abs9594_gmail_com_assignment20

December 21, 2019

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 ____ Useful Links ____
- Discussions : 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/AACZd
- Blog 1 : 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

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

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

Metric(s): * log-loss : 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 [0]: 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 [0]: df = pd.read_csv("train.csv")

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

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

```
In [0]: df.head()
```

```
Out[0]:    id  qid1  qid2                                                question1  \
        0   0     1     2  What is the step by step guide to invest in sh...
        1   1     3     4  What is the story of Kohinoor (Koh-i-Noor) Dia...
        2   2     5     6  How can I increase the speed of my internet co...
        3   3     7     8  Why am I mentally very lonely? How can I solve...
        4   4     9    10  Which one dissolve in water quikly sugar, salt...

                                                   question2  is_duplicate
        0  What is the step by step guide to invest in sh...             0
        1  What would happen if the Indian government sto...             0
        2  How can Internet speed be increased by hacking...             0
        3  Find the remainder when [math]23^{24}[/math] i...             0
        4             Which fish would survive in salt water?             0
```

```
In [0]: 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       404290 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:

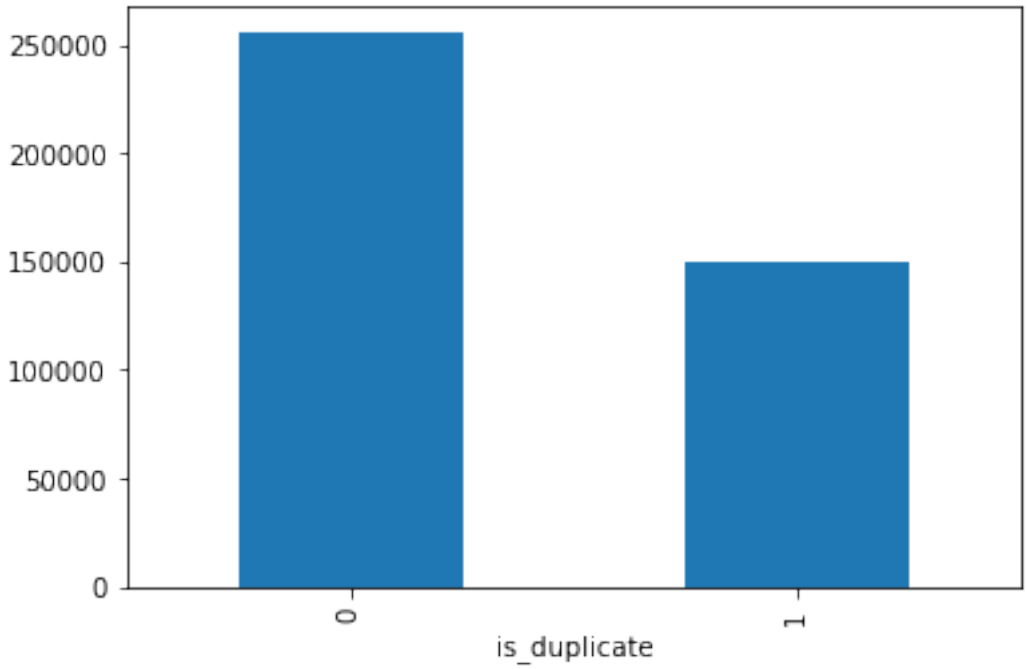
- id: Looks like a simple rowID
- qid{1, 2}: The unique ID of each question in the pair
- question{1, 2}: The actual textual contents of the questions.
- is_duplicate: The label that we are trying to predict - whether the two questions are duplicates of each other.

3.2.1 Distribution of data points among output classes

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

```
In [0]: df.groupby("is_duplicate")['id'].count().plot.bar()
```

```
Out[0]: <matplotlib.axes._subplots.AxesSubplot at 0x22b00727d30>
```




```
In [0]: print('~> Total number of question pairs for training:\n   {}'.format(len(df)))
```

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

```
In [0]: print('~> Question pairs are not Similar (is_duplicate = 0):\n   {}%'.format(100 - round
        print('\n~> Question pairs are Similar (is_duplicate = 1):\n   {}%'.format(round(df['is_
```

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

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

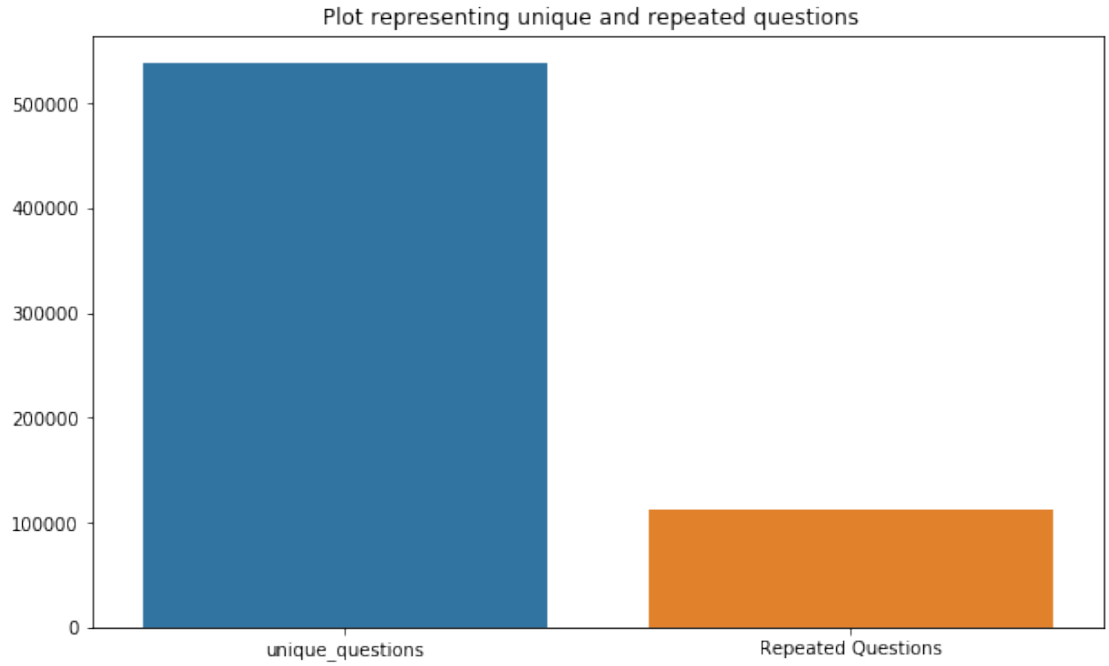
3.2.2 Number of unique questions

```
In [0]: 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
```

```
        print ('Max number of times a single question is repeated: {}\n'.format(max(qids.value_c

        q_vals=qids.value_counts()

        q_vals=q_vals.values
```

```
Total num of  Unique Questions are: 537933

Number of unique questions that appear more than one time: 111780 (20.77953945937505%)

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

```
In [0]: 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 [0]: #checking whether there are any repeated pair of questions
```

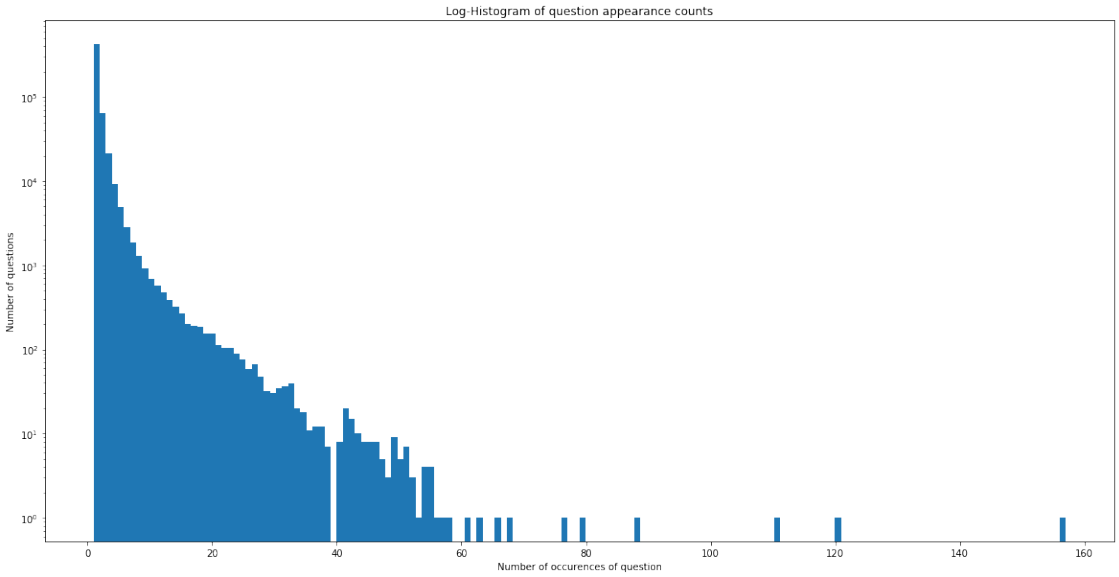
```
        pair_duplicates = df[['qid1','qid2','is_duplicate']].groupby(['qid1','qid2']).count().re

        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 [0]: 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(max(qids.val
```

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



3.2.5 Checking for NULL values

```
In [0]: #Checking whether there are any rows with null values
        nan_rows = df[df.isnull().any(1)]
        print (nan_rows)
```

```
            id    qid1    qid2                          question1 question2  \
105780  105780  174363  174364    How can I develop android app?       NaN
201841  201841  303951  174364  How can I create an Android app?       NaN

        is_duplicate
105780             0
201841             0
```

- There are two rows with null values in question2

```
In [0]: # 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 [0]: 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[0]:

	id	qid1	qid2	question1	question2	is_duplicate	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	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	14	12	10.0	23.0	0.434783	2	0
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	8	13	4.0	20.0	0.200000	5	3
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	14	10	4.0	24.0	0.166667	2	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	1	1	50	65	11	9	0.0	19.0	0.000000	2	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0	3	1	76	39	13	7	2.0	20.0	0.100000	4	2

3.3.1 Analysis of some of the extracted features

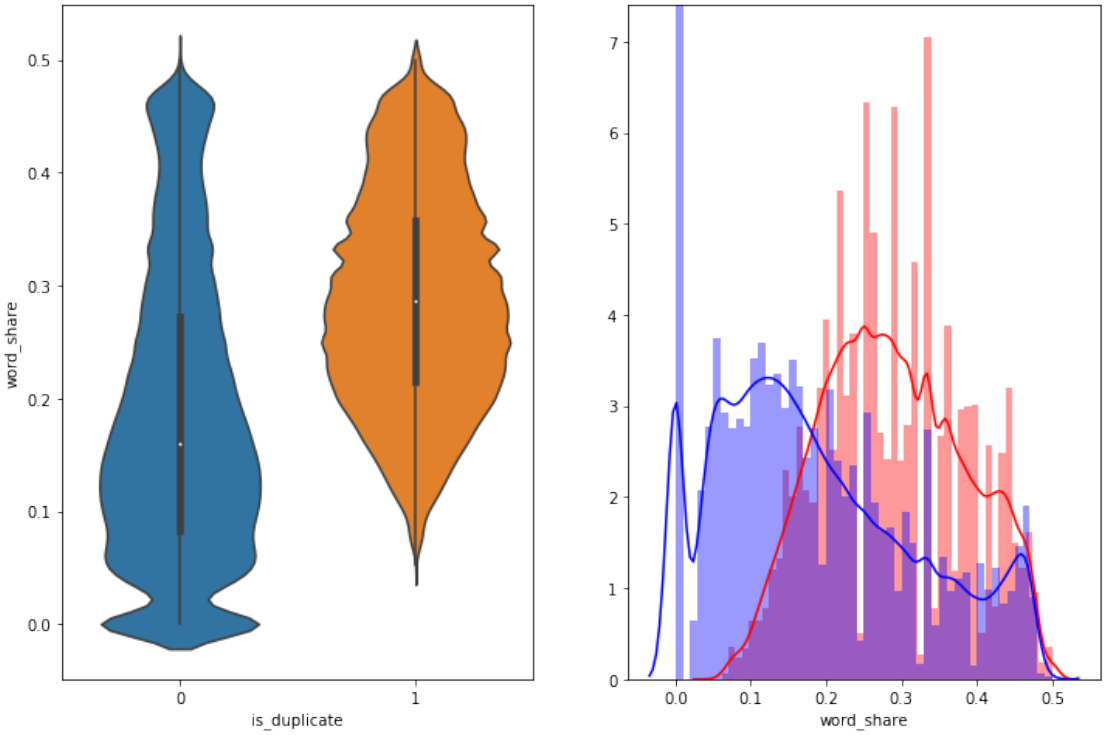
- Here are some questions have only one single words.

```
In [0]: 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_words']== 1]
        print ("Number of Questions with minimum length [question2] :", df[df['q2_n_words']== 1]
```

```
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

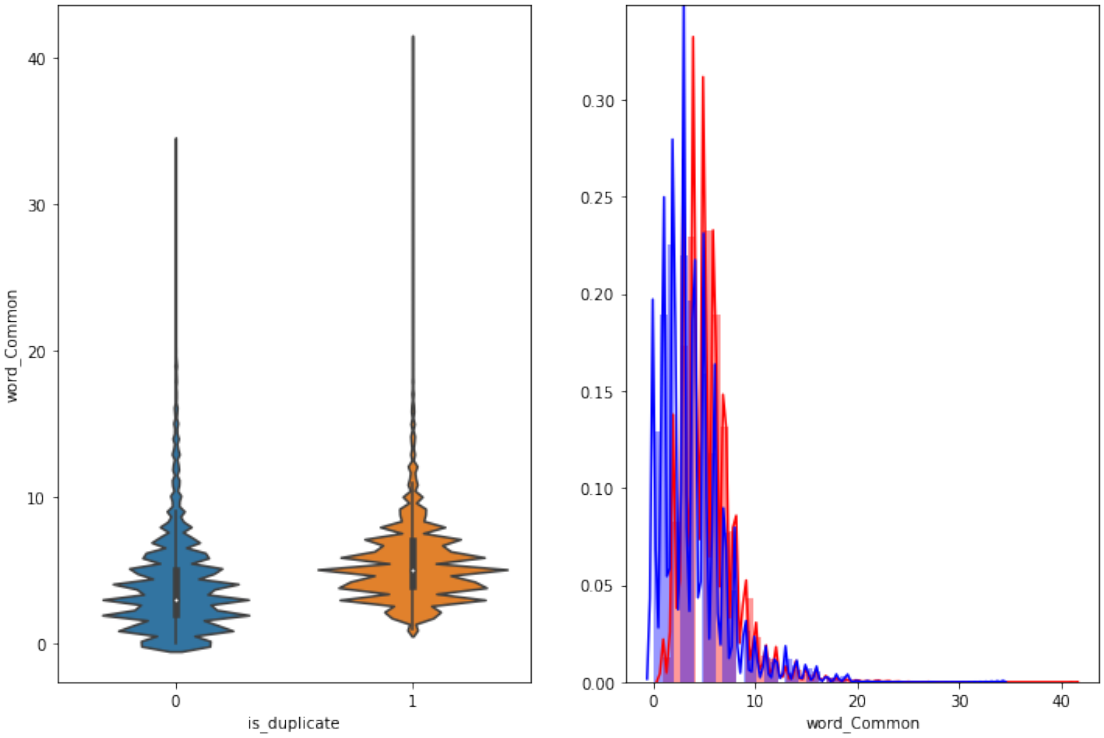
```
In [0]: plt.figure(figsize=(12, 8))

        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", color = 'red
        sns.distplot(df[df['is_duplicate'] == 0.0]['word_share'][0:] , label = "0" , color = 'bl
        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 [0]: 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", color = 're
        sns.distplot(df[df['is_duplicate'] == 0.0]['word_Common'][0:] , label = "0" , color = 'b
        plt.show()
```



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

0.0.1 1.2.1 : EDA: Advanced Feature Extraction.

```
In [0]: 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 strings
# 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-python3-6
from wordcloud import WordCloud, STOPWORDS
from os import path
from PIL import Image
```

In [0]:
```
#https://stackoverflow.com/questions/12468179/unicodedecodeerror-utf8-codec-cant-decode-
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 previous note
```

In [0]:
```
df.head(2)
```

Out[0]:

	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...			

```
1  What would happen if the Indian government sto...             0          4

   freq_qid2  q1len  q2len  q1_n_words  q2_n_words  word_Common  word_Total  \
0          1     66     57          14          12         10.0        23.0
1          1     51     88           8          13          4.0        20.0

   word_share  freq_q1+q2  freq_q1-q2
0    0.434783           2           0
1    0.200000           5           3
```

3.4 Preprocessing of Text

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

```
In [0]: # 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("", "'").replace("", "'"
                                    .replace("won't", "will not").replace("cannot", "can not").re
                                    .replace("n't", " not").replace("what's", "what is").replace(
                                    .replace("'ve", " have").replace("i'm", "i am").replace("'re"
                                    .replace("he's", "he is").replace("she's", "she is").replace(
                                    .replace("%", " percent ").replace("", " rupee ").replace("$"
                                    .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 Q2csc_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 Q2ctc_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 Q2ctc_max = common_token_count / (max(len(q1_tokens), len(q2_tokens))
- **last_word_eq** : Check if First word of both questions is equal or notlast_word_eq = int(q1_tokens[-1] == q2_tokens[-1])
- **first_word_eq** : Check if First word of both questions is equal or notfirst_word_eq = int(q1_tokens[0] == q2_tokens[0])
- **abs_len_diff** : Abs. length differenceabs_len_diff = abs(len(q1_tokens) - len(q2_tokens))
- **mean_len** : Average Token Length of both Questionsmean_len = (len(q1_tokens) + len(q2_tokens))/2
- **fuzz_ratio** : https://github.com/seatgeek/fuzzywuzzy#usage http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/
- **fuzz_partial_ratio** : https://github.com/seatgeek/fuzzywuzzy#usage http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/
- **token_sort_ratio** : https://github.com/seatgeek/fuzzywuzzy#usage http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/
- **token_set_ratio** : https://github.com/seatgeek/fuzzywuzzy#usage 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 Q2longest_substr_ratio = len(longest common substring) / (min(len(q1_tokens), len(q2_tokens))

```
In [0]: 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_DI
            token_features[1] = common_word_count / (max(len(q1_words), len(q2_words)) + SAFE_DI
            token_features[2] = common_stop_count / (min(len(q1_stops), len(q2_stops)) + SAFE_DI
            token_features[3] = common_stop_count / (max(len(q1_stops), len(q2_stops)) + SAFE_DI
            token_features[4] = common_token_count / (min(len(q1_tokens), len(q2_tokens)) + SAFE
            token_features[5] = common_token_count / (max(len(q1_tokens), len(q2_tokens)) + SAFE

            # 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"

        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-
        # https://stackoverflow.com/questions/31806695/when-to-use-which-fuzz-function-to-co
        # https://github.com/seatgeek/fuzzywuzzy
        print("fuzzy features..")

        df["token_set_ratio"]       = df.apply(lambda x: fuzz.token_set_ratio(x["question1"]
        # The token sort approach involves tokenizing the string in question, sorting the to
        # then joining them back into a string We then compare the transformed strings with
        df["token_sort_ratio"]      = df.apply(lambda x: fuzz.token_sort_ratio(x["question1"
        df["fuzz_ratio"]            = df.apply(lambda x: fuzz.QRatio(x["question1"], x["ques
        df["fuzz_partial_ratio"]    = df.apply(lambda x: fuzz.partial_ratio(x["question1"],
        df["longest_substr_ratio"]  = df.apply(lambda x: get_longest_substr_ratio(x["questio
        return df

In [0]: 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[0]:

	id	qid1	qid2	question1	question2	is_duplicate	cwc_min	cwc_max	csc_min	csc_max	...	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
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	...	0.785709	0.0	1.0	2.0	13.0	100	93	93	100	0.982759
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	...	0.466664	0.0	1.0	5.0	12.5	86	63	66	75	0.596154

[2 rows x 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 [0]: 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}} to {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')
        np.savetxt('train_n.txt', n, delimiter=' ', fmt='%s')
```

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

```
In [0]: # 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 : 16109886
Total number of words in non duplicate pair questions : 33193130
```

__ Word Clouds generated from duplicate pair question's text __

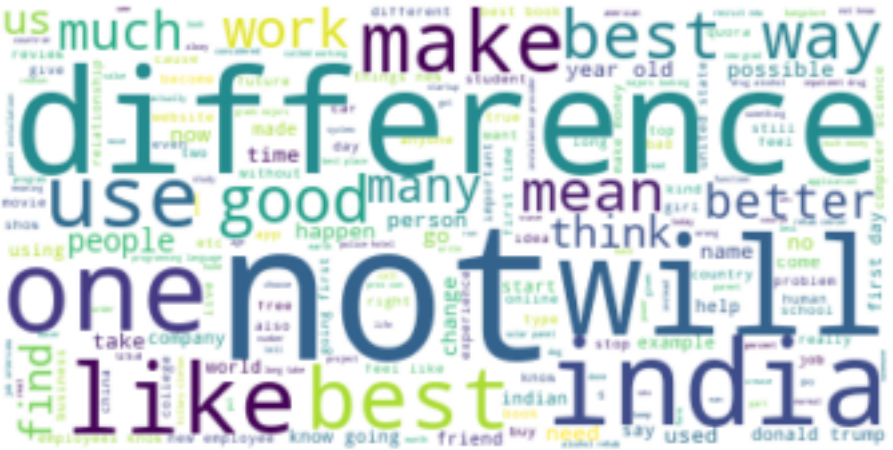
```
In [0]: wc = WordCloud(background_color="white", max_words=len(textp_w), stopwords=stopwords)
        wc.generate(textp_w)
        print ("Word Cloud for Duplicate Question pairs")
        plt.imshow(wc, interpolation='bilinear')
        plt.axis("off")
        plt.show()
```

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

__ Word Clouds generated from non duplicate pair question's text __

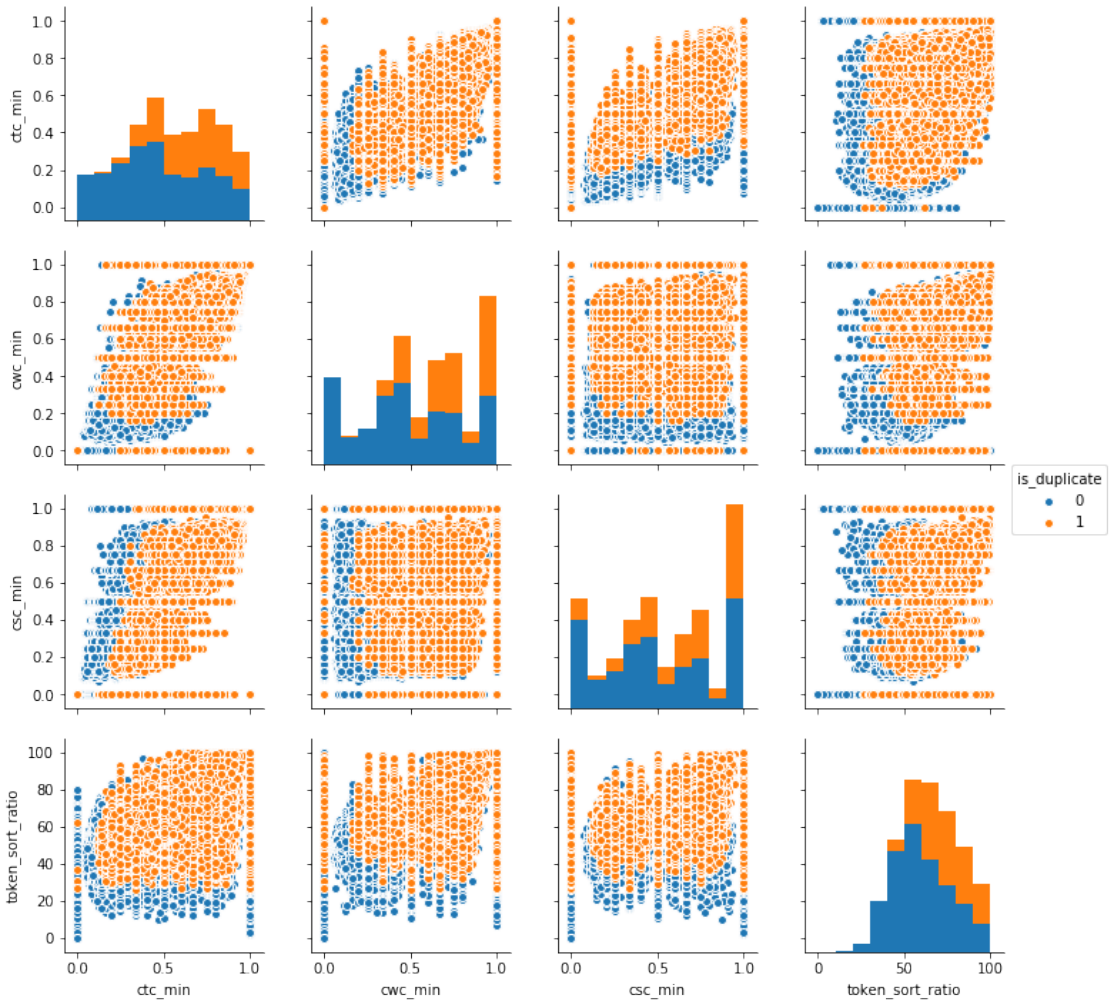
```
In [0]: wc = WordCloud(background_color="white", max_words=len(textn_w),stopwords=stopwords)
        # generate word cloud
        wc.generate(textn_w)
        print ("Word Cloud for non-Duplicate Question pairs:")
        plt.imshow(wc, interpolation='bilinear')
        plt.axis("off")
        plt.show()
```

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

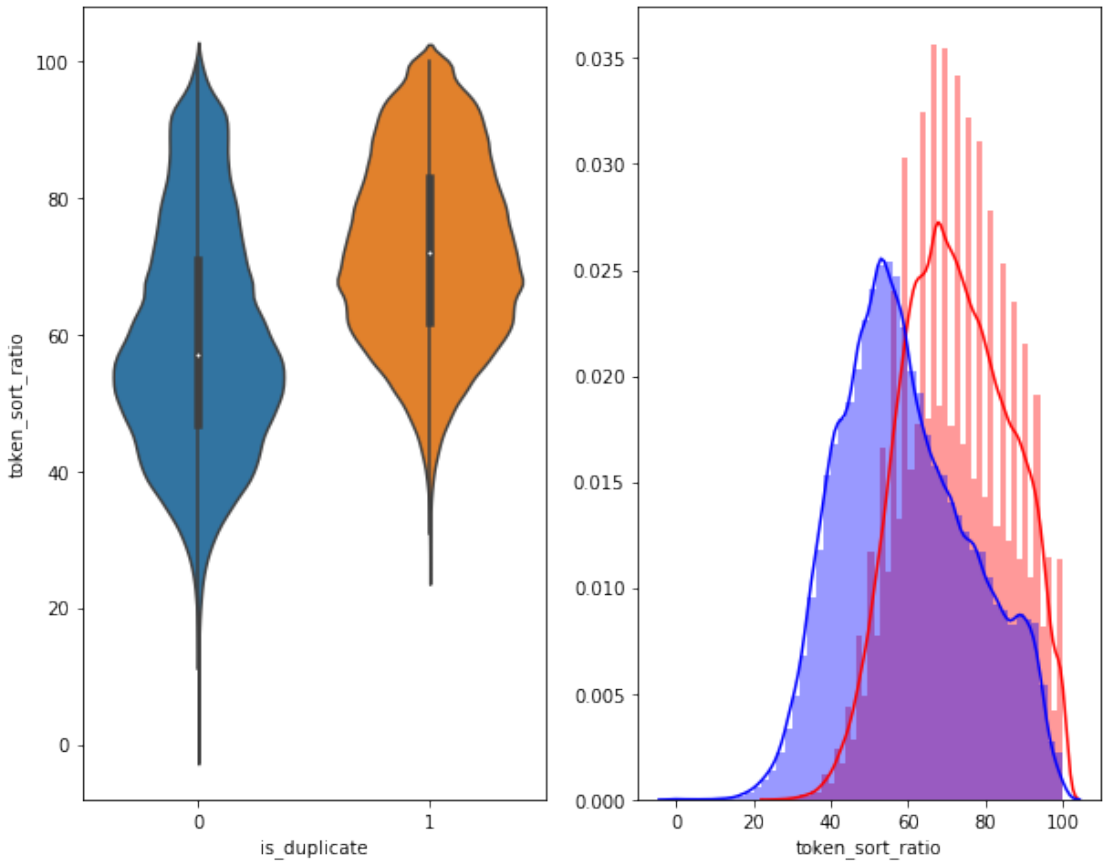


3.5.1.2 Pair plot of features ['ctc_min', 'cwc_min', 'csc_min', 'token_sort_ratio']

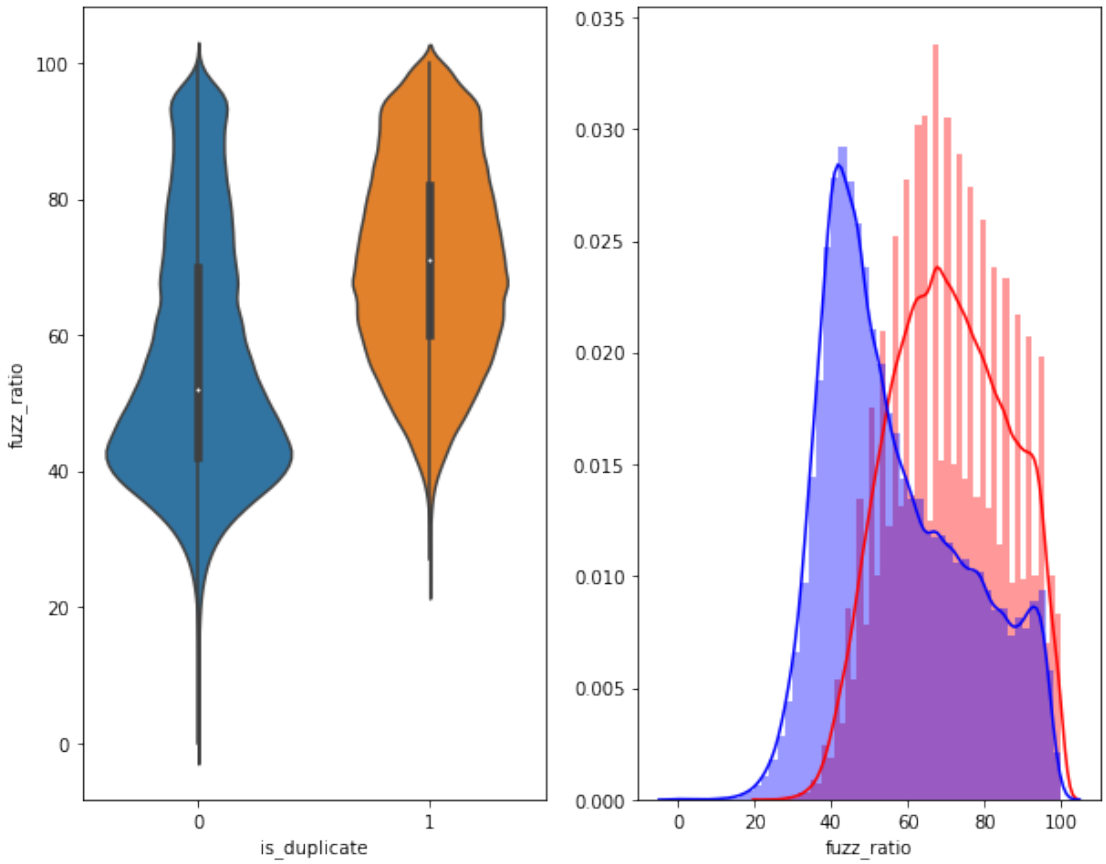
```
In [0]: n = df.shape[0]
        sns.pairplot(df[['ctc_min', 'cwc_min', 'csc_min', 'token_sort_ratio', 'is_duplicate']][0
        plt.show()
```




```
In [0]: # 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
        sns.distplot(df[df['is_duplicate'] == 0.0]['token_sort_ratio'][0:] , label = "0" , color
        plt.show()
```




```
In [0]: 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", color = 'red
        sns.distplot(df[df['is_duplicate'] == 0.0]['fuzz_ratio'][0:] , label = "0" , color = 'bl
        plt.show()
```



3.5.2 Visualization

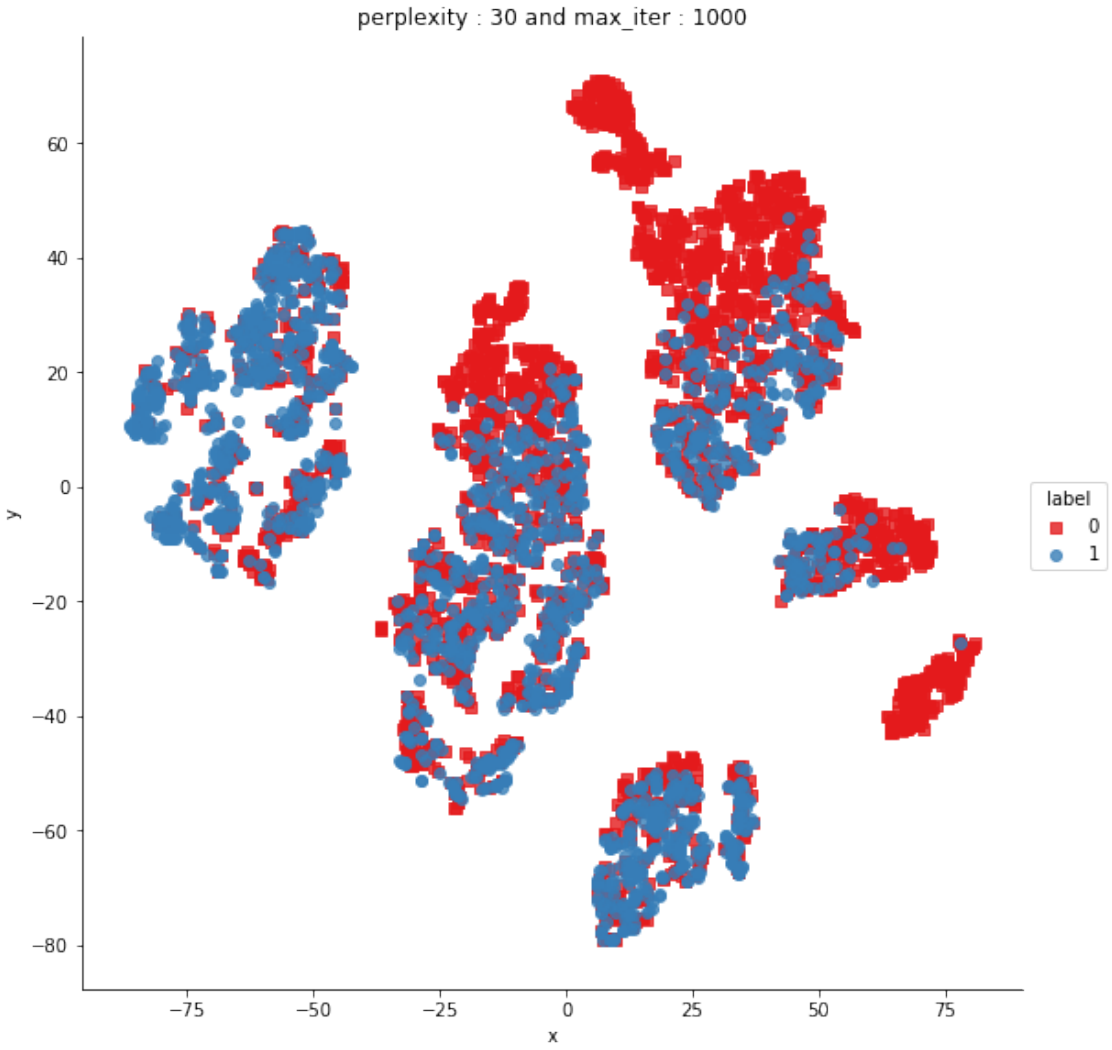
```
In [0]: # Using TSNE for Dimentionality reduction for 15 Features(Generated after cleaning the d

        from sklearn.preprocessing import MinMaxScaler

        dfp_subsampled = df[0:5000]
        X = MinMaxScaler().fit_transform(dfp_subsampled[['cwc_min', 'cwc_max', 'csc_min', 'csc_m
        y = dfp_subsampled['is_duplicate'].values
```

```
In [0]: 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 0.011s...
```

```
[t-SNE] Computed neighbors for 5000 samples in 0.912s...
[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 0.433s
[t-SNE] Iteration 50: error = 80.9244080, gradient norm = 0.0428133 (50 iterations in 13.099s)
[t-SNE] Iteration 100: error = 70.3858795, gradient norm = 0.0100968 (50 iterations in 9.067s)
[t-SNE] Iteration 150: error = 68.6138382, gradient norm = 0.0058392 (50 iterations in 9.602s)
[t-SNE] Iteration 200: error = 67.7700119, gradient norm = 0.0036596 (50 iterations in 9.121s)
[t-SNE] Iteration 250: error = 67.2725067, gradient norm = 0.0034962 (50 iterations in 11.305s)
[t-SNE] KL divergence after 250 iterations with early exaggeration: 67.272507
[t-SNE] Iteration 300: error = 1.7737305, gradient norm = 0.0011918 (50 iterations in 8.289s)
[t-SNE] Iteration 350: error = 1.3720417, gradient norm = 0.0004822 (50 iterations in 10.526s)
[t-SNE] Iteration 400: error = 1.2039998, gradient norm = 0.0002768 (50 iterations in 9.600s)
[t-SNE] Iteration 450: error = 1.1133438, gradient norm = 0.0001881 (50 iterations in 11.827s)
[t-SNE] Iteration 500: error = 1.0579143, gradient norm = 0.0001434 (50 iterations in 8.941s)
[t-SNE] Iteration 550: error = 1.0221983, gradient norm = 0.0001164 (50 iterations in 11.092s)
[t-SNE] Iteration 600: error = 0.9987167, gradient norm = 0.0001039 (50 iterations in 11.467s)
[t-SNE] Iteration 650: error = 0.9831534, gradient norm = 0.0000938 (50 iterations in 11.799s)
[t-SNE] Iteration 700: error = 0.9722011, gradient norm = 0.0000858 (50 iterations in 12.028s)
[t-SNE] Iteration 750: error = 0.9643636, gradient norm = 0.0000799 (50 iterations in 12.120s)
[t-SNE] Iteration 800: error = 0.9584482, gradient norm = 0.0000785 (50 iterations in 11.867s)
[t-SNE] Iteration 850: error = 0.9538348, gradient norm = 0.0000739 (50 iterations in 11.461s)
[t-SNE] Iteration 900: error = 0.9496906, gradient norm = 0.0000712 (50 iterations in 11.023s)
[t-SNE] Iteration 950: error = 0.9463405, gradient norm = 0.0000673 (50 iterations in 11.755s)
[t-SNE] Iteration 1000: error = 0.9432716, gradient norm = 0.0000662 (50 iterations in 11.493s)
[t-SNE] Error after 1000 iterations: 0.943272
```

```
In [0]: 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",mark
        plt.title("perplexity : {} and max_iter : {}".format(30, 1000))
        plt.show()
```




```
In [0]: 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 0.010s...
[t-SNE] Computed neighbors for 5000 samples in 0.935s...
[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 0.363s
[t-SNE] Iteration 50: error = 77.7944183, gradient norm = 0.1014017 (50 iterations in 34.931s)
[t-SNE] Iteration 100: error = 69.2682266, gradient norm = 0.0248657 (50 iterations in 15.147s)
[t-SNE] Iteration 150: error = 67.7877655, gradient norm = 0.0150941 (50 iterations in 13.761s)
[t-SNE] Iteration 200: error = 67.1991119, gradient norm = 0.0126559 (50 iterations in 13.425s)
[t-SNE] Iteration 250: error = 66.8560715, gradient norm = 0.0074975 (50 iterations in 12.904s)
[t-SNE] KL divergence after 250 iterations with early exaggeration: 66.856071
[t-SNE] Iteration 300: error = 1.2356015, gradient norm = 0.0007033 (50 iterations in 13.302s)
[t-SNE] Iteration 350: error = 0.9948602, gradient norm = 0.0001997 (50 iterations in 18.898s)
[t-SNE] Iteration 400: error = 0.9168936, gradient norm = 0.0001430 (50 iterations in 13.397s)
[t-SNE] Iteration 450: error = 0.8863022, gradient norm = 0.0000975 (50 iterations in 16.379s)
[t-SNE] Iteration 500: error = 0.8681002, gradient norm = 0.0000854 (50 iterations in 17.791s)
[t-SNE] Iteration 550: error = 0.8564141, gradient norm = 0.0000694 (50 iterations in 17.060s)
[t-SNE] Iteration 600: error = 0.8470711, gradient norm = 0.0000640 (50 iterations in 15.454s)
[t-SNE] Iteration 650: error = 0.8389117, gradient norm = 0.0000561 (50 iterations in 17.562s)
[t-SNE] Iteration 700: error = 0.8325295, gradient norm = 0.0000529 (50 iterations in 13.443s)
[t-SNE] Iteration 750: error = 0.8268463, gradient norm = 0.0000528 (50 iterations in 17.981s)
[t-SNE] Iteration 800: error = 0.8219477, gradient norm = 0.0000477 (50 iterations in 17.448s)
[t-SNE] Iteration 850: error = 0.8180174, gradient norm = 0.0000490 (50 iterations in 18.376s)
[t-SNE] Iteration 900: error = 0.8150476, gradient norm = 0.0000456 (50 iterations in 17.778s)
[t-SNE] Iteration 950: error = 0.8122067, gradient norm = 0.0000472 (50 iterations in 16.983s)
[t-SNE] Iteration 1000: error = 0.8095787, gradient norm = 0.0000489 (50 iterations in 18.581s)
[t-SNE] Error after 1000 iterations: 0.809579
```

```
In [0]: 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 features')
        fig=dict(data=data, layout=layout)
```

```
py.iplot(fig, filename='3DBubble')
```

3.6 Featurizing text data with tfidf weighted word-vectors

```
In [0]: 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
        import numpy as np
        from tqdm import tqdm

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

```
In [0]: # 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))

        df['text'] = df['question1']+ ' ' + df['question2']
```

```
In [5]: df.head()
```

```
Out[5]:    id  qid1  ...  is_duplicate                                               text
        0   0     1  ...             0  What is the step by step guide to invest in sh...
        1   1     3  ...             0  What is the story of Kohinoor (Koh-i-Noor) Dia...
        2   2     5  ...             0  How can I increase the speed of my internet co...
        3   3     7  ...             0  Why am I mentally very lonely? How can I solve...
        4   4     9  ...             0  Which one dissolve in water quikly sugar, salt...

        [5 rows x 7 columns]
```

```
In [0]: #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 notebook")

        if os.path.isfile('df_fe_without_preprocessing_train.csv'):
            dfppro = pd.read_csv("df_fe_without_preprocessing_train.csv",encoding='latin-1')
        else:
            print("download df_fe_without_preprocessing_train.csv from drive or run previous not
```

```
In [0]: df1 = dfnlp.drop(['qid1','qid2','question1','question2'],axis=1)
        df2 = dfppro.drop(['qid1','qid2','question1','question2','is_duplicate'],axis=1)
        df3 = df.drop(['qid1','qid2'],axis=1)
```

```
In [8]: # dataframe of nlp features
        df1.head()
```

Out[8]:

	id	is_duplicate	...	fuzz_partial_ratio	longest_substr_ratio
0	0	0	...	100	0.982759
1	1	0	...	75	0.596154
2	2	0	...	54	0.166667
3	3	0	...	40	0.039216
4	4	0	...	56	0.175000

[5 rows x 17 columns]

```
In [9]: # data before preprocessing
        df2.head()
```

Out[9]:

	id	freq_qid1	freq_qid2	...	word_share	freq_q1+q2	freq_q1-q2
0	0	1	1	...	0.434783	2	0
1	1	4	1	...	0.200000	5	3
2	2	1	1	...	0.166667	2	0
3	3	1	1	...	0.000000	2	0
4	4	3	1	...	0.100000	4	2

[5 rows x 12 columns]

```
In [10]: print("Number of features in nlp dataframe :", df1.shape[1])
         print("Number of features in preprocessed dataframe :", df2.shape[1])

Number of features in nlp dataframe : 17
Number of features in preprocessed dataframe : 12
```

```
In [0]: result = pd.concat([df1, df2, df3], axis=1)
```

```
In [0]: #removing duplicate columns
        result = result.loc[:,~result.columns.duplicated()]

In [13]: print(result.columns)

Index(['id', 'is_duplicate', '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', 'question1',
       'question2', 'text'],
      dtype='object')


In [0]: data = result.sample(frac=0.25,random_state=200) #random state is a seed value

In [0]: #data.drop(result.index[0], inplace=True)
        y_true = data['is_duplicate']
        data.drop(['id','is_duplicate'], axis=1, inplace=True)

In [16]: data.head()

Out[16]:          cwc_min  ...                                                text
         81194   0.666644  ...  Is there any popular service similar to Quora?...
         181271  0.000000  ...  Whatever happened to Kurt Thomas? What do futu...
         32565   0.799984  ...  Why is Saltwater taffy candy imported in Laos?...
         29667   0.499988  ...  What is the best joke you've ever heard? Pleas...
         271673  0.857131  ...  Information systems are too important to be le...

         [5 rows x 29 columns]

In [0]: 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.cross_validation 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
```

4. Machine Learning Models

4.3 Random train test split(70:30)

```
In [0]: X_train,X_test, y_train, y_test = train_test_split(data, y_true, stratify=y_true, test_s

In [19]: 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 : (70750, 29)
Number of data points in test data : (30322, 29)


In [0]: from sklearn.feature_extraction.text import TfidfVectorizer
        from sklearn.feature_extraction.text import CountVectorizer

        vectorizer = TfidfVectorizer(min_df=0.00009,ngram_range=(1,4), max_features=100000,smoot
        vectorizer.fit(X_train['text'])
```

```
        # dict key:word and value:tf-idf score
        word2tfidf = dict(zip(vectorizer.get_feature_names(), vectorizer.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
- It is trained on Wikipedia and therefore, it is stronger in terms of word semantics.

```
In [21]: # en_vectors_web_lg, which includes over 1 million unique vectors.
         nlp = spacy.load('en_core_web_sm')

         train_vecs1 = []
         # https://github.com/noamraph/tqdm
         # tqdm is used to print the progress bar
         for qu1 in tqdm(list(X_train['question1'])):
             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[str(word1)]
                 except:
                     idf = 0
                 # compute final vec
                 mean_vec1 += vec1 * idf
             mean_vec1 = mean_vec1.mean(axis=0)
             train_vecs1.append(mean_vec1)
         X_train['q1_feats_m'] = list(train_vecs1)
```

```
100%|| 70750/70750 [09:10<00:00, 128.60it/s]
```

```
In [25]: train_vecs2 = []
         for qu2 in tqdm(list(X_train['question2'])):
             doc2 = nlp(qu2)
             mean_vec2 = np.zeros([len(doc1), len(doc2[0].vector)])
             for word2 in doc2:
                 # word2vec
                 vec2 = word2.vector
                 # fetch df score
                 try:
                     idf = word2tfidf[str(word2)]
                 except:
                     #print word
                     idf = 0
```

```
                # compute final vec
                mean_vec2 += vec2 * idf
            mean_vec2 = mean_vec2.mean(axis=0)
            train_vecs2.append(mean_vec2)
        X_train['q2_feats_m'] = list(train_vecs2)
```

```
100%|| 70750/70750 [08:51<00:00, 133.23it/s]
```

```
In [23]: test_vecs1 = []
         # https://github.com/noamraph/tqdm
         # tqdm is used to print the progress bar
         for qu1 in tqdm(list(X_test['question1'])):
             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[str(word1)]
                 except:
                     idf = 0
                 # compute final vec
                 mean_vec1 += vec1 * idf
             mean_vec1 = mean_vec1.mean(axis=0)
             test_vecs1.append(mean_vec1)
         X_test['q1_feats_m'] = list(test_vecs1)
```

```
100%|| 30322/30322 [03:50<00:00, 126.20it/s]
```

```
In [26]: test_vecs2 = []
         for qu2 in tqdm(list(X_test['question2'])):
             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[str(word2)]
                 except:
                     #print word
                     idf = 0
                 # compute final vec
                 mean_vec2 += vec2 * idf
             mean_vec2 = mean_vec2.mean(axis=0)
```

```
            test_vecs2.append(mean_vec2)
        X_test['q2_feats_m'] = list(test_vecs2)
```

```
100%|| 30322/30322 [03:44<00:00, 135.31it/s]
```

```
In [0]: X_train_q1 = pd.DataFrame(X_train.q1_feats_m.values.tolist(), index= X_train.index)
        X_train_q1.columns = [str(col)[:2] + '_q1' for col in X_train_q1.columns]

        X_train_q2 = pd.DataFrame(X_train.q2_feats_m.values.tolist(), index= X_train.index)
        X_train_q2.columns = [str(col)[:2] + '_q2' for col in X_train_q2.columns]

        X_test_q1 = pd.DataFrame(X_test.q1_feats_m.values.tolist(), index= X_test.index)
        X_test_q1.columns = [str(col)[:2] + '_q1' for col in X_test_q1.columns]

        X_test_q2 = pd.DataFrame(X_test.q2_feats_m.values.tolist(), index= X_test.index)
        X_test_q2.columns = [str(col)[:2] + '_q2' for col in X_test_q2.columns]
```

```
In [0]: X_train.drop(['question1','question2','text','q1_feats_m','q2_feats_m'], axis=1, inplace
        X_test.drop(['question1','question2','text','q1_feats_m','q2_feats_m'], axis=1, inplace=
```

```
In [0]: X_train = pd.concat([X_train, X_train_q1, X_train_q2], axis=1)
        X_test = pd.concat([X_test, X_test_q1, X_test_q2], axis=1)
```

```
In [74]: print(X_train.columns)
         print(X_test.columns)
```

```
Index(['cwc_min', 'cwc_max', 'csc_min', 'csc_max', 'ctc_min', 'ctc_max',
       'last_word_eq', 'first_word_eq', 'abs_len_diff', 'mean_len',
       ...
       '86_q2', '87_q2', '88_q2', '89_q2', '90_q2', '91_q2', '92_q2', '93_q2',
       '94_q2', '95_q2'],
      dtype='object', length=218)
Index(['cwc_min', 'cwc_max', 'csc_min', 'csc_max', 'ctc_min', 'ctc_max',
       'last_word_eq', 'first_word_eq', 'abs_len_diff', 'mean_len',
       ...
       '86_q2', '87_q2', '88_q2', '89_q2', '90_q2', '91_q2', '92_q2', '93_q2',
       '94_q2', '95_q2'],
      dtype='object', length=218)
```

```
In [75]: X_train.head()
```

Out[75]:

	cwc_min	cwc_max	csc_min	...	93_q2	94_q2	95_q2
48709	0.499988	0.499988	0.499975	...	48.092580	-6.538450	18.885377
33523	0.153845	0.117646	0.727266	...	-6.036279	-73.409847	-81.272239
82206	0.333322	0.199996	0.666644	...	-14.060089	-25.043521	24.938749
307113	0.666644	0.666644	0.249994	...	19.186386	12.682846	-31.949781
244484	0.666644	0.499988	0.666644	...	10.028221	10.484505	1.730069

[5 rows x 218 columns]

```
In [48]: 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_
         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[0])/test_len, "Class 1: ",int(test_distr[1])/test_len)
```

```
---------- Distribution of output variable in train data ----------
Class 0:  0.6307137809187279 Class 1:  0.36928621908127207
---------- Distribution of output variable in train data ----------
Class 0:  0.6306971835630895 Class 1:  0.3693028164369105
```

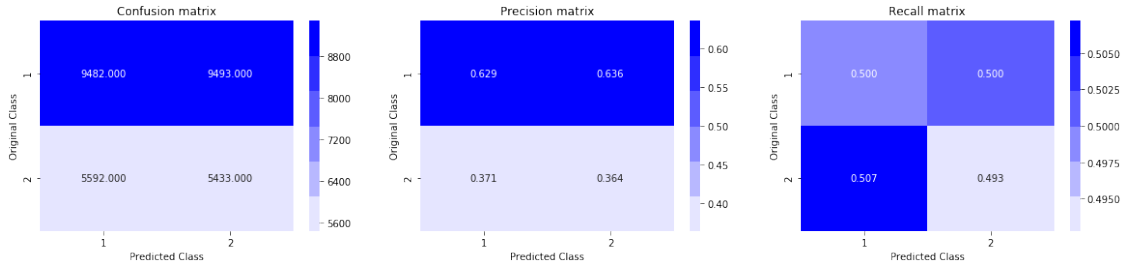
```
In [0]: # This function plots the confusion matrices given y_i, y_i_hat.
        def plot_confusion_matrix(test_y, predict_y):
            C = confusion_matrix(test_y, predict_y)
            # C = 9,9 matrix, each cell (i,j) represents number of points of class i are predict

            A =(((C.T)/(C.sum(axis=1))).T)
            #divid each element of the confusion matrix with the sum of elements in that 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 tu
            # 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 that row
            # C = [[1, 2],
            #     [3, 4]]
            # C.sum(axis = 0)  axis=0 corresonds to columns and axis=1 corresponds to rows in tu
            # 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, yticklabels=lab
    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, yticklabels=lab
    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, yticklabels=lab
    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 [0]: # 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 their 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 0.887242646958
```

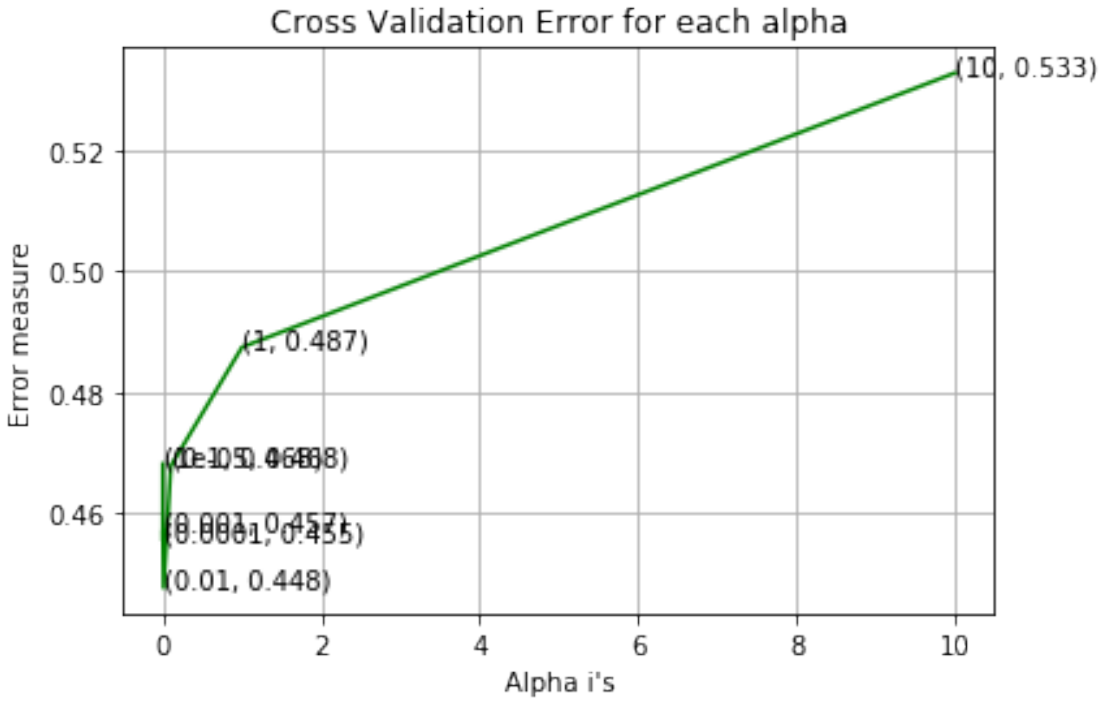


4.4 Logistic Regression with hyperparameter tuning

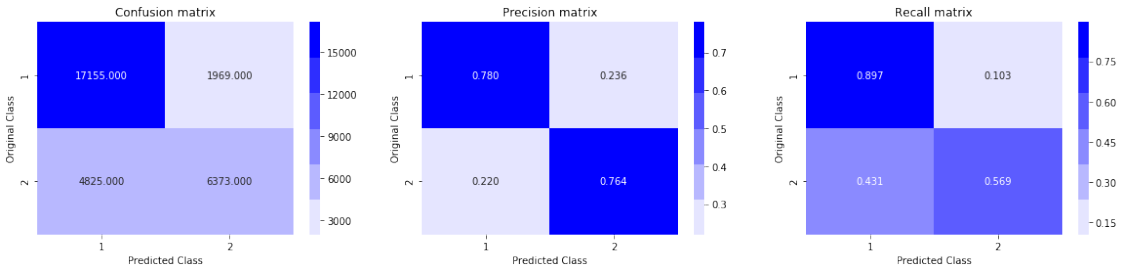
```
In [50]: alpha = [10 ** x for x in range(-5, 2)] # hyperparam for SGD classifier.

         # read more about SGDClassifier() at http://scikit-learn.org/stable/modules/generated/s
         # ------------------------------
         # default parameters
         # SGDClassifier(loss=hinge, penalty=l2, alpha=0.0001, l1_ratio=0.15, fit_intercept=True
         # shuffle=True, verbose=0, epsilon=0.1, n_jobs=1, random_state=None, learning_rate=opti
         # 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 Stochastic Grad
         # 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_, eps=1e-15))
             print('For values of alpha = ', i, "The log loss is:",log_loss(y_test, predict_y, l

         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 is:",log_lo
        predict_y = sig_clf.predict_proba(X_test)
        print('For values of best alpha = ', alpha[best_alpha], "The test log loss is:",log_los
        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: 0.46806619841345376
For values of alpha =  0.0001 The log loss is: 0.45528165449592
For values of alpha =  0.001 The log loss is: 0.4570942913882386
For values of alpha =  0.01 The log loss is: 0.44753027271115714
For values of alpha =  0.1 The log loss is: 0.46771033286101743
For values of alpha =  1 The log loss is: 0.48740112472637187
For values of alpha =  10 The log loss is: 0.5328428568778584
```




```
For values of best alpha =  0.01 The train log loss is: 0.443639272708717
For values of best alpha =  0.01 The test log loss is: 0.44753027271115714
Total number of data points : 30322
```

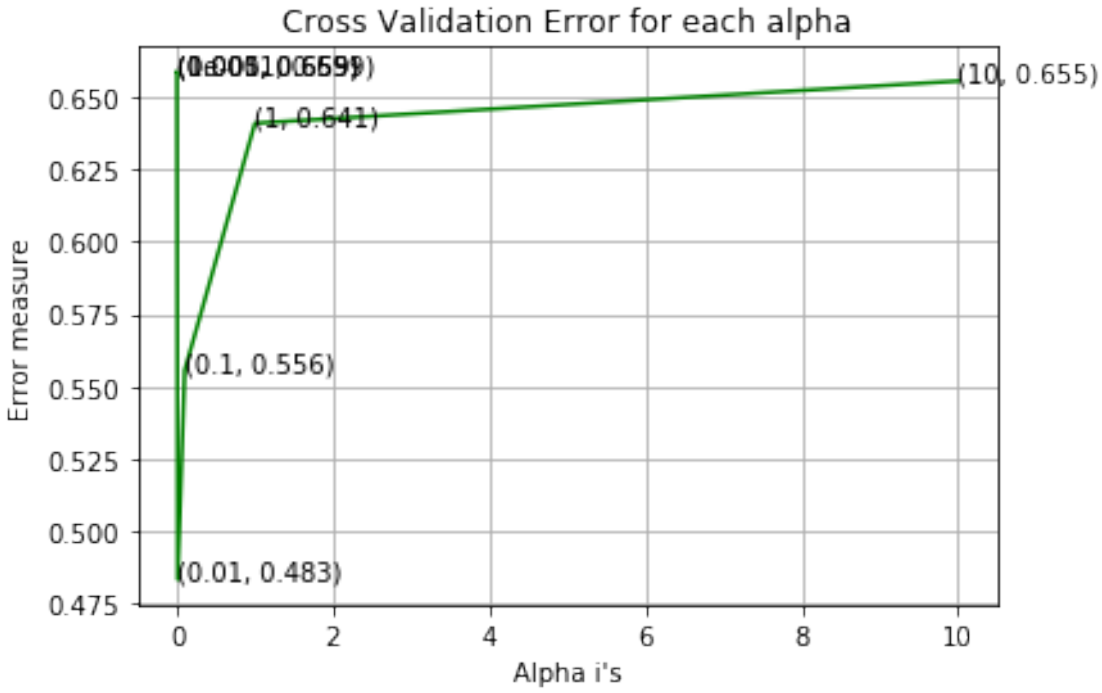


4.5 Linear SVM with hyperparameter tuning

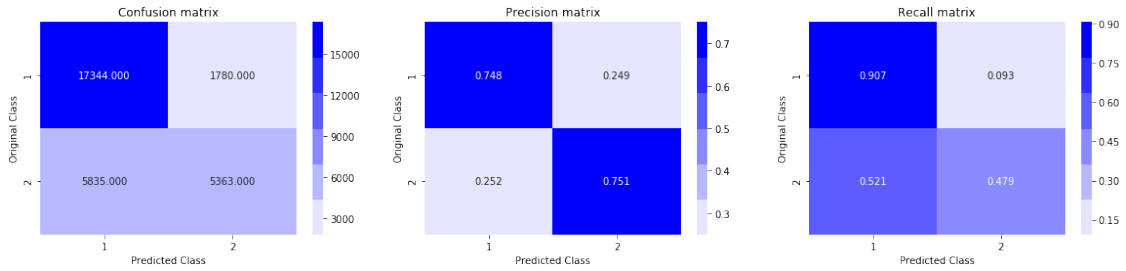
```
In [51]: alpha = [10 ** x for x in range(-5, 2)] # hyperparam for SGD classifier.

         # read more about SGDClassifier() at http://scikit-learn.org/stable/modules/generated/s
         # ------------------------------
         # default parameters
         # SGDClassifier(loss=hinge, penalty=l2, alpha=0.0001, l1_ratio=0.15, fit_intercept=True
         # shuffle=True, verbose=0, epsilon=0.1, n_jobs=1, random_state=None, learning_rate=opti
         # 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 Stochastic Grad
         # 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_, eps=1e-15))
             print('For values of alpha = ', i, "The log loss is:",log_loss(y_test, predict_y, l

         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', random_state=4
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 is:",log_lo
predict_y = sig_clf.predict_proba(X_test)
print('For values of best alpha = ', alpha[best_alpha], "The test log loss is:",log_los
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: 0.6585835850527367
For values of alpha =  0.0001 The log loss is: 0.6585835850527367
For values of alpha =  0.001 The log loss is: 0.6585835850527367
For values of alpha =  0.01 The log loss is: 0.4834232105740497
For values of alpha =  0.1 The log loss is: 0.5555604159728487
For values of alpha =  1 The log loss is: 0.640852156574335
For values of alpha =  10 The log loss is: 0.6553867608607545
```



```
For values of best alpha =  0.01 The train log loss is: 0.47674828851123074
For values of best alpha =  0.01 The test log loss is: 0.4834232105740497
Total number of data points : 30322
```



4.6 XGBoost

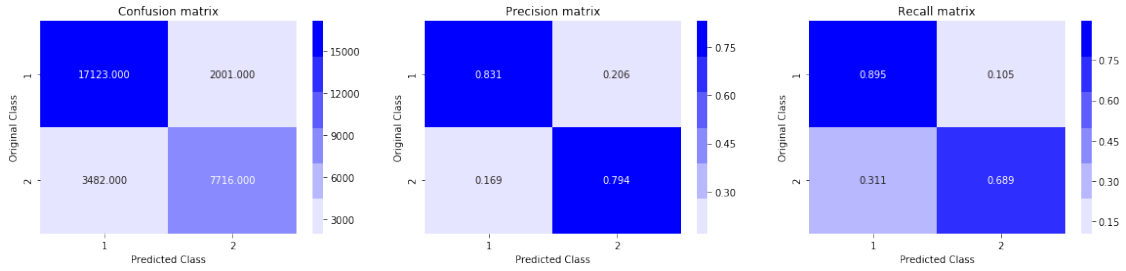
```
In [76]: import xgboost as xgb
         params = {}
         params['objective'] = 'binary:logistic'
         params['eval_metric'] = 'logloss'
         params['eta'] = 0.02
         params['max_depth'] = 4

         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, verbose_eval

         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-1
```

```
[0]        train-logloss:0.684775        valid-logloss:0.68488
Multiple eval metrics have been passed: 'valid-logloss' will be used for early stopping.

Will train until valid-logloss hasn't improved in 20 rounds.
[10]        train-logloss:0.615145        valid-logloss:0.616072
[20]        train-logloss:0.564162        valid-logloss:0.565732
[30]        train-logloss:0.525753        valid-logloss:0.527837
[40]        train-logloss:0.495971        valid-logloss:0.498557
[50]        train-logloss:0.472908        valid-logloss:0.475924
[60]        train-logloss:0.45449        valid-logloss:0.457965
[70]        train-logloss:0.439519        valid-logloss:0.443357
[80]        train-logloss:0.427418        valid-logloss:0.431562
[90]        train-logloss:0.417645        valid-logloss:0.42207
```

```
[100]        train-logloss:0.40971        valid-logloss:0.414473
[110]        train-logloss:0.402963        valid-logloss:0.407998
[120]        train-logloss:0.397423        valid-logloss:0.402673
[130]        train-logloss:0.392421        valid-logloss:0.397861
[140]        train-logloss:0.388383        valid-logloss:0.394008
[150]        train-logloss:0.384921        valid-logloss:0.390803
[160]        train-logloss:0.381772        valid-logloss:0.387902
[170]        train-logloss:0.379111        valid-logloss:0.385489
[180]        train-logloss:0.376784        valid-logloss:0.383387
[190]        train-logloss:0.374619        valid-logloss:0.381449
[200]        train-logloss:0.372683        valid-logloss:0.379686
[210]        train-logloss:0.370896        valid-logloss:0.378177
[220]        train-logloss:0.369188        valid-logloss:0.376756
[230]        train-logloss:0.367782        valid-logloss:0.375559
[240]        train-logloss:0.366326        valid-logloss:0.374372
[250]        train-logloss:0.364832        valid-logloss:0.373152
[260]        train-logloss:0.363287        valid-logloss:0.371796
[270]        train-logloss:0.361897        valid-logloss:0.370672
[280]        train-logloss:0.360653        valid-logloss:0.369702
[290]        train-logloss:0.359304        valid-logloss:0.368682
[300]        train-logloss:0.358127        valid-logloss:0.367761
[310]        train-logloss:0.356925        valid-logloss:0.366922
[320]        train-logloss:0.355791        valid-logloss:0.366091
[330]        train-logloss:0.354713        valid-logloss:0.365321
[340]        train-logloss:0.353628        valid-logloss:0.364513
[350]        train-logloss:0.352634        valid-logloss:0.36377
[360]        train-logloss:0.351646        valid-logloss:0.363083
[370]        train-logloss:0.350685        valid-logloss:0.362364
[380]        train-logloss:0.349855        valid-logloss:0.361767
[390]        train-logloss:0.349017        valid-logloss:0.361236
[399]        train-logloss:0.348213        valid-logloss:0.360611
The test log loss is: 0.36061111083243125
```

```
In [78]: 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 : 30322
```



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 Featurizing text data with tfidf vectors

```
In [0]: 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))

        df['text'] = df['question1']+ ' ' + df['question2']
```

```
In [0]: #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 notebook")

        if os.path.isfile('df_fe_without_preprocessing_train.csv'):
            dfppro = pd.read_csv("df_fe_without_preprocessing_train.csv",encoding='latin-1')
        else:
            print("download df_fe_without_preprocessing_train.csv from drive or run previous not
```

```
In [0]: df1 = dfnlp.drop(['qid1','qid2','question1','question2'],axis=1)
        df2 = dfppro.drop(['qid1','qid2','question1','question2','is_duplicate'],axis=1)
        df3 = df.drop(['qid1','qid2'],axis=1)
```

```
In [0]: # dataframe of nlp features
        df1.head()
```

```
Out[0]:    id  is_duplicate  ...  fuzz_partial_ratio  longest_substr_ratio
        0   0             0  ...                 100              0.982759
        1   1             0  ...                  75              0.596154
        2   2             0  ...                  54              0.166667
        3   3             0  ...                  40              0.039216
        4   4             0  ...                  56              0.175000

        [5 rows x 17 columns]
```

```
In [0]: # data before preprocessing
        df2.head()
```

Out[0]:

	id	freq_qid1	freq_qid2	...	word_share	freq_q1+q2	freq_q1-q2
0	0	1	1	...	0.434783	2	0
1	1	4	1	...	0.200000	5	3
2	2	1	1	...	0.166667	2	0
3	3	1	1	...	0.000000	2	0
4	4	3	1	...	0.100000	4	2

[5 rows x 12 columns]

```
In [0]: print("Number of features in nlp dataframe :", df1.shape[1])
        print("Number of features in preprocessed dataframe :", df2.shape[1])
```

```
Number of features in nlp dataframe : 17
Number of features in preprocessed dataframe : 12
```

```
In [0]: result = pd.concat([df1, df2, df3], axis=1)
```

```
In [0]: #removing duplicate columns
        result = result.loc[:,~result.columns.duplicated()]
```

```
In [0]: print(result.columns)
```

```
Index(['id', 'is_duplicate', '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', 'question1',
       'question2', 'text'],
      dtype='object')
```

```
In [0]: data = result.sample(frac=0.25,random_state=200) #random state is a seed value
```

```
In [0]: #data.drop(result.index[0], inplace=True)
        y_true = data['is_duplicate']
        data.drop(['id','is_duplicate'], axis=1, inplace=True)
```

```
In [0]: data.head()
```

Out[0]:

	cwc_min	...	text
81194	0.666644	...	Is there any popular service similar to Quora?...
181271	0.000000	...	Whatever happened to Kurt Thomas? What do futu...
32565	0.799984	...	Why is Saltwater taffy candy imported in Laos?...
29667	0.499988	...	What is the best joke you've ever heard? Pleas...
271673	0.857131	...	Information systems are too important to be le...

[5 rows x 29 columns]

5.3 Random train test split(70:30)

```
In [0]: X_train,X_test, y_train, y_test = train_test_split(data,y_true, stratify=y_true, test_si

In [0]: vectorizer = TfidfVectorizer(min_df=0.00009,ngram_range=(1,4), max_features=100000,smoot
        vectorizer.fit(X_train['text'])

        tfidf_train_q1 = vectorizer.transform(X_train['question1'])
        tfidf_train_q2 = vectorizer.transform(X_train['question2'])

        tfidf_test_q1 = vectorizer.transform(X_test['question1'])
        tfidf_test_q2 = vectorizer.transform(X_test['question2'])

        print('No of Tfidf features',len(vectorizer.get_feature_names()))

No of Tfidf features 53411


In [0]: X_train.drop(['question1','question2','text'], axis=1, inplace=True)
        X_test.drop(['question1','question2','text'], axis=1, inplace=True)

In [0]: X_train.columns
        X_test.columns

Out[0]: Index(['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'],
              dtype='object')

In [0]: from scipy.sparse import hstack
        X_train_tfidf = hstack((X_train.values,tfidf_train_q1,tfidf_train_q2))
        X_test_tfidf = hstack((X_test.values,tfidf_test_q1,tfidf_test_q2))

In [0]: print("Number of data points in train data :",X_train_tfidf.shape)
        print("Number of data points in test data :",X_test_tfidf.shape)

Number of data points in train data : (70750, 106848)
Number of data points in test data : (30322, 106848)


In [0]: 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_l
        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[0])/test_len, "Class 1: ",int(test_distr[1])/test_len)
```

```
---------- Distribution of output variable in train data ----------
Class 0:  0.6307137809187279 Class 1:  0.36928621908127207
---------- Distribution of output variable in train data ----------
Class 0:  0.6306971835630895 Class 1:  0.3693028164369105
```

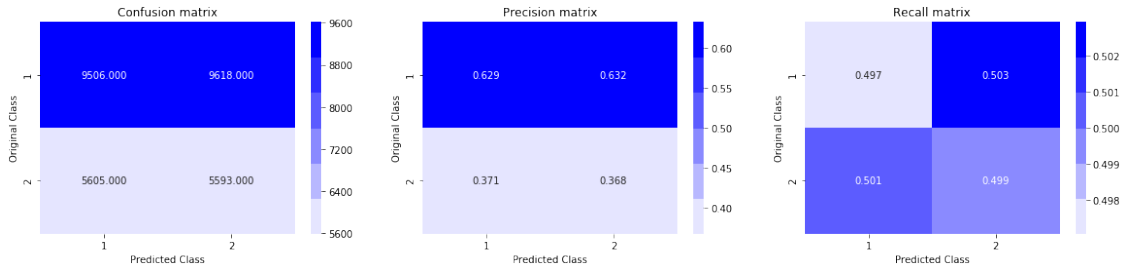
```
In [0]: # This function plots the confusion matrices given y_i, y_i_hat.
        def plot_confusion_matrix(test_y, predict_y):
            C = confusion_matrix(test_y, predict_y)
            # C = 9,9 matrix, each cell (i,j) represents number of points of class i are predict

            A =(((C.T)/(C.sum(axis=1))).T)
            #divid each element of the confusion matrix with the sum of elements in that 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 tu
            # 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 that row
            # C = [[1, 2],
            #     [3, 4]]
            # C.sum(axis = 0)  axis=0 corresonds to columns and axis=1 corresponds to rows in tu
            # 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, yticklabels=lab
            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, yticklabels=lab
```

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

    plt.show()
```

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

```
In [0]: # 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 their 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 0.8870054752077356
```

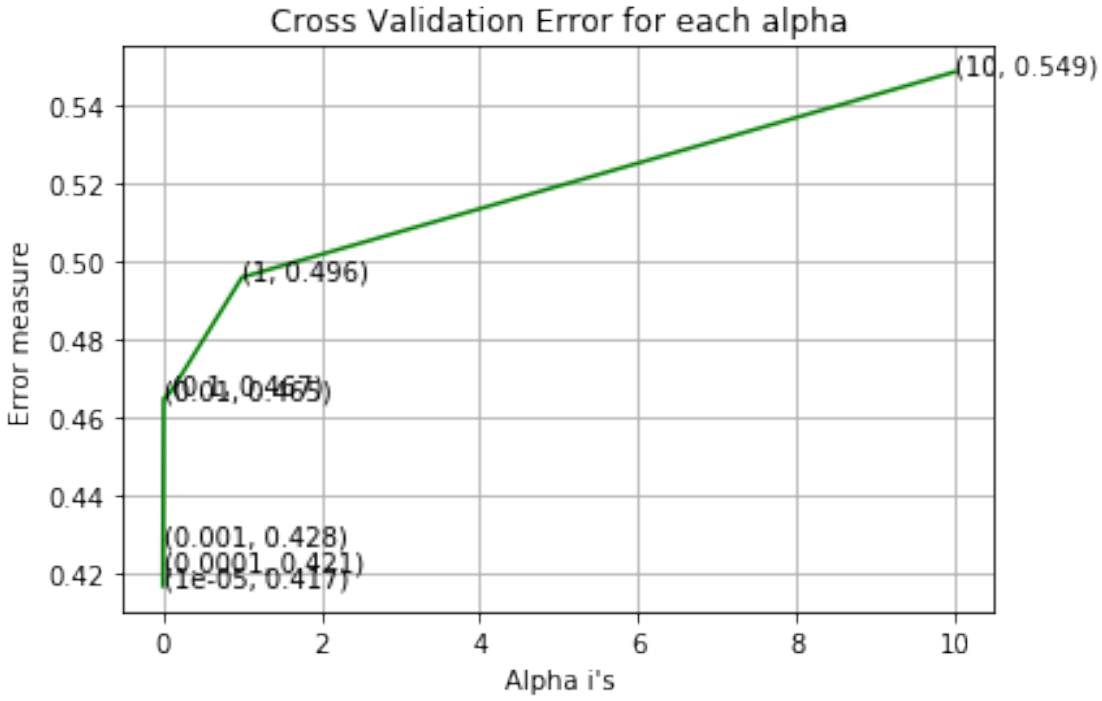


5.5 Logistic Regression with hyperparameter tuning

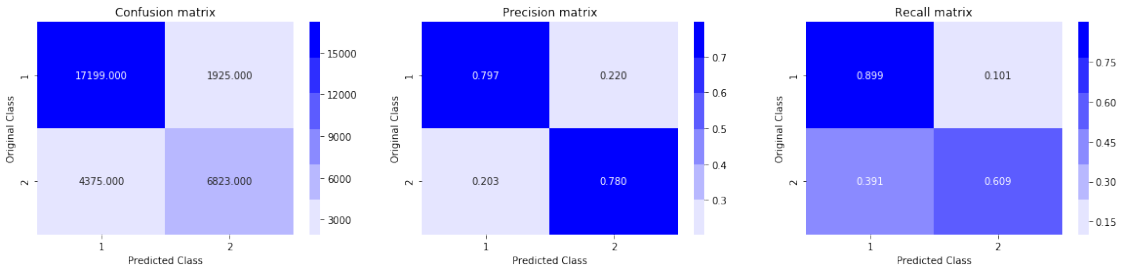
```
In [0]: alpha = [10 ** x for x in range(-5, 2)] # hyperparam for SGD classifier.

        # read more about SGDClassifier() at http://scikit-learn.org/stable/modules/generated/sk
        # ------------------------------
        # default parameters
```

```
# SGDClassifier(loss=hinge, penalty=l2, alpha=0.0001, l1_ratio=0.15, fit_intercept=True,
# shuffle=True, verbose=0, epsilon=0.1, n_jobs=1, random_state=None, learning_rate=optim
# 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 Stochastic Gradi
# 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_tfidf, y_train)
    sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
    sig_clf.fit(X_train_tfidf, y_train)
    predict_y = sig_clf.predict_proba(X_test_tfidf)
    log_error_array.append(log_loss(y_test, predict_y, labels=clf.classes_, eps=1e-15))
    print('For values of alpha = ', i, "The log loss is:",log_loss(y_test, predict_y, la

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_tfidf, y_train)
sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
sig_clf.fit(X_train_tfidf, y_train)

predict_y = sig_clf.predict_proba(X_train_tfidf)
print('For values of best alpha = ', alpha[best_alpha], "The train log loss is:",log_los
predict_y = sig_clf.predict_proba(X_test_tfidf)
print('For values of best alpha = ', alpha[best_alpha], "The test log loss is:",log_loss
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: 0.41699200807385883
For values of alpha =  0.0001 The log loss is: 0.42114586083917255
For values of alpha =  0.001 The log loss is: 0.42812272678738417
For values of alpha =  0.01 The log loss is: 0.46496200483914557
For values of alpha =  0.1 The log loss is: 0.4666941412753165
For values of alpha =  1 The log loss is: 0.49607234536805983
For values of alpha =  10 The log loss is: 0.5486936583287718
```




```
For values of best alpha =  1e-05 The train log loss is: 0.40722784329536943
For values of best alpha =  1e-05 The test log loss is: 0.41699200807385883
Total number of data points : 30322
```

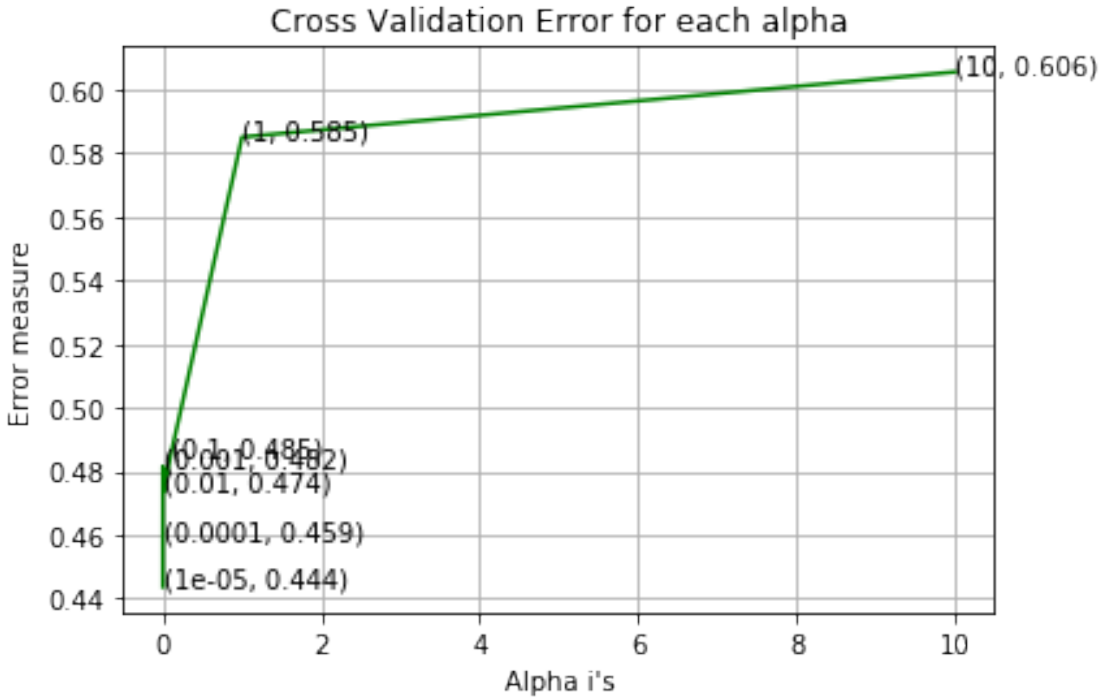


5.6 Linear SVM with hyperparameter tuning

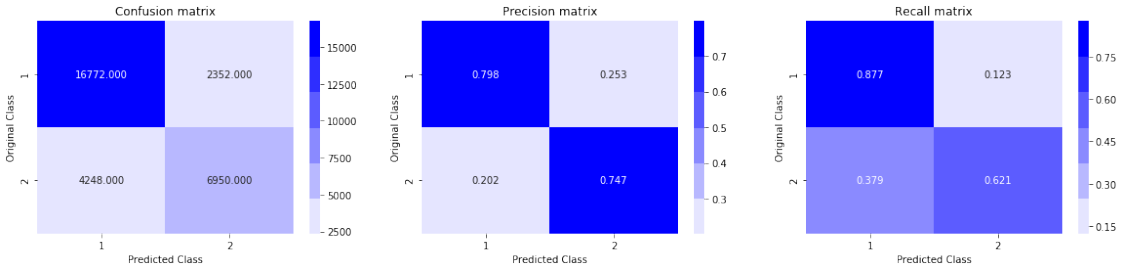
```
In [0]: alpha = [10 ** x for x in range(-5, 2)] # hyperparam for SGD classifier.

        # read more about SGDClassifier() at http://scikit-learn.org/stable/modules/generated/sk
        # ------------------------------
        # default parameters
        # SGDClassifier(loss=hinge, penalty=l2, alpha=0.0001, l1_ratio=0.15, fit_intercept=True,
        # shuffle=True, verbose=0, epsilon=0.1, n_jobs=1, random_state=None, learning_rate=optim
        # 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 Stochastic Gradi
        # 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_tfidf, y_train)
            sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
            sig_clf.fit(X_train_tfidf, y_train)
            predict_y = sig_clf.predict_proba(X_test_tfidf)
            log_error_array.append(log_loss(y_test, predict_y, labels=clf.classes_, eps=1e-15))
            print('For values of alpha = ', i, "The log loss is:",log_loss(y_test, predict_y, la

        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', random_state=42
        clf.fit(X_train_tfidf, y_train)
        sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
        sig_clf.fit(X_train_tfidf, y_train)

        predict_y = sig_clf.predict_proba(X_train_tfidf)
        print('For values of best alpha = ', alpha[best_alpha], "The train log loss is:",log_los
        predict_y = sig_clf.predict_proba(X_test_tfidf)
```

```
        print('For values of best alpha = ', alpha[best_alpha], "The test log loss is:",log_loss
        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: 0.4436860506812293
For values of alpha =  0.0001 The log loss is: 0.45861686822835995
For values of alpha =  0.001 The log loss is: 0.48156968603462885
For values of alpha =  0.01 The log loss is: 0.47381851430925126
For values of alpha =  0.1 The log loss is: 0.4851609147593898
For values of alpha =  1 The log loss is: 0.585050088638443
For values of alpha =  10 The log loss is: 0.6055084516261086
```




```
For values of best alpha =  1e-05 The train log loss is: 0.43337322427545905
For values of best alpha =  1e-05 The test log loss is: 0.4436860506812293
Total number of data points : 30322
```



5.7 XGBoost

```
In [0]: from xgboost import XGBClassifier
        from sklearn.model_selection import RandomizedSearchCV
        from sklearn.calibration import CalibratedClassifierCV
        import datetime
        import time

        parameters = {
            'eta': [0.01, 0.1, 0.3],
            'max_depth': [3,4,5],
            'subsample': [0.9, 1.0],
            'colsample_bytree': [0.9, 1.0],
            'learning_rate' : [0.01,0.1],
            }


        clf = XGBClassifier(silent=False,eval_metric='logloss',num_boost_round=50,n_estimators=1

        clf_random = RandomizedSearchCV(estimator = clf, param_distributions = parameters, n_ite

        start_time = datetime.datetime.now().time().strftime('%H:%M:%S')

        clf_random.fit(X_train_tfidf, y_train)

        end_time = datetime.datetime.now().time().strftime('%H:%M:%S')
        total_time=(datetime.datetime.strptime(end_time,'%H:%M:%S') - datetime.datetime.strptime

        print("Total time taken : ",total_time)
        print("paramters : ",clf_random.best_params_)
```

```
Fitting 3 folds for each of 40 candidates, totalling 120 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent workers.
Memmapping (shape=(4346097,), dtype=int32) to new file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to new file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
```

```
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done   1 tasks      | elapsed:  1.6min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done   2 tasks      | elapsed:  1.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done   3 tasks      | elapsed:  1.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
```

```
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done   4 tasks      | elapsed:  1.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done   5 tasks      | elapsed:  3.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done   6 tasks      | elapsed:  3.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done   7 tasks      | elapsed:  3.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done   8 tasks      | elapsed:  3.5min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
```

```
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done   9 tasks      | elapsed:  4.1min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  10 tasks      | elapsed:  4.1min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  11 tasks      | elapsed:  4.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  12 tasks      | elapsed:  4.6min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  13 tasks      | elapsed:  5.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
```

```
[Parallel(n_jobs=-1)]: Done  14 tasks      | elapsed:  6.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  15 tasks      | elapsed:  6.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  16 tasks      | elapsed:  6.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  17 tasks      | elapsed:  7.5min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  18 tasks      | elapsed:  7.6min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  19 tasks      | elapsed:  8.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
```

```
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  20 tasks      | elapsed:  8.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  21 tasks      | elapsed:  9.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  22 tasks      | elapsed:  9.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  23 tasks      | elapsed:  9.7min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  24 tasks      | elapsed:  9.7min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
```

```
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  25 tasks      | elapsed: 10.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed: 10.5min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  27 tasks      | elapsed: 10.8min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  28 tasks      | elapsed: 10.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  29 tasks      | elapsed: 11.6min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
```

```
[Parallel(n_jobs=-1)]: Done  30 tasks      | elapsed: 11.6min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  31 tasks      | elapsed: 12.7min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  32 tasks      | elapsed: 12.8min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  33 tasks      | elapsed: 12.8min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  34 tasks      | elapsed: 13.5min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  35 tasks      | elapsed: 13.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
```

```
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  36 tasks      | elapsed: 13.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  37 tasks      | elapsed: 14.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  38 tasks      | elapsed: 15.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  39 tasks      | elapsed: 15.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  40 tasks      | elapsed: 15.5min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
```

```
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  41 tasks      | elapsed: 15.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  42 tasks      | elapsed: 16.6min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  43 tasks      | elapsed: 17.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  44 tasks      | elapsed: 17.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  45 tasks      | elapsed: 17.7min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
```

```
[Parallel(n_jobs=-1)]: Done  46 tasks      | elapsed: 18.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  47 tasks      | elapsed: 19.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  48 tasks      | elapsed: 19.1min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  49 tasks      | elapsed: 19.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  50 tasks      | elapsed: 19.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  51 tasks      | elapsed: 20.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
```

```
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  52 tasks      | elapsed: 20.5min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  53 tasks      | elapsed: 20.5min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  54 tasks      | elapsed: 21.1min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  55 tasks      | elapsed: 21.8min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  56 tasks      | elapsed: 21.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
```

```
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  57 tasks      | elapsed: 22.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  58 tasks      | elapsed: 22.7min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  59 tasks      | elapsed: 23.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  60 tasks      | elapsed: 23.5min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  61 tasks      | elapsed: 23.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
```

```
[Parallel(n_jobs=-1)]: Done  62 tasks      | elapsed: 24.7min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  63 tasks      | elapsed: 25.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  64 tasks      | elapsed: 25.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  65 tasks      | elapsed: 25.8min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  66 tasks      | elapsed: 26.5min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  67 tasks      | elapsed: 27.1min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
```

```
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  68 tasks      | elapsed: 27.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  69 tasks      | elapsed: 27.6min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  70 tasks      | elapsed: 28.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  71 tasks      | elapsed: 28.6min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  72 tasks      | elapsed: 28.7min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
```

```
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  73 tasks      | elapsed: 29.1min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  74 tasks      | elapsed: 29.6min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  75 tasks      | elapsed: 29.8min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  76 tasks      | elapsed: 30.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  77 tasks      | elapsed: 30.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
```

```
[Parallel(n_jobs=-1)]: Done  78 tasks      | elapsed: 30.7min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  79 tasks      | elapsed: 31.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  80 tasks      | elapsed: 31.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  81 tasks      | elapsed: 31.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  82 tasks      | elapsed: 31.8min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  83 tasks      | elapsed: 32.1min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
```

```
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  84 tasks      | elapsed: 32.5min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  85 tasks      | elapsed: 32.5min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  86 tasks      | elapsed: 32.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  87 tasks      | elapsed: 33.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  88 tasks      | elapsed: 34.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
```

```
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  89 tasks      | elapsed: 34.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  90 tasks      | elapsed: 34.8min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  91 tasks      | elapsed: 35.1min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  92 tasks      | elapsed: 35.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  93 tasks      | elapsed: 36.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
```

```
[Parallel(n_jobs=-1)]: Done  94 tasks      | elapsed: 36.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  95 tasks      | elapsed: 36.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  96 tasks      | elapsed: 37.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  97 tasks      | elapsed: 38.1min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  98 tasks      | elapsed: 38.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done  99 tasks      | elapsed: 38.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
```

```
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 100 tasks      | elapsed: 38.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 101 tasks      | elapsed: 39.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 102 tasks      | elapsed: 39.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 103 tasks      | elapsed: 40.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 104 tasks      | elapsed: 40.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
```

```
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 105 tasks      | elapsed: 40.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 106 tasks      | elapsed: 41.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 107 tasks      | elapsed: 41.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 108 tasks      | elapsed: 41.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 109 tasks      | elapsed: 42.4min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
```

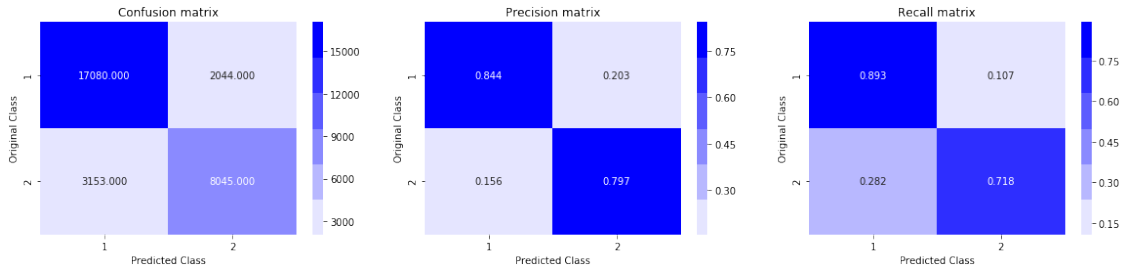
```
[Parallel(n_jobs=-1)]: Done 110 tasks      | elapsed: 42.8min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 111 tasks      | elapsed: 43.2min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 112 tasks      | elapsed: 43.3min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 113 tasks      | elapsed: 43.9min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47166,), dtype=int64).
Pickling array (shape=(23584,), dtype=int64).
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 115 out of 120 | elapsed: 45.1min remaining:  2.0min
Memmapping (shape=(4346097,), dtype=int32) to old file /dev/shm/joblib_memmapping_folder_147_896
Pickling array (shape=(70751,), dtype=int32).
Memmapping (shape=(4346097,), dtype=float64) to old file /dev/shm/joblib_memmapping_folder_147_8
```

```
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(70750,), dtype=int64).
Pickling array (shape=(47167,), dtype=int64).
Pickling array (shape=(23583,), dtype=int64).
[Parallel(n_jobs=-1)]: Done 117 out of 120 | elapsed: 45.8min remaining:  1.2min
[Parallel(n_jobs=-1)]: Done 120 out of 120 | elapsed: 46.4min finished
Total time taken :  0:47:03
paramters :  {'subsample': 0.9, 'max_depth': 5, 'learning_rate': 0.1, 'eta': 0.01, 'colsample_by
```

```
In [0]: print("paramters : ",clf_random.best_params_)
        print("The best train log loss is : ", clf_random.best_score_)

paramters :  {'subsample': 0.9, 'max_depth': 5, 'learning_rate': 0.1, 'eta': 0.01, 'colsample_by
The best train log loss is :  0.8278727915194346
```

```
In [0]: clf = XGBClassifier(silent=False,eval_metric='logloss',num_boost_round=100,subsample = 0
        clf.fit(X_train_tfidf, y_train)

        sig_clf = CalibratedClassifierCV(clf, method="sigmoid")
        sig_clf.fit(X_train_tfidf, y_train)

        predict_y = sig_clf.predict_proba(X_train_tfidf)
        print("The train log loss is:",log_loss(y_train, predict_y, labels=clf.classes_, eps=1e-
        predict_y = sig_clf.predict_proba(X_test_tfidf)
        print("The test log loss is:",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)
```

```
The train log loss is: 0.33327964514677144
The test log loss is: 0.3555942362170514
Total number of data points : 30322
```



SUMMARY

```
In [82]: import tabletext

         data = [['Model','Vectorizer','Tr Los','Ts Los'],
                 ['Random','-','-','0.88'],
                 ['Logistic Regression','TFIDF-W2V','0.44','0.44'],
                 ['Linear SVM','TFIDF-W2V','0.47','0.48'],
                 ['XGBOOST','TFIDF-W2V','0.34','0.36'],
                 ['Logistic Regression','TFIDF','0.40','0.41'],
                 ['Linear SVM','TFIDF','0.43','0.44'],
                 ['XGBOOST','TFIDF','0.33','0.35'],
                ]

         print(tabletext.to_text(data))
```

```
 Model                Vectorizer  Tr Los  Ts Los

 Random               -           -       0.88

 Logistic Regression  TFIDF-W2V   0.44    0.44

 Linear SVM           TFIDF-W2V   0.47    0.48

 XGBOOST              TFIDF-W2V   0.34    0.36

 Logistic Regression  TFIDF       0.40    0.41

 Linear SVM           TFIDF       0.43    0.44

 XGBOOST              TFIDF       0.33    0.35
```