'):
#     result3  = X_test
#     result3.to_csv('X_test.csv')
# # storing the final features to csv file
# if not os.path.isfile('y_test.csv'):
#     result4  = y_test
#     result4.to_csv('y_test.csv')
```

In [78]:
```
X_test_q1 = pd.DataFrame(X_test.q1_feats_m.values.tolist(), index= X_test.inde
x)
X_test_q2 = pd.DataFrame(X_test.q2_feats_m.values.tolist(), index= X_test.inde
x)
```

In [79]:
```
X_test.drop(['q1_feats_m','q2_feats_m'],axis=1)
```

Out[79]:

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
159393	159393	looking back what was the most important risk...	what is the most embarrassing music you have e...	0.249997	0.199998	0.714276	0.555549	0.466664
84324	84324	what are some good one line jokes	what are the best one line jokes	0.749981	0.749981	0.666644	0.666644	0.714276
261269	261269	what are the differences between a belgian mal...	what is the difference between a belgian malin...	0.799984	0.799984	0.833319	0.833319	0.749994
15668	15668	how do i make an android application using pyt...	how do i get started with android application ...	0.399992	0.399992	0.749981	0.749981	0.555549
171958	171958	what is the best ola hack for unlimited ola mo...	can you give me that coupon code by which i ca...	0.571420	0.333331	0.624992	0.333331	0.499997
354064	354064	is species singular or plural how do you sa...	how is the word crisis spelled in plural form	0.399992	0.285710	0.749981	0.428565	0.555549
258161	258161	if you had one extra time in a day what would ...	what would you do with extra time if one day i...	0.999980	0.714276	0.666656	0.571420	0.692302
83575	83575	will science answer why life came to be	is there anywhere that i can find the history ...	0.499988	0.181817	0.499988	0.199998	0.499994
278936	278936	does china have hard water if so what proble...	will china run out of water	0.666644	0.399992	0.000000	0.000000	0.333328
121078	121078	where do i find ca final law amendments for no...	what are the applicable amendments for nov 201...	0.714276	0.555549	0.249994	0.199996	0.545450
220465	220465	what is the translation in english for tum ap...	can anyone help me in translating a paragraph ...	0.166664	0.124998	0.333328	0.285710	0.249998
213027	213027	where do i find the hood latch on a 1990 ford ...	where do i find the hood latch on a 1990 thund...	0.999980	0.833319	0.999983	0.999983	0.999991

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
126106	126106	how will trump own presidency affect indian st...	what does a trump presidency mean for indian s...	0.499994	0.399996	0.571420	0.399996	0.470585
120994	120994	should i study chemical engineering because i ...	is it a good idea to take chemical engineering...	0.833319	0.714276	0.333328	0.285710	0.538457
73458	73458	is death real is there evidence that suggests...	what does it mean to own an ngo can you use t...	0.000000	0.000000	0.222220	0.181817	0.090909
89859	89859	why should i study biotechnology	why do we need to study biotechnology	0.999950	0.666644	0.333322	0.249994	0.599988
3903	3903	who is more desirable deepika padukone or priy...	who is the better actress between priyanka cho...	0.799984	0.666656	0.499988	0.399992	0.666659
231393	231393	what is the hardest jobs in the world	what is the hardest job in the world	0.666644	0.666644	0.999975	0.999975	0.749991
170302	170302	what are the best ways to make money online	what are some easy ways to make done extra mon...	0.799984	0.571420	0.749981	0.749981	0.777769
378307	378307	who has the highest iq	who has the highest recorded iq of all time	0.999950	0.499988	0.999967	0.599988	0.999980
398592	398592	what is the best way to prepare steak	what is the best way to cook steak	0.749981	0.749981	0.999975	0.999975	0.874989
175688	175688	how big is a photon compared to an atom	how big is a photon	0.999950	0.499988	0.999967	0.599988	0.999980
194963	194963	could our universe actually be a computer prog...	is our world a computer program why	0.666644	0.399992	0.666644	0.499988	0.571420
228998	228998	to print an artwork in high quality on a4 pape...	how much does it cost a company with a high sp...	0.249998	0.230767	0.555549	0.384612	0.347825
326013	326013	what is the smartest thing you have seen an an...	what is the smartest thing you have seen a chi...	0.749981	0.749981	0.857131	0.857131	0.818174

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
375875	375875	how do i connect xampp with zend studio	can i study 2nd pg through distance mode and p...	0.000000	0.000000	0.499988	0.222220	0.249997
10609	10609	what kind of freight train types can we see	what kinds of cats can easily be trained	0.000000	0.000000	0.749981	0.749981	0.374995
314304	314304	what is the biggest black hole we know	what was the size of the biggest black hole	0.749981	0.749981	0.499988	0.499988	0.624992
294641	294641	where can i get wide range of floor tile wall...	where can i found very durable and easy to cle...	0.428565	0.333330	0.833319	0.714276	0.571424
154730	154730	can i really jailbreak ios 10 0 2	how do i jailbreak ios 10 0 1	0.799984	0.666656	0.499975	0.333322	0.624992
...	...	...	...	...	...	...	...	...
154267	154267	how do you change a phone number through strai...	does changing your phone number remove a tap o...	0.399992	0.399992	0.249994	0.199996	0.299997
112639	112639	how can i improve on javascript	how should i improve my javascript skills	0.999950	0.666644	0.499988	0.499988	0.666656
123266	123266	will china ever become a democracy	will china become a democracy	0.999967	0.749981	0.999950	0.999950	0.999980
124968	124968	i am startup from patna bihar how do i conne...	how can i make the best experience when using ...	0.000000	0.000000	0.399992	0.249997	0.181817
226048	226048	what happened during new year own eve celebrat...	what really happened at mg road bangalore on ...	0.833319	0.555549	0.499988	0.499988	0.699993
74831	74831	why can not the us and russia just be friends	why can not the us become allies with russia	0.666644	0.499988	0.799984	0.571420	0.666659
213667	213667	how long can you bake chicken thighs at 400 de...	how long should i boil chicken thighs for	0.749981	0.374995	0.249994	0.199996	0.499994

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
252025	252025	if all the humans in the world were distribute...	if you could get everybody on earth to agree o...	0.249997	0.166665	0.285710	0.285710	0.249998
245061	245061	from an employers perspective how does wagewo...	what is the hyperloop how does it work	0.499975	0.249994	0.499988	0.333328	0.374995
222871	222871	my husband hates my family specially my father...	my wife hates my family specially my father w...	0.799984	0.799984	0.999980	0.999980	0.749994
263285	263285	what are some things new employees should know...	what are some things new employees should know...	0.888879	0.888879	0.999986	0.999986	0.937494
144622	144622	how will you explain web services to a beginner	can anyone help me to understand bacnet web se...	0.749981	0.299997	0.199996	0.142855	0.444440
271661	271661	what is the effect on the three financial stat...	what is your review of accounts payable	0.666644	0.222220	0.499988	0.285710	0.571420
347411	347411	what is the fastest way to get smart	how can i get smart	0.999950	0.499988	0.000000	0.000000	0.399992
356716	356716	what should i do when someone marks my questio...	why every time quora marks my question as nee...	0.399992	0.285710	0.666644	0.285710	0.399996
233464	233464	will hillary clinton trigger ww3	will hillary clinton start a nuclear war with ...	0.499988	0.333328	0.999900	0.333322	0.599988
118320	118320	i am in the ece branch how can i clarify the ...	where can i ask all my technical doubts relate...	0.571420	0.444440	0.333328	0.285710	0.461535
296809	296809	i am in 10 class and i want to prepare for iit...	i am in class 10 in cbse school so what should...	0.499992	0.499992	0.874989	0.777769	0.624996
303477	303477	how do european nobles live today	i have iata catification and now want to integ...	0.000000	0.000000	0.000000	0.000000	0.000000

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
231698	231698	banning 500 and 1k rupee notes is appreciated ...	why is 500 and 1k rupee notes discontinued	0.799984	0.363633	0.999967	0.428565	0.874989
301711	301711	life how can i motivate myself for a long time	how do i read study with focus concentration a...	0.000000	0.000000	0.399992	0.333328	0.199998
267963	267963	where can i found local directories in australia	where is the best place to visit	0.000000	0.000000	0.249994	0.249994	0.142855
253440	253440	what do astronomers use scientific notation for	why do astronomers use scientific notation wh...	0.999975	0.799984	0.333322	0.199996	0.714276
274848	274848	where can i get faster skip bin hire services ...	where can i get fast and convenient delivery o...	0.571420	0.499994	0.999975	0.571420	0.727266
308876	308876	which will be the best day of your life	what was the best day of your life and what h...	0.999967	0.749981	0.499992	0.499992	0.666659
255878	255878	what are ways to start a conversation	how do i start a conversation	0.999950	0.666644	0.249994	0.249994	0.499992
36717	36717	how can one prepare for writing copy for digit...	how can i prepare for writing copy for digital...	0.999980	0.833319	0.999967	0.749981	0.799992
297608	297608	what are the main causes of global warming	what are factors that stimulate global warming...	0.499988	0.399992	0.499988	0.285710	0.499994
46501	46501	is it normal to still be dealing with body ach...	why do i still get winded after climbing 9 fli...	0.333330	0.199999	0.428565	0.272725	0.374998
323608	323608	who own your favorite writer and what book of ...	what is your favorite book of al time	0.499988	0.333328	0.749981	0.333330	0.624992

15000 rows × 34 columns

In [80]:
```
if (1):
    X_test_q1['id']=X_test['id']
    X_test_q2['id']=X_test['id']
    df2  = X_test_q1.merge(X_test_q2, on='id',how='left')
    X_test  = X_test.merge(df2, on='id',how='left')
```

In [81]:
```
X_train.shape
```

Out[81]:
```
(35000, 228)
```

In [82]:
```
X_test.shape
```

Out[82]:
```
(15000, 228)
```

Storing final features and their targets with respective splitting

In [67]:
```
X_train.to_pickle("X_train.txt")
y_train.to_pickle("y_train.txt")
X_test.to_pickle("X_test.txt")
y_test.to_pickle("y_test.txt")
```

In [4]:
```
X_train = pd.read_pickle("X_train.txt")
y_train = pd.read_pickle("y_train.txt")
X_test = pd.read_pickle("X_test.txt")
y_test = pd.read_pickle("y_test.txt")
```

In [83]:
```
X_train = X_train.drop(['id','question1_x','question2_x','q1_feats_m','q2_feat
s_m','qid1_y','qid2_y','question1_y','question2_y','is_duplicate_y'],axis=1)
# X_train.drop(X_train.index[0], inplace=True)
X_test = X_test.drop(['id','question1_x','question2_x','q1_feats_m','q2_feats_
m','qid1_y','qid2_y','question1_y','question2_y','is_duplicate_y'],axis=1)
# X_test.drop(X_test.index[0], inplace=True)
```

In [84]:
```
X_train
```

Out[84]:

	cwc_min	cwc_max	csc_min	csc_max	ctc_min	ctc_max	last_word_eq	first_word_eq
0	0.599988	0.428565	0.399992	0.181817	0.499995	0.249999	1.0	0.0
1	0.333322	0.249994	0.799984	0.799984	0.624992	0.499995	1.0	1.0
2	0.000000	0.000000	0.333322	0.249994	0.142855	0.142855	0.0	0.0
3	0.499975	0.333322	0.599988	0.374995	0.571420	0.363633	0.0	0.0
4	0.799984	0.499994	0.333322	0.199996	0.454541	0.384612	0.0	0.0
5	0.571420	0.444440	0.666656	0.571420	0.571424	0.470585	0.0	1.0
6	0.749981	0.499992	0.499988	0.499988	0.624992	0.499995	1.0	1.0
7	0.749981	0.599988	0.799984	0.799984	0.777769	0.699993	1.0	1.0
8	0.714276	0.624992	0.249994	0.199996	0.499996	0.499996	0.0	0.0
9	0.999967	0.599988	0.666644	0.499988	0.833319	0.499995	0.0	1.0
10	0.599988	0.599988	0.499988	0.499988	0.555549	0.555549	1.0	0.0
11	0.999980	0.999980	0.999986	0.999986	0.923070	0.857137	1.0	1.0
12	0.599988	0.214284	0.249994	0.166664	0.444440	0.173912	0.0	0.0
13	0.333322	0.099999	0.333322	0.142855	0.333328	0.105263	0.0	1.0
14	0.999950	0.666644	0.499975	0.499975	0.749981	0.599988	0.0	0.0
15	0.499975	0.333322	0.499988	0.499988	0.499992	0.428565	0.0	1.0
16	0.499988	0.499988	0.499975	0.249994	0.499992	0.374995	0.0	1.0
17	0.666656	0.666656	0.333322	0.199996	0.555549	0.416663	0.0	0.0
18	0.000000	0.000000	0.199996	0.166664	0.111110	0.090908	0.0	0.0
19	0.999950	0.999950	0.333322	0.333322	0.599988	0.599988	1.0	1.0
20	0.749981	0.749981	0.999975	0.799984	0.777769	0.777769	0.0	1.0
21	0.749981	0.333330	0.999900	0.333322	0.571420	0.399996	0.0	0.0
22	0.749981	0.599988	0.749981	0.749981	0.749991	0.666659	1.0	1.0
23	0.749981	0.599988	0.599988	0.599988	0.666659	0.499996	1.0	1.0
24	0.999950	0.999950	0.499975	0.499975	0.749981	0.749981	0.0	0.0
25	0.999950	0.499988	0.749981	0.599988	0.833319	0.555549	0.0	1.0
26	0.249994	0.124998	0.999950	0.249997	0.499992	0.166666	0.0	1.0
27	0.571420	0.499994	0.000000	0.000000	0.333331	0.285712	0.0	0.0
28	0.499994	0.444440	0.249994	0.199996	0.416663	0.333331	0.0	0.0
29	0.999975	0.799984	0.999980	0.999980	0.999989	0.692302	0.0	1.0
...	...	...	...	...	...	...	...	...
34970	0.999980	0.833319	0.499975	0.249994	0.749991	0.666659	1.0	0.0
34971	0.000000	0.000000	0.333322	0.083333	0.199996	0.033333	0.0	0.0
34972	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.0
34973	0.099999	0.083333	0.428565	0.272725	0.235293	0.159999	0.0	0.0

	cwc_min	cwc_max	csc_min	csc_max	ctc_min	ctc_max	last_word_eq	first_word_eq
34974	0.749981	0.749981	0.499988	0.499988	0.624992	0.624992	1.0	0.0
34975	0.999967	0.749981	0.666644	0.399992	0.833319	0.555549	1.0	1.0
34976	0.749981	0.749981	0.599988	0.428565	0.666659	0.499996	0.0	1.0
34977	0.749981	0.499992	0.249994	0.249994	0.499994	0.399996	1.0	0.0
34978	0.000000	0.000000	0.199996	0.142855	0.083333	0.062500	0.0	0.0
34979	0.666644	0.666644	0.857131	0.857131	0.799992	0.799992	1.0	1.0
34980	0.499992	0.428565	0.499988	0.285710	0.416663	0.333331	0.0	1.0
34981	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.0
34982	0.749981	0.749981	0.599988	0.499992	0.666659	0.599994	1.0	1.0
34983	0.333322	0.333322	0.333322	0.333322	0.333328	0.333328	1.0	0.0
34984	0.666644	0.666644	0.749981	0.749981	0.714276	0.714276	0.0	1.0
34985	0.666644	0.499988	0.999975	0.571420	0.857131	0.545450	0.0	1.0
34986	0.599988	0.333330	0.714276	0.454541	0.571424	0.363635	0.0	0.0
34987	0.399992	0.222220	0.499975	0.199996	0.428565	0.187499	0.0	1.0
34988	0.833319	0.833319	0.999983	0.999983	0.916659	0.785709	0.0	1.0
34989	0.749981	0.499992	0.799984	0.799984	0.777769	0.636358	1.0	1.0
34990	0.666644	0.666644	0.499975	0.499975	0.599988	0.599988	0.0	0.0
34991	0.599988	0.599988	0.999950	0.399992	0.714276	0.499995	1.0	0.0
34992	0.833319	0.833319	0.999986	0.999986	0.857137	0.857137	1.0	1.0
34993	0.799984	0.666656	0.749981	0.599988	0.777769	0.636358	1.0	0.0
34994	0.799984	0.499994	0.749981	0.599988	0.699993	0.583328	0.0	1.0
34995	0.499992	0.428565	0.499988	0.399992	0.499995	0.416663	1.0	0.0
34996	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.0
34997	0.833319	0.714276	0.199996	0.199996	0.545450	0.499996	0.0	0.0
34998	0.666644	0.499988	0.799984	0.666656	0.666659	0.599994	0.0	0.0
34999	0.999950	0.666644	0.999975	0.999975	0.999983	0.857131	0.0	1.0

35000 rows × 218 columns

In [85]:
```
X_test
```

Out[85]:

	cwc_min	cwc_max	csc_min	csc_max	ctc_min	ctc_max	last_word_eq	first_word_eq
0	0.249997	0.199998	0.714276	0.555549	0.466664	0.368419	0.0	0.0
1	0.749981	0.749981	0.666644	0.666644	0.714276	0.714276	1.0	1.0
2	0.799984	0.799984	0.833319	0.833319	0.749994	0.749994	1.0	1.0
3	0.399992	0.399992	0.749981	0.749981	0.555549	0.555549	0.0	1.0
4	0.571420	0.333331	0.624992	0.333331	0.499997	0.290322	0.0	0.0
5	0.399992	0.285710	0.749981	0.428565	0.555549	0.333331	0.0	0.0
6	0.999980	0.714276	0.666656	0.571420	0.692302	0.692302	0.0	0.0
7	0.499988	0.181817	0.499988	0.199998	0.499994	0.181817	1.0	0.0
8	0.666644	0.399992	0.000000	0.000000	0.333328	0.166665	0.0	0.0
9	0.714276	0.555549	0.249994	0.199996	0.545450	0.428568	0.0	0.0
10	0.166664	0.124998	0.333328	0.285710	0.249998	0.187499	0.0	0.0
11	0.999980	0.833319	0.999983	0.999983	0.999991	0.916659	1.0	1.0
12	0.499994	0.399996	0.571420	0.399996	0.470585	0.444442	0.0	0.0
13	0.833319	0.714276	0.333328	0.285710	0.538457	0.499996	1.0	0.0
14	0.000000	0.000000	0.222220	0.181817	0.090909	0.086956	0.0	0.0
15	0.999950	0.666644	0.333322	0.249994	0.599988	0.428565	1.0	1.0
16	0.799984	0.666656	0.499988	0.399992	0.666659	0.545450	0.0	1.0
17	0.666644	0.666644	0.999975	0.999975	0.749991	0.749991	1.0	1.0
18	0.799984	0.571420	0.749981	0.749981	0.777769	0.636358	1.0	1.0
19	0.999950	0.499988	0.999967	0.599988	0.999980	0.555549	0.0	1.0
20	0.749981	0.749981	0.999975	0.999975	0.874989	0.874989	1.0	1.0
21	0.999950	0.499988	0.999967	0.599988	0.999980	0.555549	0.0	1.0
22	0.666644	0.399992	0.666644	0.499988	0.571420	0.499994	0.0	0.0
23	0.249998	0.230767	0.555549	0.384612	0.347825	0.285713	0.0	0.0
24	0.749981	0.749981	0.857131	0.857131	0.818174	0.818174	1.0	1.0
25	0.000000	0.000000	0.499988	0.222220	0.249997	0.105263	0.0	0.0
26	0.000000	0.000000	0.749981	0.749981	0.374995	0.333330	0.0	1.0
27	0.749981	0.749981	0.499988	0.499988	0.624992	0.555549	0.0	1.0
28	0.428565	0.333330	0.833319	0.714276	0.571424	0.499997	1.0	1.0
29	0.799984	0.666656	0.499975	0.333322	0.624992	0.624992	0.0	0.0
...	...	...	...	...	...	...	...	...
14970	0.399992	0.399992	0.249994	0.199996	0.299997	0.272725	0.0	0.0
14971	0.999950	0.666644	0.499988	0.499988	0.666656	0.571420	0.0	1.0
14972	0.999967	0.749981	0.999950	0.999950	0.999980	0.833319	1.0	1.0
14973	0.000000	0.000000	0.399992	0.249997	0.181817	0.111110	0.0	0.0

	cwc_min	cwc_max	csc_min	csc_max	ctc_min	ctc_max	last_word_eq	first_word_eq
14974	0.833319	0.555549	0.499988	0.499988	0.699993	0.538457	0.0	1.0
14975	0.666644	0.499988	0.799984	0.571420	0.666659	0.599994	0.0	1.0
14976	0.749981	0.374995	0.249994	0.199996	0.499994	0.307690	0.0	1.0
14977	0.249997	0.166665	0.285710	0.285710	0.249998	0.181817	0.0	1.0
14978	0.499975	0.249994	0.499988	0.333328	0.374995	0.374995	1.0	0.0
14979	0.799984	0.799984	0.999980	0.999980	0.749994	0.749994	1.0	1.0
14980	0.888879	0.888879	0.999986	0.999986	0.937494	0.937494	0.0	1.0
14981	0.749981	0.299997	0.199996	0.142855	0.444440	0.153846	0.0	0.0
14982	0.666644	0.222220	0.499988	0.285710	0.571420	0.235293	0.0	1.0
14983	0.999950	0.499988	0.000000	0.000000	0.399992	0.249997	1.0	0.0
14984	0.399992	0.285710	0.666644	0.285710	0.399996	0.333331	0.0	0.0
14985	0.499988	0.333328	0.999900	0.333322	0.599988	0.333330	0.0	1.0
14986	0.571420	0.444440	0.333328	0.285710	0.461535	0.299999	0.0	0.0
14987	0.499992	0.499992	0.874989	0.777769	0.624996	0.588232	0.0	1.0
14988	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.0
14989	0.799984	0.363633	0.999967	0.428565	0.874989	0.318180	0.0	0.0
14990	0.000000	0.000000	0.399992	0.333328	0.199998	0.166665	0.0	0.0
14991	0.000000	0.000000	0.249994	0.249994	0.142855	0.124998	0.0	1.0
14992	0.999975	0.799984	0.333322	0.199996	0.714276	0.499995	0.0	0.0
14993	0.571420	0.499994	0.999975	0.571420	0.727266	0.533330	1.0	1.0
14994	0.999967	0.749981	0.499992	0.499992	0.666659	0.545450	0.0	0.0
14995	0.999950	0.666644	0.249994	0.249994	0.499992	0.428565	1.0	0.0
14996	0.999980	0.833319	0.999967	0.749981	0.799992	0.799992	1.0	1.0
14997	0.499988	0.399992	0.499988	0.285710	0.499994	0.333331	0.0	1.0
14998	0.333330	0.199999	0.428565	0.272725	0.374998	0.214285	0.0	0.0
14999	0.499988	0.333328	0.749981	0.333330	0.624992	0.333331	0.0	0.0

15000 rows × 218 columns

In [86]:
```
# after we read from sql table each entry was read it as a string
# we convert all the features into numaric before we apply any model
cols = list(X_train.columns)
for i in cols:
#     data[i] = data[i].apply(pd.to_numeric)
    print(i)
```

```
cwc_min
cwc_max
csc_min
csc_max
ctc_min
ctc_max
last_word_eq
first_word_eq
abs_len_diff
mean_len
token_set_ratio
token_sort_ratio
fuzz_ratio
fuzz_partial_ratio
longest_substr_ratio
freq_qid1
freq_qid2
q1len
q2len
q1_n_words
q2_n_words
word_Common
word_Total
word_share
freq_q1+q2
freq_q1-q2
0_x
1_x
2_x
3_x
4_x
5_x
6_x
7_x
8_x
9_x
10_x
11_x
12_x
13_x
14_x
15_x
16_x
17_x
18_x
19_x
20_x
21_x
22_x
23_x
24_x
25_x
26_x
27_x
28_x
29_x
30_x
```

```
31_x
32_x
33_x
34_x
35_x
36_x
37_x
38_x
39_x
40_x
41_x
42_x
43_x
44_x
45_x
46_x
47_x
48_x
49_x
50_x
51_x
52_x
53_x
54_x
55_x
56_x
57_x
58_x
59_x
60_x
61_x
62_x
63_x
64_x
65_x
66_x
67_x
68_x
69_x
70_x
71_x
72_x
73_x
74_x
75_x
76_x
77_x
78_x
79_x
80_x
81_x
82_x
83_x
84_x
85_x
86_x
87_x
```

```
88_x
89_x
90_x
91_x
92_x
93_x
94_x
95_x
0_y
1_y
2_y
3_y
4_y
5_y
6_y
7_y
8_y
9_y
10_y
11_y
12_y
13_y
14_y
15_y
16_y
17_y
18_y
19_y
20_y
21_y
22_y
23_y
24_y
25_y
26_y
27_y
28_y
29_y
30_y
31_y
32_y
33_y
34_y
35_y
36_y
37_y
38_y
39_y
40_y
41_y
42_y
43_y
44_y
45_y
46_y
47_y
48_y
```

```
49_y
50_y
51_y
52_y
53_y
54_y
55_y
56_y
57_y
58_y
59_y
60_y
61_y
62_y
63_y
64_y
65_y
66_y
67_y
68_y
69_y
70_y
71_y
72_y
73_y
74_y
75_y
76_y
77_y
78_y
79_y
80_y
81_y
82_y
83_y
84_y
85_y
86_y
87_y
88_y
89_y
90_y
91_y
92_y
93_y
94_y
95_y
```

In [87]:
```
# after we read from sql table each entry was read it as a string
# we convert all the features into numaric before we apply any model
cols = list(X_test.columns)
for i in cols:
#     data[i] = data[i].apply(pd.to_numeric)
    print(i)
```

```
cwc_min
cwc_max
csc_min
csc_max
ctc_min
ctc_max
last_word_eq
first_word_eq
abs_len_diff
mean_len
token_set_ratio
token_sort_ratio
fuzz_ratio
fuzz_partial_ratio
longest_substr_ratio
freq_qid1
freq_qid2
q1len
q2len
q1_n_words
q2_n_words
word_Common
word_Total
word_share
freq_q1+q2
freq_q1-q2
0_x
1_x
2_x
3_x
4_x
5_x
6_x
7_x
8_x
9_x
10_x
11_x
12_x
13_x
14_x
15_x
16_x
17_x
18_x
19_x
20_x
21_x
22_x
23_x
24_x
25_x
26_x
27_x
28_x
29_x
30_x
```

```
31_x
32_x
33_x
34_x
35_x
36_x
37_x
38_x
39_x
40_x
41_x
42_x
43_x
44_x
45_x
46_x
47_x
48_x
49_x
50_x
51_x
52_x
53_x
54_x
55_x
56_x
57_x
58_x
59_x
60_x
61_x
62_x
63_x
64_x
65_x
66_x
67_x
68_x
69_x
70_x
71_x
72_x
73_x
74_x
75_x
76_x
77_x
78_x
79_x
80_x
81_x
82_x
83_x
84_x
85_x
86_x
87_x
```

```
88_x
89_x
90_x
91_x
92_x
93_x
94_x
95_x
0_y
1_y
2_y
3_y
4_y
5_y
6_y
7_y
8_y
9_y
10_y
11_y
12_y
13_y
14_y
15_y
16_y
17_y
18_y
19_y
20_y
21_y
22_y
23_y
24_y
25_y
26_y
27_y
28_y
29_y
30_y
31_y
32_y
33_y
34_y
35_y
36_y
37_y
38_y
39_y
40_y
41_y
42_y
43_y
44_y
45_y
46_y
47_y
48_y
```

```
49_y
50_y
51_y
52_y
53_y
54_y
55_y
56_y
57_y
58_y
59_y
60_y
61_y
62_y
63_y
64_y
65_y
66_y
67_y
68_y
69_y
70_y
71_y
72_y
73_y
74_y
75_y
76_y
77_y
78_y
79_y
80_y
81_y
82_y
83_y
84_y
85_y
86_y
87_y
88_y
89_y
90_y
91_y
92_y
93_y
94_y
95_y
```

4th Notebook:ML_models.ipynb

In [88]:
```
import pandas as pd
import matplotlib.pyplot as plt
import re
import time
import warnings
import sqlite3
from sqlalchemy import create_engine # database connection
import csv
import os
warnings.filterwarnings("ignore")
import datetime as dt
import numpy as np
from nltk.corpus import stopwords
from sklearn.decomposition import TruncatedSVD
from sklearn.preprocessing import normalize
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.manifold import TSNE
import seaborn as sns
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics.classification import accuracy_score, log_loss
from sklearn.feature_extraction.text import TfidfVectorizer
from collections import Counter
from scipy.sparse import hstack
from sklearn.multiclass import OneVsRestClassifier
from sklearn.svm import SVC
from sklearn.model_selection import StratifiedKFold
from collections import Counter, defaultdict
from sklearn.calibration import CalibratedClassifierCV
from sklearn.naive_bayes import MultinomialNB
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
import math
from sklearn.metrics import normalized_mutual_info_score
from sklearn.ensemble import RandomForestClassifier



from sklearn.model_selection import cross_val_score
from sklearn.linear_model import SGDClassifier
from mlxtend.classifier import StackingClassifier

from sklearn import model_selection
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import precision_recall_curve, auc, roc_curve
```

In [89]:
```
print("Number of data points in train data :",X_train.shape)
print("Number of data points in test data :",X_test.shape)
```

```
Number of data points in train data : (35000, 218)
Number of data points in test data : (15000, 218)
```

In [90]:
```
print("-"*10, "Distribution of output variable in train data", "-"*10)
train_distr = Counter(y_train)
train_len = len(y_train)
print("Class 0: ",int(train_distr[0])/train_len,"Class 1: ", int(train_distr[1
])/train_len)
print("-"*10, "Distribution of output variable in train data", "-"*10)
test_distr = Counter(y_test)
test_len = len(y_test)
print("Class 0: ",int(test_distr[1])/test_len, "Class 1: ",int(test_distr[1])/
test_len)
```

```
---------- Distribution of output variable in train data ----------
Class 0:  0.6284285714285714 Class 1:  0.37157142857142855
---------- Distribution of output variable in train data ----------
Class 0:  0.3715333333333333 Class 1:  0.3715333333333333
```

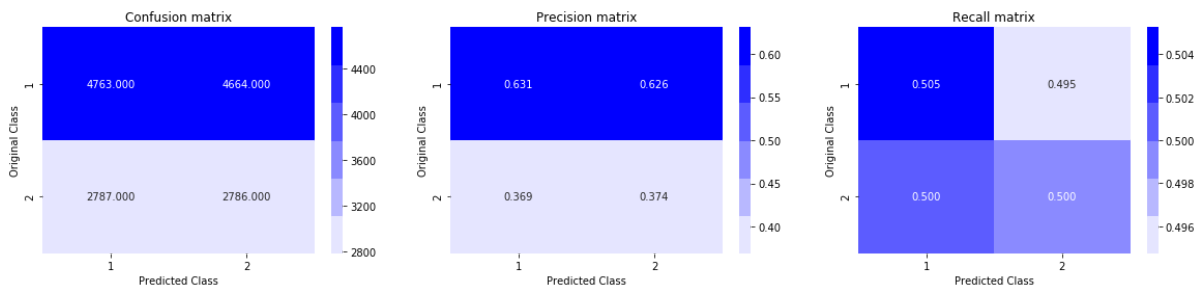
In [91]:
```
# This function plots the confusion matrices given y_i, y_i_hat.
def plot_confusion_matrix(test_y, predict_y):
    C = confusion_matrix(test_y, predict_y)
    # C = 9,9 matrix, each cell (i,j) represents number of points of class i a
re predicted class j

    A =(((C.T)/(C.sum(axis=1))).T)
    #divid each element of the confusion matrix with the sum of elements in th
at column

    # C = [[1, 2],
    #     [3, 4]]
    # C.T = [[1, 3],
    #        [2, 4]]
    # C.sum(axis = 1)  axis=0 corresonds to columns and axis=1 corresponds to
 rows in two diamensional array
    # C.sum(axix =1) = [[3, 7]]
    # ((C.T)/(C.sum(axis=1))) = [[1/3, 3/7]
    #                           [2/3, 4/7]]

    # ((C.T)/(C.sum(axis=1))).T = [[1/3, 2/3]
    #                           [3/7, 4/7]]
    # sum of row elements = 1

    B =(C/C.sum(axis=0))
    #divid each element of the confusion matrix with the sum of elements in th
at row
    # C = [[1, 2],
    #     [3, 4]]
    # C.sum(axis = 0)  axis=0 corresonds to columns and axis=1 corresponds to
 rows in two diamensional array
    # C.sum(axix =0) = [[4, 6]]
    # (C/C.sum(axis=0)) = [[1/4, 2/6],
    #                      [3/4, 4/6]]
    plt.figure(figsize=(20,4))

    labels = [1,2]
    # representing A in heatmap format
    cmap=sns.light_palette("blue")
    plt.subplot(1, 3, 1)
    sns.heatmap(C, annot=True, cmap=cmap, fmt=".3f", xticklabels=labels, ytick
labels=labels)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.title("Confusion matrix")

    plt.subplot(1, 3, 2)
    sns.heatmap(B, annot=True, cmap=cmap, fmt=".3f", xticklabels=labels, ytick
labels=labels)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.title("Precision matrix")

    plt.subplot(1, 3, 3)
    # representing B in heatmap format
    sns.heatmap(A, annot=True, cmap=cmap, fmt=".3f", xticklabels=labels, ytick
```

```
labels=labels)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.title("Recall matrix")

    plt.show()
```

4.4 Building a random model (Finding worst-case log-loss)

In [92]:
```
# we need to generate 9 numbers and the sum of numbers should be 1
# one solution is to genarate 9 numbers and divide each of the numbers by thei
r sum
# ref: https://stackoverflow.com/a/18662466/4084039
# we create a output array that has exactly same size as the CV data
predicted_y = np.zeros((test_len,2))
for i in range(test_len):
    rand_probs = np.random.rand(1,2)
    predicted_y[i] = ((rand_probs/sum(sum(rand_probs)))[0])
print("Log loss on Test Data using Random Model",log_loss(y_test, predicted_y,
eps=1e-15))

predicted_y =np.argmax(predicted_y, axis=1)
plot_confusion_matrix(y_test, predicted_y)
```

```
Log loss on Test Data using Random Model 0.8822391322845902
```



4.4 Logistic Regression with hyperparameter tuning

In [93]:
```
X_train.shape
```

Out[93]:
```
(35000, 218)
```

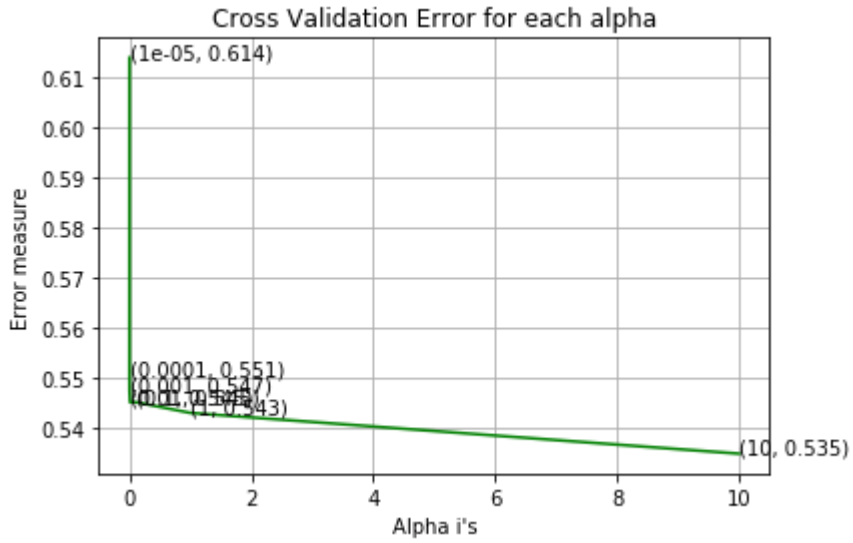
In [94]:
```
y_train.shape
```

Out[94]:
```
(35000,)
```

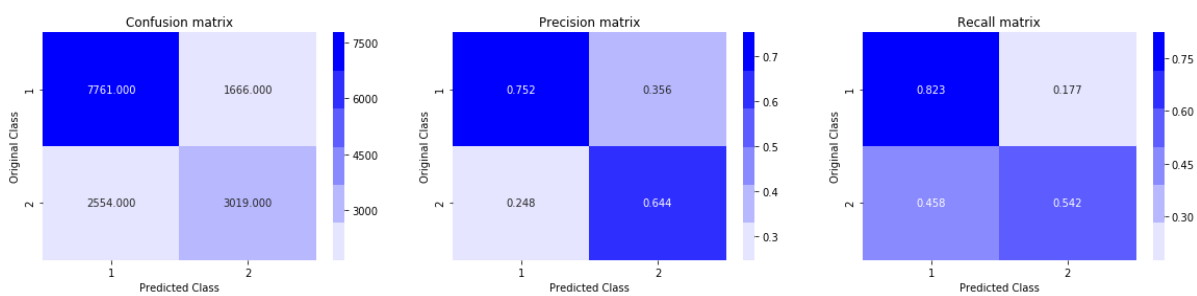
In [95]:
```
alpha = [10 ** x for x in range(-5, 2)] # hyperparam for SGD classifier.

# read more about SGDClassifier() at http://scikit-learn.org/stable/modules/ge
nerated/sklearn.linear_model.SGDClassifier.html
# ------------------------------
# default parameters
# SGDClassifier(loss='hinge', penalty='l2', alpha=0.0001, l1_ratio=0.15, fit_i
ntercept=True, max_iter=None, tol=None,
# shuffle=True, verbose=0, epsilon=0.1, n_jobs=1, random_state=None, learning_
rate='optimal', eta0=0.0, power_t=0.5,
# class_weight=None, warm_start=False, average=False, n_iter=None)

# some of methods
# fit(X, y[, coef_init, intercept_init, …])     Fit linear model with Stochast
ic Gradient Descent.
# predict(X)    Predict class labels for samples in X.

#-------------------------------
# video link:
#------------------------------


log_error_array=[]
for i in alpha:
    clf = SGDClassifier(alpha=i, penalty='l2', loss='log', random_state=42)
    clf.fit(X_train, y_train)
    sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
    sig_clf.fit(X_train, y_train)
    predict_y = sig_clf.predict_proba(X_test)
    log_error_array.append(log_loss(y_test, predict_y, labels=clf.classes_, ep
s=1e-15))
    print('For values of alpha = ', i, "The log loss is:",log_loss(y_test, pre
dict_y, labels=clf.classes_, eps=1e-15))

fig, ax = plt.subplots()
ax.plot(alpha, log_error_array,c='g')
for i, txt in enumerate(np.round(log_error_array,3)):
    ax.annotate((alpha[i],np.round(txt,3)), (alpha[i],log_error_array[i]))
plt.grid()
plt.title("Cross Validation Error for each alpha")
plt.xlabel("Alpha i's")
plt.ylabel("Error measure")
plt.show()


best_alpha = np.argmin(log_error_array)
clf = SGDClassifier(alpha=alpha[best_alpha], penalty='l2', loss='log', random_
state=42)
clf.fit(X_train, y_train)
sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
sig_clf.fit(X_train, y_train)

predict_y = sig_clf.predict_proba(X_train)
print('For values of best alpha = ', alpha[best_alpha], "The train log loss i
s:",log_loss(y_train, predict_y, labels=clf.classes_, eps=1e-15))
predict_y = sig_clf.predict_proba(X_test)
```

```
print('For values of best alpha = ', alpha[best_alpha], "The test log loss i
s:",log_loss(y_test, predict_y, labels=clf.classes_, eps=1e-15))
predicted_y =np.argmax(predict_y,axis=1)
print("Total number of data points :", len(predicted_y))
plot_confusion_matrix(y_test, predicted_y)
```

```
For values of alpha =  1e-05 The log loss is: 0.6139108254675448
For values of alpha =  0.0001 The log loss is: 0.5506413815858594
For values of alpha =  0.001 The log loss is: 0.5474480863407483
For values of alpha =  0.01 The log loss is: 0.5450587130323646
For values of alpha =  0.1 The log loss is: 0.545096165621808
For values of alpha =  1 The log loss is: 0.5429380505011125
For values of alpha =  10 The log loss is: 0.5348015870524657
```




```
For values of best alpha =  10 The train log loss is: 0.5160872276182437
For values of best alpha =  10 The test log loss is: 0.5348015870524657
Total number of data points : 15000
```

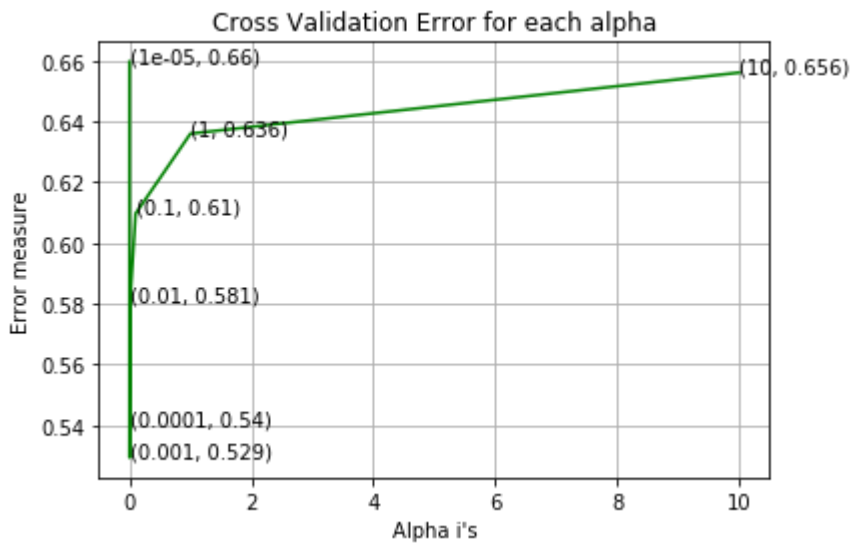


4.5 Linear SVM with hyperparameter tuning

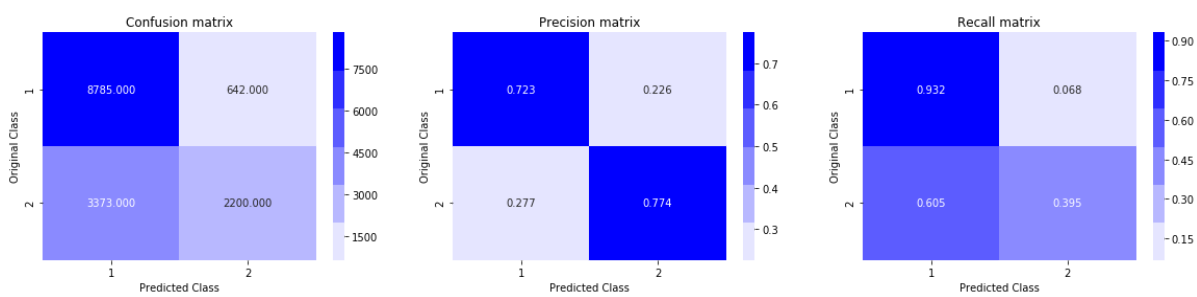
In [96]:
```
alpha = [10 ** x for x in range(-5, 2)] # hyperparam for SGD classifier.

# read more about SGDClassifier() at http://scikit-learn.org/stable/modules/ge
nerated/sklearn.linear_model.SGDClassifier.html
# ------------------------------
# default parameters
# SGDClassifier(loss='hinge', penalty='l2', alpha=0.0001, l1_ratio=0.15, fit_i
ntercept=True, max_iter=None, tol=None,
# shuffle=True, verbose=0, epsilon=0.1, n_jobs=1, random_state=None, learning_
rate='optimal', eta0=0.0, power_t=0.5,
# class_weight=None, warm_start=False, average=False, n_iter=None)

# some of methods
# fit(X, y[, coef_init, intercept_init, …])     Fit linear model with Stochast
ic Gradient Descent.
# predict(X)    Predict class labels for samples in X.

#-------------------------------
# video link:
#------------------------------


log_error_array=[]
for i in alpha:
    clf = SGDClassifier(alpha=i, penalty='l1', loss='hinge', random_state=42)
    clf.fit(X_train, y_train)
    sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
    sig_clf.fit(X_train, y_train)
    predict_y = sig_clf.predict_proba(X_test)
    log_error_array.append(log_loss(y_test, predict_y, labels=clf.classes_, ep
s=1e-15))
    print('For values of alpha = ', i, "The log loss is:",log_loss(y_test, pre
dict_y, labels=clf.classes_, eps=1e-15))

fig, ax = plt.subplots()
ax.plot(alpha, log_error_array,c='g')
for i, txt in enumerate(np.round(log_error_array,3)):
    ax.annotate((alpha[i],np.round(txt,3)), (alpha[i],log_error_array[i]))
plt.grid()
plt.title("Cross Validation Error for each alpha")
plt.xlabel("Alpha i's")
plt.ylabel("Error measure")
plt.show()


best_alpha = np.argmin(log_error_array)
clf = SGDClassifier(alpha=alpha[best_alpha], penalty='l1', loss='hinge', rando
m_state=42)
clf.fit(X_train, y_train)
sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
sig_clf.fit(X_train, y_train)

predict_y = sig_clf.predict_proba(X_train)
print('For values of best alpha = ', alpha[best_alpha], "The train log loss i
s:",log_loss(y_train, predict_y, labels=clf.classes_, eps=1e-15))
predict_y = sig_clf.predict_proba(X_test)
```

```
print('For values of best alpha = ', alpha[best_alpha], "The test log loss i
s:",log_loss(y_test, predict_y, labels=clf.classes_, eps=1e-15))
predicted_y =np.argmax(predict_y,axis=1)
print("Total number of data points :", len(predicted_y))
plot_confusion_matrix(y_test, predicted_y)
```

```
For values of alpha =  1e-05 The log loss is: 0.6597667139263508
For values of alpha =  0.0001 The log loss is: 0.539880581781623
For values of alpha =  0.001 The log loss is: 0.5292727905290536
For values of alpha =  0.01 The log loss is: 0.5807879548496996
For values of alpha =  0.1 The log loss is: 0.6096658493336757
For values of alpha =  1 The log loss is: 0.6360273963693932
For values of alpha =  10 The log loss is: 0.6561313121658249
```




```
For values of best alpha =  0.001 The train log loss is: 0.5245241440419677
For values of best alpha =  0.001 The test log loss is: 0.5292727905290536
Total number of data points : 15000
```



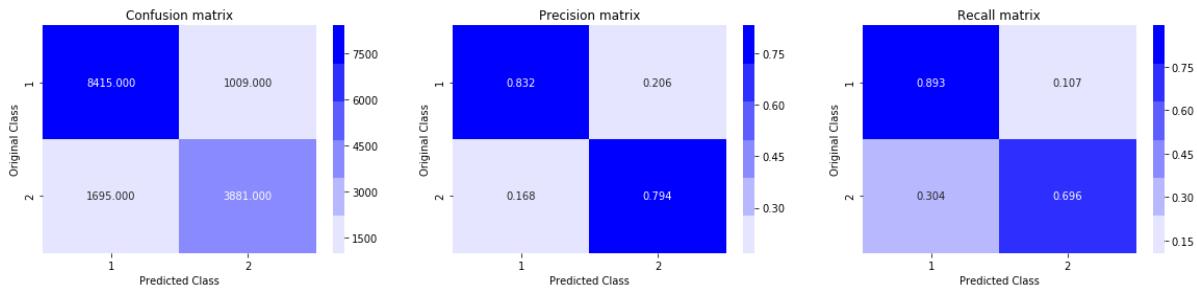
4.6 XGBoost

In [19]:
```
import xgboost as xgb
params = {}
params['objective'] = 'binary:logistic'
params['eval_metric'] = 'logloss'
params['eta'] = 0.02
params['max_depth'] = 4

d_train = xgb.DMatrix(X_train, label=y_train)
d_test = xgb.DMatrix(X_test, label=y_test)

watchlist = [(d_train, 'train'), (d_test, 'valid')]

bst = xgb.train(params, d_train, 400, watchlist, early_stopping_rounds=20, ver
bose_eval=10)

xgdmat = xgb.DMatrix(X_train,y_train)
predict_y = bst.predict(d_test)
print("The test log loss is:",log_loss(y_test, predict_y, labels=clf.classes_,
eps=1e-15))
```

```
[0]	train-logloss:0.684851	valid-logloss:0.684953
Multiple eval metrics have been passed: 'valid-logloss' will be used for earl
y stopping.

Will train until valid-logloss hasn't improved in 20 rounds.
[10]	train-logloss:0.615672	valid-logloss:0.617235
[20]	train-logloss:0.564235	valid-logloss:0.566696
[30]	train-logloss:0.52571	valid-logloss:0.52901
[40]	train-logloss:0.496076	valid-logloss:0.49987
[50]	train-logloss:0.472854	valid-logloss:0.477215
[60]	train-logloss:0.454286	valid-logloss:0.459067
[70]	train-logloss:0.439118	valid-logloss:0.444304
[80]	train-logloss:0.42691	valid-logloss:0.432455
[90]	train-logloss:0.4169	valid-logloss:0.422847
[100]	train-logloss:0.408725	valid-logloss:0.414995
[110]	train-logloss:0.401912	valid-logloss:0.408567
[120]	train-logloss:0.396066	valid-logloss:0.403061
[130]	train-logloss:0.391374	valid-logloss:0.398884
[140]	train-logloss:0.38691	valid-logloss:0.394923
[150]	train-logloss:0.383345	valid-logloss:0.391835
[160]	train-logloss:0.380073	valid-logloss:0.389033
[170]	train-logloss:0.377205	valid-logloss:0.386466
[180]	train-logloss:0.374542	valid-logloss:0.384183
[190]	train-logloss:0.372159	valid-logloss:0.382154
[200]	train-logloss:0.370135	valid-logloss:0.380523
[210]	train-logloss:0.368309	valid-logloss:0.379112
[220]	train-logloss:0.366559	valid-logloss:0.377752
[230]	train-logloss:0.364878	valid-logloss:0.376428
[240]	train-logloss:0.362923	valid-logloss:0.375018
[250]	train-logloss:0.36116	valid-logloss:0.373739
[260]	train-logloss:0.359671	valid-logloss:0.372714
[270]	train-logloss:0.358002	valid-logloss:0.371578
[280]	train-logloss:0.35634	valid-logloss:0.370489
[290]	train-logloss:0.354737	valid-logloss:0.369435
[300]	train-logloss:0.353165	valid-logloss:0.368448
[310]	train-logloss:0.351784	valid-logloss:0.36757
[320]	train-logloss:0.350317	valid-logloss:0.366676
[330]	train-logloss:0.348978	valid-logloss:0.365911
[340]	train-logloss:0.347671	valid-logloss:0.365167
[350]	train-logloss:0.346535	valid-logloss:0.364578
[360]	train-logloss:0.345207	valid-logloss:0.363859
[370]	train-logloss:0.343978	valid-logloss:0.363139
[380]	train-logloss:0.342937	valid-logloss:0.362694
[390]	train-logloss:0.341738	valid-logloss:0.362075
[399]	train-logloss:0.340574	valid-logloss:0.361421
The test log loss is: 0.3614204904499153
```

In [20]:
```
predicted_y =np.array(predict_y>0.5,dtype=int)
print("Total number of data points :", len(predicted_y))
plot_confusion_matrix(y_test, predicted_y)
```

```
Total number of data points : 15000
```



5. Assignments

1. Try out models (Logistic regression, Linear-SVM) with simple TF-IDF vectors instead of TD_IDF weighted word2Vec.
2. Hyperparameter tune XgBoost using RandomSearch to reduce the log-loss.

5.1 : Try out models (Logistic regression, Linear-SVM) with simple TF-IDF vectors instead of TD_IDF weighted word2Vec.

In [11]:
```
df4  = df1.merge(df3, on='id',how='left')
```

In [12]:
```
df4 = df4.sample(n=50000)
```

In [13]:
```
df4.shape
```

Out[13]:
```
(50000, 35)
```

In [14]:
```
y_true = df4['is_duplicate_x']
```

In [15]:
```
y_true
```

Out[15]:
```
107040    0
165041    0
64671     0
371007    0
322766    0
          ..
306493    0
206537    1
225852    0
73709     1
269458    0
Name: is_duplicate_x, Length: 50000, dtype: int64
```

In [16]:
```
df4 = df4.drop(['is_duplicate_x'],axis=1)
```

In [17]:
```
df4.shape
```

Out[17]:
```
(50000, 34)
```

In [18]:
```
X_train,X_test, y_train, y_test = train_test_split(df4, y_true, stratify=y_tru
e, test_size=0.3)
```

In [19]:
```
X_train.shape
```

Out[19]:
```
(35000, 34)
```

In [20]:
```
X_test.shape
```

Out[20]:
```
(15000, 34)
```

In [21]:
```
X_train.shape[0] + X_test.shape[0]
```

Out[21]:
```
50000
```

In [22]:
```
# Filling the null values with ' '
X_train = X_train.fillna(' ')
nan_rows1 = X_train[X_train.isnull().any(1)]
print (nan_rows1)

# Filling the null values with ' '
X_test = X_test.fillna(' ')
nan_rows2 = X_test[X_test.isnull().any(1)]
print (nan_rows2)
```

```
Empty DataFrame
Columns: [id, question1_x, question2_x, cwc_min, cwc_max, csc_min, csc_max, c
tc_min, ctc_max, last_word_eq, first_word_eq, abs_len_diff, mean_len, token_s
et_ratio, token_sort_ratio, fuzz_ratio, fuzz_partial_ratio, longest_substr_ra
tio, qid1_y, qid2_y, question1_y, question2_y, is_duplicate_y, freq_qid1, fre
q_qid2, q1len, q2len, q1_n_words, q2_n_words, word_Common, word_Total, word_s
hare, freq_q1+q2, freq_q1-q2]
Index: []

[0 rows x 34 columns]
Empty DataFrame
Columns: [id, question1_x, question2_x, cwc_min, cwc_max, csc_min, csc_max, c
tc_min, ctc_max, last_word_eq, first_word_eq, abs_len_diff, mean_len, token_s
et_ratio, token_sort_ratio, fuzz_ratio, fuzz_partial_ratio, longest_substr_ra
tio, qid1_y, qid2_y, question1_y, question2_y, is_duplicate_y, freq_qid1, fre
q_qid2, q1len, q2len, q1_n_words, q2_n_words, word_Common, word_Total, word_s
hare, freq_q1+q2, freq_q1-q2]
Index: []

[0 rows x 34 columns]
```

In [23]:
```
X_train
```

Out[23]:

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
60596	60596	how do i insert a sigma symbol into a word doc...	how can i insert a line break in ms word	0.399992	0.399992	0.599988	0.599988	0.499995
383067	383067	how do i get a job as a film and tv critic	how does someone become a professional tv or f...	0.599988	0.499992	0.499988	0.333328	0.499995
293924	293924	how do i apply for a pan card online	how can i apply pan card with initial in surna...	0.749981	0.374995	0.599988	0.374995	0.666659
29120	29120	what is the meaning of a clean sheet in soccer	in soccer what does it mean when a club is in...	0.249994	0.249994	0.666656	0.571420	0.499995
67034	67034	why should i learn algebra	when will the em drive be disproved	0.000000	0.000000	0.000000	0.000000	0.000000
7714	7714	how do i advertise my business on quora	how do i advertise on quora	0.999950	0.666644	0.999975	0.799984	0.999983
131675	131675	how can you get fat from not eating	can you get fat from eating salad	0.999967	0.749981	0.999967	0.599988	0.857131
105411	105411	would you marry a non virgin woman	why would you marry a non virgin	0.999975	0.799984	0.999950	0.666644	0.857131
112404	112404	how to activate whatsapp on my new iphone with...	can we use same mobile number on two different...	0.444440	0.307690	0.399992	0.153845	0.428568
394409	394409	what is it like having an attractive spouse	why is it that other people own spouses appear...	0.333322	0.166664	0.399992	0.249997	0.374995
168963	168963	is work experience necessary for pursuing mims...	i have below average gpa and gre scores and ro...	0.444440	0.285712	0.499995	0.416663	0.391303
334360	334360	is it possible to build vehicles that transfor...	is it really possible to make robots like we h...	0.499994	0.499994	0.714276	0.624992	0.599996
101722	101722	why did the us not see the attack on pearl har...	in what ways was the us unprepared for an atta...	0.666656	0.666656	0.399992	0.285710	0.499996

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
102929	102929	what competencies are required for software pr...	how do i grow professionally from being a soft...	0.249994	0.249994	0.000000	0.000000	0.142855
257488	257488	what is the working principle of steam engine	how do steam engines work	0.333322	0.249994	0.000000	0.000000	0.199996
85248	85248	why does youtube video loads faster than other...	what should i do if videos are not loading on ...	0.333322	0.199996	0.000000	0.000000	0.111110
311109	311109	what is the difference between mathematics and...	does listening to music while studying help	0.000000	0.000000	0.000000	0.000000	0.000000
212024	212024	how do you make a webcam work	how do you make your webcam work	0.999967	0.999967	0.749981	0.749981	0.857131
339948	339948	is the new 2k rupee note really implementing n...	is the new 2k rupees note gps enabled	0.666656	0.399996	0.999950	0.999950	0.749991
273800	273800	what are the safety precautions on handling sh...	what are the safety precautions on handling sh...	0.857131	0.749991	0.999983	0.999983	0.857137
301615	301615	what should i do to apply for ipu cet mbbs cou...	how can i apply for ipu cet 2016	0.999975	0.666656	0.499988	0.333328	0.749991
194226	194226	what is the yugioh card of last will	why is the yugioh card of last will illegal	0.999967	0.749981	0.799984	0.799984	0.874989
155877	155877	what are some tips on making it through the jo...	what are some tips on making it through the jo...	0.714276	0.714276	0.999988	0.999988	0.866661
19024	19024	which are the best movies to watch	what is best movies to watch	0.999967	0.999967	0.333322	0.249994	0.666656
16131	16131	how does one get uk citizenship	how do i get uk citizenship	0.999967	0.749981	0.499975	0.333322	0.666656
246339	246339	what do you think about the removal of usb por...	do you think it was a mistake for apple to get...	0.499994	0.444440	0.571420	0.399996	0.499997

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
84434	84434	where did pizza originate	where was pizza invented	0.499975	0.499975	0.499975	0.499975	0.499988
220526	220526	how can you see what a passenger rates an uber...	can an ola or uber driver see the rating a pas...	0.799984	0.571420	0.499992	0.428565	0.636358
312814	312814	is hillary clinton ill	obama presidency first term 2009 13 is us ...	0.999967	0.230767	0.999900	0.166664	0.999975
222175	222175	what are some good tips for essay writing	what are the best tips for essay writing	0.749981	0.749981	0.749981	0.749981	0.749991
...	...	...	...	...	...	...	...	...
58421	58421	what are the arguments for and against interra...	what are some arguments for interracial marriage	0.999967	0.999967	0.749981	0.499992	0.857131
7345	7345	vision eyesight what are some tricks to spe...	why does death cause a dilated pupil	0.249994	0.099999	0.000000	0.000000	0.142855
252859	252859	what were the effects of the spanish inquisition	was the spanish inquisition a cause for the ri...	0.666644	0.333328	0.249994	0.199996	0.374995
129692	129692	what is the best way to download a video from ...	how can i download video from youtube	0.999967	0.428565	0.249994	0.142855	0.571420
294731	294731	can we use jio 4g sim to 3g handsets	how do i use the jio 4g sim in 3g cellphone	0.833319	0.833319	0.000000	0.000000	0.555549
337632	337632	when were public libraries first created	why were public libraries first created	0.999975	0.999975	0.499975	0.499975	0.833319
257417	257417	how do rich men treat women	why is it difficult to become a billionaire	0.000000	0.000000	0.000000	0.000000	0.000000
41373	41373	what is empathy	what causes empathy	0.999900	0.499975	0.999900	0.499975	0.666644
168443	168443	what is the most painless and quick way to die	what are some fairly painless ways to die	0.499988	0.499988	0.499988	0.333328	0.499994

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
84813	84813	how do you mass save photos from text messages...	is there an easy way save my text messages on ...	0.666656	0.666656	0.399992	0.333328	0.499996
396340	396340	why do people feel the need to make others fee...	why do i feel bad for other underprivileged pe...	0.499992	0.428565	0.499992	0.374995	0.461535
380893	380893	is gatsby own love for daisy genuine	the great gatsby 1925 book did daisy know t...	0.499988	0.222220	0.000000	0.000000	0.285710
145057	145057	i am so sad what can i do	i am sad what do i do	0.999900	0.999900	0.999975	0.666656	0.714276
57568	57568	i ended up feelings on my friend but she told ...	what are some programs similar to deepsound	0.000000	0.000000	0.499988	0.117646	0.285710
243705	243705	is there a way to hack someone own instagram	how do you get in someone own instagram	0.666644	0.499988	0.199996	0.199996	0.374995
298900	298900	what are some lesser known facts about pm nare...	what are some lesser known facts incidents rel...	0.833319	0.714276	0.749981	0.749981	0.799992
227181	227181	what are the chances of a massive earthquake h...	do cbse students suffer in class 12 they get...	0.000000	0.000000	0.249994	0.166664	0.066666
232466	232466	why is there a proc directory in linux	why are there so many binary directories in li...	0.333322	0.249994	0.599988	0.599988	0.499994
30698	30698	with a macbook pro mid 2010 is it time to upg...	with a macbook pro mid 2010 is it time to upg...	0.999983	0.999983	0.999980	0.999980	0.999991
167930	167930	which is the best book for core java	what is the best book for learning java	0.749981	0.749981	0.749981	0.749981	0.749991
387637	387637	how do you write 15 december 2016 without usin...	how do you write math 23 textrm rd math ...	0.571420	0.333331	0.999967	0.599988	0.699993
246181	246181	how can i add participants to a whatsapp group...	how do i add a photograph as icon for a broadc...	0.499992	0.374995	0.428565	0.428565	0.428568

	id	question1_x	question2_x	cwc_min	cwc_max	csc_min	csc_max	ctc_min
184512	184512	what is the importance of sketching in enginee...	why is engineering important to my community	0.333322	0.333322	0.249994	0.199996	0.285710
366093	366093	what represents typical account fees	what is stamp fees	0.499975	0.249994	0.999900	0.499975	0.499988
133475	133475	how competent skilled was major dhyan chand	how good was major dhyan chand in hockey	0.599988	0.599988	0.999950	0.666644	0.714276
148454	148454	how does paytm earn money after giving cashbacks	how does paytm earn by giving extra cash back ...	0.599988	0.333330	0.666644	0.499988	0.624992
399144	399144	what do not regret in your life	what do you regret in your life that you did n...	0.999950	0.999950	0.999980	0.624992	0.999986
129576	129576	what is the cut off in coep for mechanical girls	what is the cut off for coep	0.999950	0.499988	0.999980	0.833319	0.999986
203966	203966	is it bad that tilapia from walmart at least ...	where can you find a chart of carbon monoxide ...	0.399992	0.285710	0.000000	0.000000	0.199998
378334	378334	under what presidents did we have a great econ...	under what presidents did we have a great econ...	0.999992	0.999992	0.999989	0.999989	0.916663

35000 rows × 34 columns

Preparing train data matrix

Tfidf vectorization on train data

In [24]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
questions_train = list(X_train['question1_x']) + list(X_train['question2_x'])
vectorizer_tfidf_ques = TfidfVectorizer(lowercase=False,min_df=10)
vectorizer_tfidf_ques.fit(questions_train)

q1_tfidf_train = vectorizer_tfidf_ques.transform(X_train['question1_x'])
q2_tfidf_train = vectorizer_tfidf_ques.transform(X_train['question2_x'])
print("Shape of matrix after q1_tfidf_train ",q1_tfidf_train.shape)
print("Shape of matrix after q2_tfidf_train ",q2_tfidf_train.shape)
```

```
Shape of matrix after q1_tfidf_train  (35000, 5197)
Shape of matrix after q2_tfidf_train  (35000, 5197)
```

In [25]:
```
X_train = X_train.drop(['id','question1_x','question2_x','qid1_y','qid2_y','qu
estion1_y','question2_y','is_duplicate_y'],axis=1)
```

In [26]:
```
X_train = hstack((X_train,q1_tfidf_train,q2_tfidf_train))
```

In [27]:
```
X_train
```

Out[27]:
```
<35000x10420 sparse matrix of type '<class 'numpy.float64'>'
        with 1474407 stored elements in COOrdinate format>
```

In [ ]:
```
```

Preparing test data matrix

Tfidf vectorization on test data

In [28]:
```
from sklearn.feature_extraction.text import TfidfVectorizer

q1_tfidf_test = vectorizer_tfidf_ques.transform(X_test['question1_x'])
q2_tfidf_test = vectorizer_tfidf_ques.transform(X_test['question2_x'])
print("Shape of matrix after q1_tfidf_test ",q1_tfidf_test.shape)
print("Shape of matrix after q2_tfidf_test ",q2_tfidf_test.shape)
```

```
Shape of matrix after q1_tfidf_test  (15000, 5197)
Shape of matrix after q2_tfidf_test  (15000, 5197)
```

In [29]:
```
X_test = X_test.drop(['id','question1_x','question2_x','qid1_y','qid2_y','ques
tion1_y','question2_y','is_duplicate_y'],axis=1)
```

In [30]:
```
X_test = hstack((X_test,q1_tfidf_test,q2_tfidf_test))
```

In [31]:
```
X_test
```

Out[31]:
```
<15000x10420 sparse matrix of type '<class 'numpy.float64'>'
        with 630327 stored elements in COOrdinate format>
```

Logistic Regression with hyperparameter tuning

In [64]:
```
X_train.shape
```

Out[64]:
```
(35000, 10476)
```

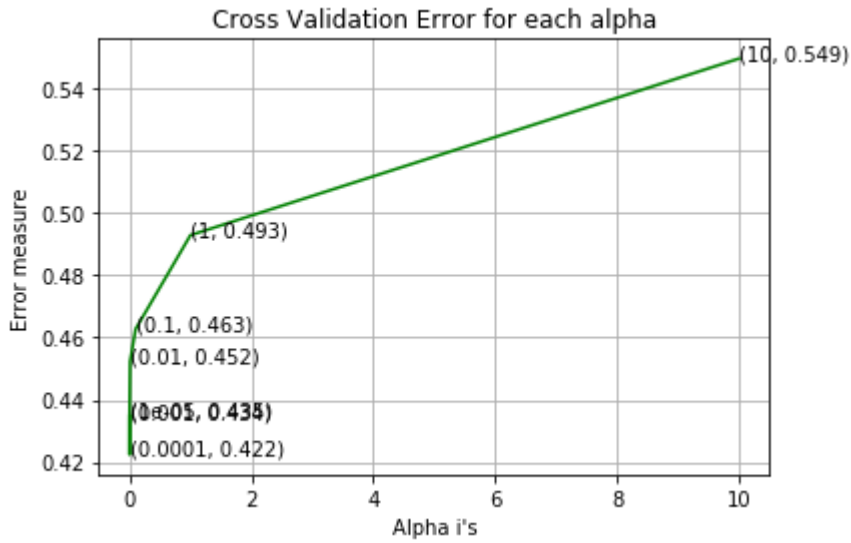
In [65]:
```
y_train.shape
```

Out[65]:
```
(35000,)
```

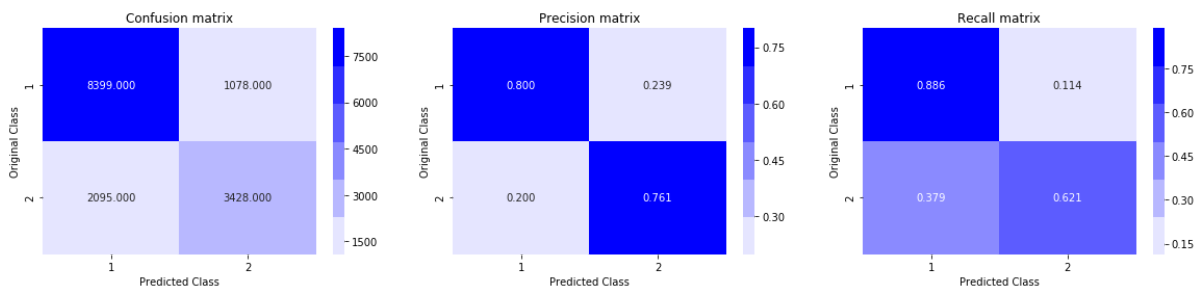
In [48]:
```
alpha = [10 ** x for x in range(-5, 2)] # hyperparam for SGD classifier.

# read more about SGDClassifier() at http://scikit-learn.org/stable/modules/ge
nerated/sklearn.linear_model.SGDClassifier.html
# ------------------------------
# default parameters
# SGDClassifier(loss='hinge', penalty='l2', alpha=0.0001, l1_ratio=0.15, fit_i
ntercept=True, max_iter=None, tol=None,
# shuffle=True, verbose=0, epsilon=0.1, n_jobs=1, random_state=None, learning_
rate='optimal', eta0=0.0, power_t=0.5,
# class_weight=None, warm_start=False, average=False, n_iter=None)

# some of methods
# fit(X, y[, coef_init, intercept_init, …])     Fit linear model with Stochast
ic Gradient Descent.
# predict(X)    Predict class labels for samples in X.

#-------------------------------
# video link:
#------------------------------


log_error_array=[]
for i in alpha:
    clf = SGDClassifier(alpha=i, penalty='l2', loss='log', random_state=42)
    clf.fit(X_train, y_train)
    sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
    sig_clf.fit(X_train, y_train)
    predict_y = sig_clf.predict_proba(X_test)
    log_error_array.append(log_loss(y_test, predict_y, labels=clf.classes_, ep
s=1e-15))
    print('For values of alpha = ', i, "The log loss is:",log_loss(y_test, pre
dict_y, labels=clf.classes_, eps=1e-15))

fig, ax = plt.subplots()
ax.plot(alpha, log_error_array,c='g')
for i, txt in enumerate(np.round(log_error_array,3)):
    ax.annotate((alpha[i],np.round(txt,3)), (alpha[i],log_error_array[i]))
plt.grid()
plt.title("Cross Validation Error for each alpha")
plt.xlabel("Alpha i's")
plt.ylabel("Error measure")
plt.show()


best_alpha = np.argmin(log_error_array)
clf = SGDClassifier(alpha=alpha[best_alpha], penalty='l2', loss='log', random_
state=42)
clf.fit(X_train, y_train)
sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
sig_clf.fit(X_train, y_train)

predict_y = sig_clf.predict_proba(X_train)
print('For values of best alpha = ', alpha[best_alpha], "The train log loss i
s:",log_loss(y_train, predict_y, labels=clf.classes_, eps=1e-15))
predict_y = sig_clf.predict_proba(X_test)
```

```
print('For values of best alpha = ', alpha[best_alpha], "The test log loss i
s:",log_loss(y_test, predict_y, labels=clf.classes_, eps=1e-15))
predicted_y =np.argmax(predict_y,axis=1)
print("Total number of data points :", len(predicted_y))
plot_confusion_matrix(y_test, predicted_y)
```

```
For values of alpha =  1e-05 The log loss is: 0.4348345550728513
For values of alpha =  0.0001 The log loss is: 0.42239048766503967
For values of alpha =  0.001 The log loss is: 0.43395388114416994
For values of alpha =  0.01 The log loss is: 0.4522644937425442
For values of alpha =  0.1 The log loss is: 0.462661133296654
For values of alpha =  1 The log loss is: 0.49279458859004877
For values of alpha =  10 The log loss is: 0.5494454892556244
```




```
For values of best alpha =  0.0001 The train log loss is: 0.41862086549062827
For values of best alpha =  0.0001 The test log loss is: 0.42239048766503967
Total number of data points : 15000
```

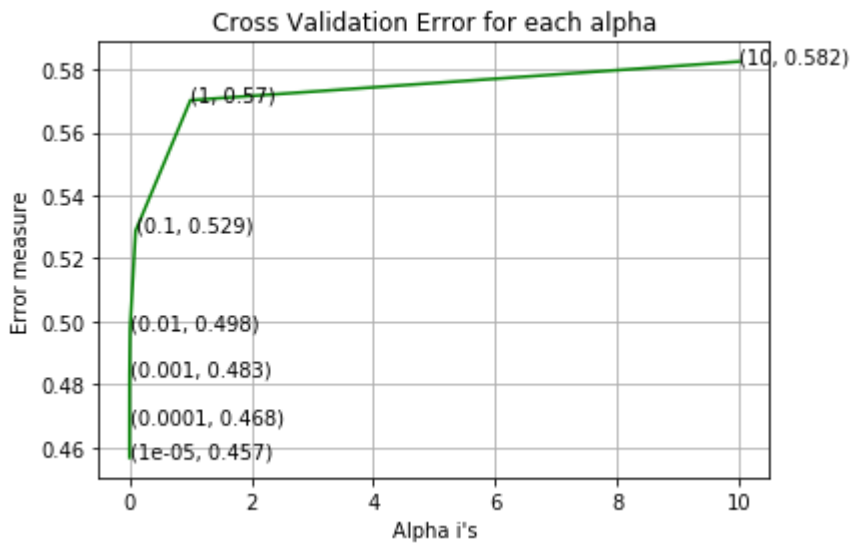


Linear SVM with hyperparameter tuning

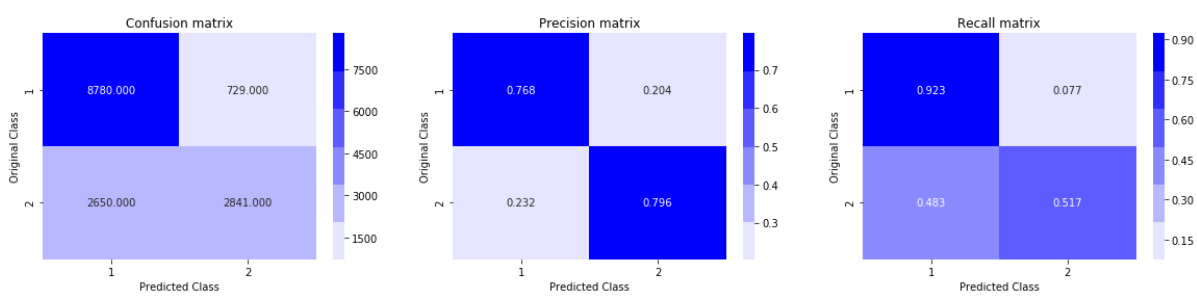
In [67]:
```
alpha = [10 ** x for x in range(-5, 2)] # hyperparam for SGD classifier.

# read more about SGDClassifier() at http://scikit-learn.org/stable/modules/ge
nerated/sklearn.linear_model.SGDClassifier.html
# ------------------------------
# default parameters
# SGDClassifier(loss='hinge', penalty='l2', alpha=0.0001, l1_ratio=0.15, fit_i
ntercept=True, max_iter=None, tol=None,
# shuffle=True, verbose=0, epsilon=0.1, n_jobs=1, random_state=None, learning_
rate='optimal', eta0=0.0, power_t=0.5,
# class_weight=None, warm_start=False, average=False, n_iter=None)

# some of methods
# fit(X, y[, coef_init, intercept_init, …])     Fit linear model with Stochast
ic Gradient Descent.
# predict(X)    Predict class labels for samples in X.

#-------------------------------
# video link:
#------------------------------


log_error_array=[]
for i in alpha:
    clf = SGDClassifier(alpha=i, penalty='l1', loss='hinge', random_state=42)
    clf.fit(X_train, y_train)
    sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
    sig_clf.fit(X_train, y_train)
    predict_y = sig_clf.predict_proba(X_test)
    log_error_array.append(log_loss(y_test, predict_y, labels=clf.classes_, ep
s=1e-15))
    print('For values of alpha = ', i, "The log loss is:",log_loss(y_test, pre
dict_y, labels=clf.classes_, eps=1e-15))

fig, ax = plt.subplots()
ax.plot(alpha, log_error_array,c='g')
for i, txt in enumerate(np.round(log_error_array,3)):
    ax.annotate((alpha[i],np.round(txt,3)), (alpha[i],log_error_array[i]))
plt.grid()
plt.title("Cross Validation Error for each alpha")
plt.xlabel("Alpha i's")
plt.ylabel("Error measure")
plt.show()


best_alpha = np.argmin(log_error_array)
clf = SGDClassifier(alpha=alpha[best_alpha], penalty='l1', loss='hinge', rando
m_state=42)
clf.fit(X_train, y_train)
sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
sig_clf.fit(X_train, y_train)

predict_y = sig_clf.predict_proba(X_train)
print('For values of best alpha = ', alpha[best_alpha], "The train log loss i
s:",log_loss(y_train, predict_y, labels=clf.classes_, eps=1e-15))
predict_y = sig_clf.predict_proba(X_test)
```

```
print('For values of best alpha = ', alpha[best_alpha], "The test log loss i
s:",log_loss(y_test, predict_y, labels=clf.classes_, eps=1e-15))
predicted_y =np.argmax(predict_y,axis=1)
print("Total number of data points :", len(predicted_y))
plot_confusion_matrix(y_test, predicted_y)
```

```
For values of alpha =  1e-05 The log loss is: 0.45672268969407215
For values of alpha =  0.0001 The log loss is: 0.4676851980307176
For values of alpha =  0.001 The log loss is: 0.4828686012238199
For values of alpha =  0.01 The log loss is: 0.49771951744742765
For values of alpha =  0.1 The log loss is: 0.5287577129629455
For values of alpha =  1 The log loss is: 0.5702303405814794
For values of alpha =  10 The log loss is: 0.5824917994428067
```




```
For values of best alpha =  1e-05 The train log loss is: 0.46144952981653664
For values of best alpha =  1e-05 The test log loss is: 0.45672268969407215
Total number of data points : 15000
```



In [ ]:

4.6 XGBoost

In [49]:
```
from sklearn.model_selection import RandomizedSearchCV
import xgboost as xgb
params = {
    'max_depth': [3, 4, 5, 6, 7, 8],
    'eta' : [0.01, 0.02, 0.05, 0.1]
}

xgb = xgb.XGBClassifier()
random_search = RandomizedSearchCV(xgb, param_distributions=params, scoring='n
eg_log_loss', n_jobs=-1, verbose=10, random_state=42 ,return_train_score=True)
random_search.fit(X_train, y_train)
```

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 18 concurrent workers.
[Parallel(n_jobs=-1)]: Done   3 out of  30 | elapsed:   17.1s remaining:  2.6
min
[Parallel(n_jobs=-1)]: Done   7 out of  30 | elapsed:   25.6s remaining:  1.4
min
[Parallel(n_jobs=-1)]: Done  11 out of  30 | elapsed:   31.7s remaining:   5
4.8s
[Parallel(n_jobs=-1)]: Done  15 out of  30 | elapsed:   36.0s remaining:   3
6.0s
[Parallel(n_jobs=-1)]: Done  19 out of  30 | elapsed:   37.2s remaining:   2
1.5s
[Parallel(n_jobs=-1)]: Done  23 out of  30 | elapsed:   39.6s remaining:   1
2.1s
[Parallel(n_jobs=-1)]: Done  27 out of  30 | elapsed:   49.1s remaining:
5.5s
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed:   59.9s finished
```

Out[49]:
```
RandomizedSearchCV(cv='warn', error_score='raise-deprecating',
                   estimator=XGBClassifier(base_score=0.5, booster='gbtree',
                                           colsample_bylevel=1,
                                           colsample_bynode=1,
                                           colsample_bytree=1, gamma=0,
                                           learning_rate=0.1, max_delta_step=
0,
                                           max_depth=3, min_child_weight=1,
                                           missing=None, n_estimators=100,
                                           n_jobs=1, nthread=None,
                                           objective='binary:logistic',
                                           random_state=0, reg_alpha=0,
                                           reg_lambda=1, scale_pos_weight=1,
                                           seed=None, silent=None, subsample=
1,
                                           verbosity=1),
                   iid='warn', n_iter=10, n_jobs=-1,
                   param_distributions={'eta': [0.01, 0.02, 0.05, 0.1],
                                        'max_depth': [3, 4, 5, 6, 7, 8]},
                   pre_dispatch='2*n_jobs', random_state=42, refit=True,
                   return_train_score=True, scoring='neg_log_loss', verbose=1
0)
```

In [50]:
```
print('Best hyperparameters:')
print(random_search.best_params_)
```

```
Best hyperparameters:
{'eta': 0.02, 'max_depth': 8}
```

In [56]:
```
import xgboost as xgb
params = {}
params['objective'] = 'binary:logistic'
params['eval_metric'] = 'logloss'
params['eta'] = 0.02
params['max_depth'] = 8

d_train = xgb.DMatrix(X_train, label=y_train)
d_test = xgb.DMatrix(X_test, label=y_test)

watchlist = [(d_train, 'train'), (d_test, 'valid')]

bst = xgb.train(params, d_train, 400, watchlist, early_stopping_rounds=20, ver
bose_eval=10)

xgdmat = xgb.DMatrix(X_train,y_train)
predict_y = bst.predict(d_test)
print("The test log loss is:",log_loss(y_test, predict_y, labels=clf.classes_,
eps=1e-15))
```

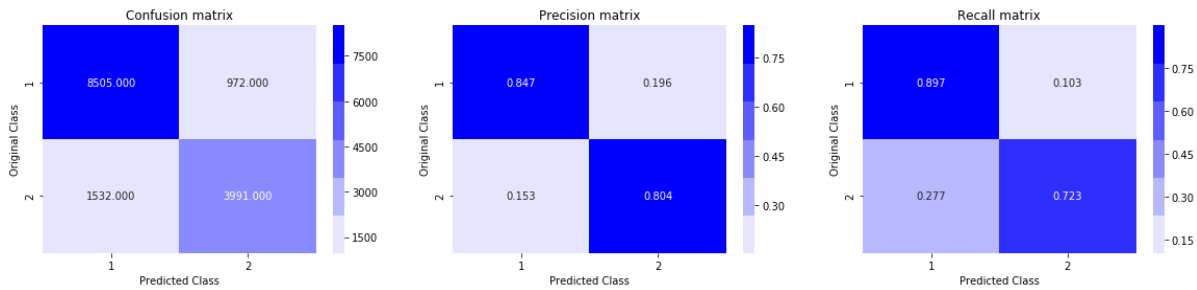
```
[0]	train-logloss:0.683136	valid-logloss:0.683294
Multiple eval metrics have been passed: 'valid-logloss' will be used for earl
y stopping.

Will train until valid-logloss hasn't improved in 20 rounds.
[10]	train-logloss:0.600507	valid-logloss:0.602926
[20]	train-logloss:0.540315	valid-logloss:0.54569
[30]	train-logloss:0.495154	valid-logloss:0.503447
[40]	train-logloss:0.460269	valid-logloss:0.471273
[50]	train-logloss:0.432758	valid-logloss:0.446433
[60]	train-logloss:0.410453	valid-logloss:0.426939
[70]	train-logloss:0.392581	valid-logloss:0.411771
[80]	train-logloss:0.377723	valid-logloss:0.399318
[90]	train-logloss:0.365933	valid-logloss:0.389516
[100]	train-logloss:0.3559	valid-logloss:0.381551
[110]	train-logloss:0.347084	valid-logloss:0.375032
[120]	train-logloss:0.339534	valid-logloss:0.369721
[130]	train-logloss:0.333442	valid-logloss:0.365377
[140]	train-logloss:0.328079	valid-logloss:0.361857
[150]	train-logloss:0.32329	valid-logloss:0.358693
[160]	train-logloss:0.319361	valid-logloss:0.356207
[170]	train-logloss:0.31589	valid-logloss:0.353918
[180]	train-logloss:0.312881	valid-logloss:0.352137
[190]	train-logloss:0.30974	valid-logloss:0.350555
[200]	train-logloss:0.307392	valid-logloss:0.349221
[210]	train-logloss:0.305339	valid-logloss:0.348169
[220]	train-logloss:0.30304	valid-logloss:0.347102
[230]	train-logloss:0.301474	valid-logloss:0.346287
[240]	train-logloss:0.300088	valid-logloss:0.34569
[250]	train-logloss:0.298736	valid-logloss:0.345087
[260]	train-logloss:0.297417	valid-logloss:0.344499
[270]	train-logloss:0.2962	valid-logloss:0.344016
[280]	train-logloss:0.29479	valid-logloss:0.343474
[290]	train-logloss:0.293647	valid-logloss:0.343059
[300]	train-logloss:0.292204	valid-logloss:0.342627
[310]	train-logloss:0.290784	valid-logloss:0.342127
[320]	train-logloss:0.289592	valid-logloss:0.341714
[330]	train-logloss:0.288643	valid-logloss:0.34144
[340]	train-logloss:0.287647	valid-logloss:0.341184
[350]	train-logloss:0.286852	valid-logloss:0.340901
[360]	train-logloss:0.286029	valid-logloss:0.340713
[370]	train-logloss:0.285243	valid-logloss:0.340487
[380]	train-logloss:0.28439	valid-logloss:0.340339
[390]	train-logloss:0.283515	valid-logloss:0.340138
[399]	train-logloss:0.282731	valid-logloss:0.339888
The test log loss is: 0.33988687149450025
```

In [58]:
```
print("The test log loss is:",log_loss(y_test, predict_y, labels=random_search
.classes_, eps=1e-15))
```

```
The test log loss is: 0.33988687149450025
```

In [60]:
```
predicted_y =np.array(predict_y>0.5,dtype=int)
print("Total number of data points :", len(predicted_y))
plot_confusion_matrix(y_test, predicted_y)
```

```
Total number of data points : 15000
```



In [2]:
```
from prettytable import PrettyTable
x = PrettyTable()
x.field_names = ["Vectorizer", "Model", "Test log loss"]
x.add_row(["TFIDF", "Random Model", 0.88])
x.add_row(["TFIDF", "Logistic Regression", 0.42])
x.add_row(["TFIDF", "Linear SVM", 0.45])
x.add_row(["TFIDF", "XGBoost", 0.33])
print(x)
```

```
+------------+---------------------+---------------+
| Vectorizer |        Model        | Test log loss |
+------------+---------------------+---------------+
|   TFIDF    |     Random Model    |      0.88     |
|   TFIDF    | Logistic Regression |      0.42     |
|   TFIDF    |      Linear SVM     |      0.45     |
|   TFIDF    |       XGBoost       |      0.33     |
+------------+---------------------+---------------+
```

Procedure to solve this Case Study:

1. First I have checked the distribution of class 0 and class 1 points in the whole dataset , found out to be 63.08% of the points belong to class 0 and 36.92% belong to class 1
2. Next i plotted the graph regarding number of unique questions and Repeated questions, found out that no. of unique questions are more than 5L, and no. of repeated questions are about 1L.
3. Then I created some new basic features like: a. freq_qid1 = Frequency of qid1's b. freq_qid2 = Frequency of qid2's c. q1len = Length of q1
   d. q2len = Length of q2
   e. q1_n_words = Number of words in Question 1 f. q2_n_words = Number of words in Question 2
   g. word_Common = (Number of common unique words in Question 1 and Question 2)
   h. word_Total =(Total num of words in Question 1 + Total num of words in Question 2)
   i. word_share = (word_common)/(word_Total) j. freq_q1+freq_q2 = sum total of frequency of qid1 and qid2
   k. freq_q1-freq_q2 = absolute difference of frequency of qid1 and qid2
4. The distributions for word_share have some overlap on the right-hand side, i.e., there are quite a lot of questions with high word similarity
5. The average word share and Common no. of words of qid1 and qid2 is more when they are duplicate
6. Then I have done some basic preprocessing like: a. Removing html tags
   b. Removing Punctuations
   c. Performing stemming d. Removing Stopwords
   e. Expanding contractions.
7. Apart from basic features as discussed in point 3, I have also extracted some advanced features i.e NL and Fuzzy Features like: a. cwc_min b. cwc_max c. csc_min d. csc_max e. ctc_min f. ctc_max g. last_word_eq h. first_word_eq i. abs_len_diff j. mean_len k. fuzz_ratio l. fuzz_partial_ratio m. token_sort_ratio n. token_set_ratio o. longest_substr_ratio
8. Then I combined all the additional features which are formed from point 3,7 and merged with original dataframe and randomly sampled 50k points
9. Then splitted the data to X_train, X_test where 70% belong to X_train and 30% belong to X_test
10. Checked for nan values and filled with white space, due to issues in tfidf vectorization
11. Then I did TFIDFW2V vectorization on question1 and question 2 on both X_train and X_test and added these features to X_train and X_test respectively
12. Dropped the redundant columns
13. First I build a random model which gives Log-loss = 0.88 , and for TFIDFW2W vectorized features on Test data for: a. Logistic regression = 0.53 b. Linear SVM = 0.52 c. XGBoost = 0.36

Assignment:

1. Instead of TFIDFW2V, I tried only on TFIDF vectorization , where I got Log-loss for : a. Logistic Regression = 0.42, b. Linear SVM = 0.45 c. For XGBoost = 0.33

In [ ]: