Dawood Sarfraz

Duplicate Questions using KNN, Decision Tree, MLP, RandomForestClassifier and XGBoostClassifier with Advance Features

Dataset Description

The ground truth is the set of labels that have been supplied by human experts. The ground truth labels are inherently subjective, as the true meaning of sentences can never be known with certainty. Human labeling is also a 'noisy' process, and reasonable people will disagree. As a result, the ground truth labels on this dataset should be taken to be 'informed' but not 100% accurate, and may include incorrect labeling. We believe the labels, on the whole, to represent a reasonable consensus, but this may often not be true on a case by case basis for individual items in the dataset.

Please note: All of the questions in the training set are genuine examples from Quora.

Data fields

*** id -** the id of a training set question pair
*** qid1, qid2 -** - unique ids of each question (only available in train.csv)
*** question1, question2 -** - the full text of each question
*** is_duplicate -** the target variable, set to 1 if question1 and question2 have essentially the same meaning, and 0 otherwise.

Without Feature Engineering

In [10]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

import re
from bs4 import BeautifulSoup

import warnings
warnings.filterwarnings('ignore')
```

In [11]:

```
df = pd.read_csv("train.csv")
```

In [12]:

```
df.shape
```

Out[12]:

```
(404290, 6)
```

In [13]:

```
df.head(5)
```

Out[13]:

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

In [14]:

```
df.tail(5)
```

Out[14]:

	id	qid1	qid2	question1	question2	is_duplicate
404285	404285	433578	379845	How many keywords are there in the Racket prog...	How many keywords are there in PERL Programmin...	0
404286	404286	18840	155606	Do you believe there is life after death?	Is it true that there is life after death?	1
404287	404287	537928	537929	What is one coin?	What's this coin?	0
404288	404288	537930	537931	What is the approx annual cost of living while...	I am having little hairfall problem but I want...	0
404289	404289	537932	537933	What is like to have sex with cousin?	What is it like to have sex with your cousin?	0

In [15]:

```
df.sample(5)
```

Out[15]:

	id	qid1	qid2	question1	question2	is_duplicate
132233	132233	211804	22372	Why PM Narendra Modi scrapped Rs 500, Rs 1000 ...	Who suggested Narendra Modi to stop the circul...	1
270019	270019	109863	197612	I'm interested in the stock market. Where shou...	What are the best ways to invest money?	0
106506	106506	175427	175428	How important for AAP is to win Punjab Assembl...	How strong is AAP in Punjab?	0
209673	209673	314021	9518	How can I get more followers in fb?	How do I get more followers on Instagram?	0
339899	339899	111655	287598	What is a good solar panel installation provid...	What is a good solar panel installation provid...	0

In [16]:

```
df.info()
```

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

In [17]:

```
df.isnull().sum()
```

Out[17]:

```
id              0
qid1            0
qid2            0
question1       1
question2       2
is_duplicate    0
dtype: int64
```

In [18]:

```
df = df.dropna()
```

In [19]:

```
df.shape
```

Out[19]:

```
(404287, 6)
```

In [20]:

```
df.isnull().sum()
```

Out[20]:

```
id              0
qid1            0
qid2            0
question1       0
question2       0
is_duplicate    0
dtype: int64
```

In [21]:

```
df.duplicated().sum()
```

Out[21]:

```
0
```

In [22]:

```
print(df["is_duplicate"].value_counts())
print((df["is_duplicate"].value_counts()/df["is_duplicate"].count())*100)
```

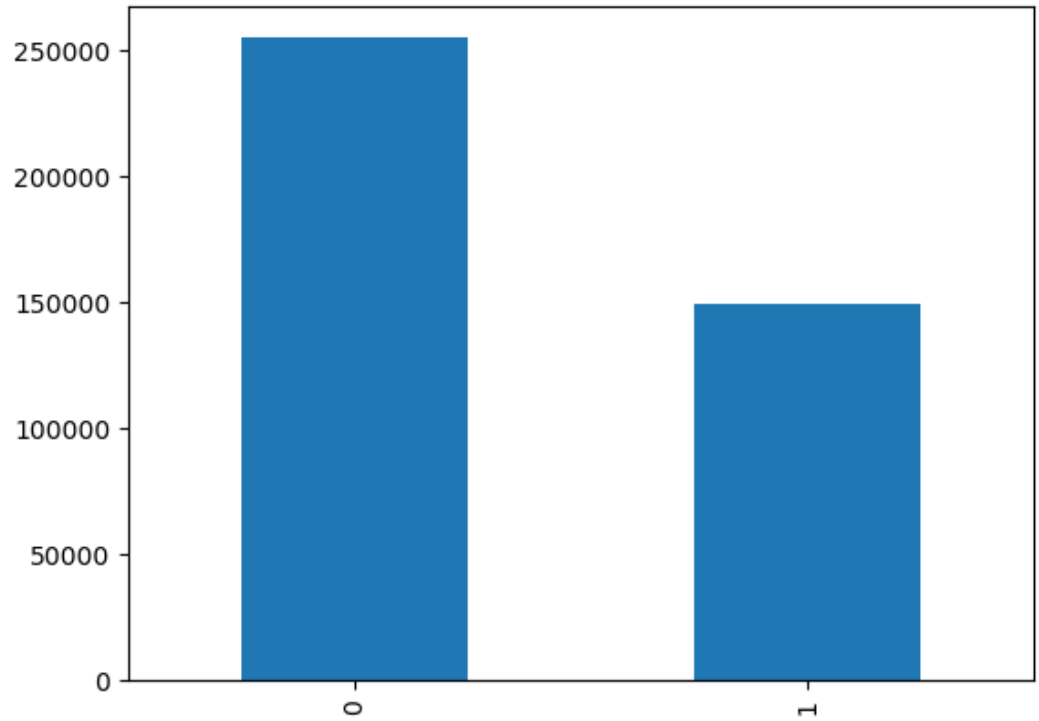
```
0    255024
1    149263
Name: is_duplicate, dtype: int64
0    63.079941
1    36.920059
Name: is_duplicate, dtype: float64
```

In [23]:

```
df["is_duplicate"].value_counts().plot(kind="bar")
```

Out[23]:

```
<Axes: >
```

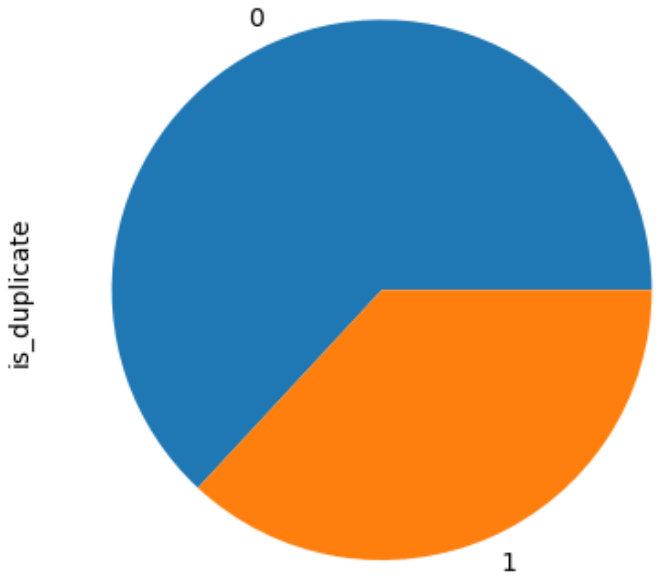


In [24]:

```
df["is_duplicate"].value_counts().plot(kind="pie")
```

Out[24]:

```
<Axes: ylabel='is_duplicate'>
```



In [25]:

```
qid = pd.Series(df["qid1"].tolist() + df["qid2"].tolist())
print("# of Unique Questions",np.unique(qid).shape[0])
```

```
# of Unique Questions 537929
```

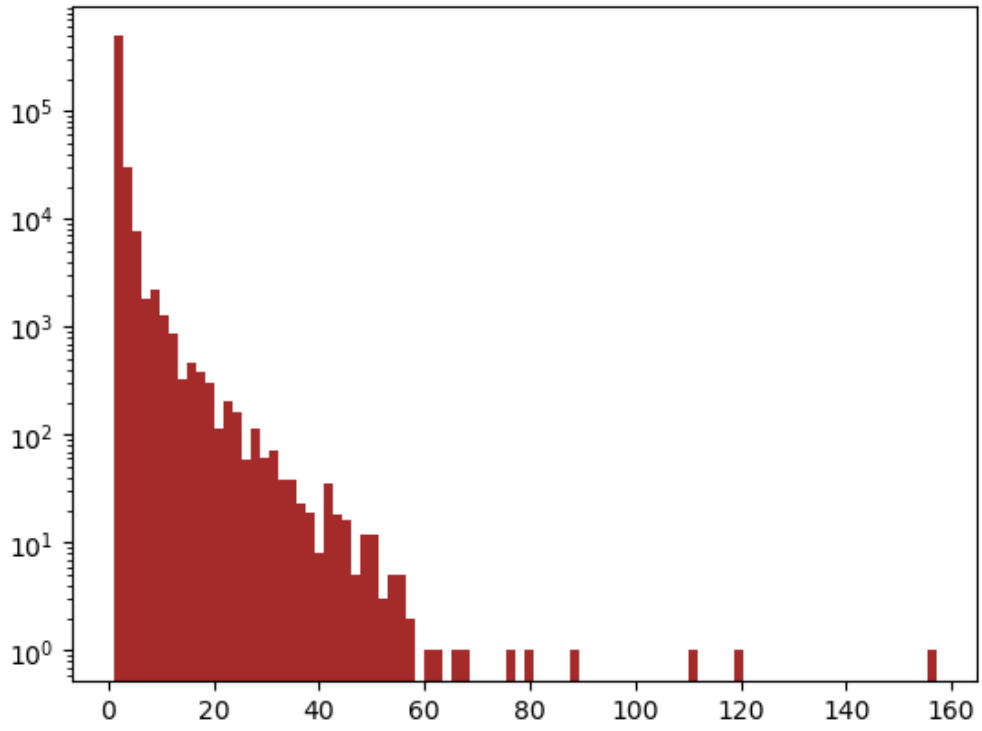
In [26]:

```
x = qid.value_counts()>1
print("# of Questions Qepeated",x[x].shape[0])
```

```
# of Questions Qepeated 111778
```

In [27]:

```
plt.hist(qid.value_counts().values,bins=90,color="brown")
plt.yscale("log")
plt.show()
```



In [28]:

```
new_df = df
```

In [29]:

```
new_df.shape
```

Out[29]:

```
(404287, 6)
```

In [30]:

```
new_df.isnull().sum()
```

Out[30]:

```
id              0
qid1            0
qid2            0
question1       0
question2       0
is_duplicate    0
dtype: int64
```

In [31]:

```
new_df = new_df.dropna()
```

In [32]:

```
new_df = df.sample(30000)
```

In [33]:

```
new_df.shape
```

Out[33]:

```
(30000, 6)
```

In [34]:

```
new_df.isnull().sum()
```

Out[34]:

```
id              0
qid1            0
qid2            0
question1       0
question2       0
is_duplicate    0
dtype: int64
```

In [35]:

```
new_df.duplicated().sum()
```

Out[35]:

```
0
```

In [36]:

```
new_df.shape
```

Out[36]:

```
(30000, 6)
```

In [37]:

```
ques_df = new_df[['question1','question2']]
ques_df.head()
```

Out[37]:

	question1	question2
181917	Why does religion matter?	What is a spiritual way of life?
280639	How do I get my "groundbreaking" app idea noti...	How do I get funding for my startup idea befor...
364988	What is Rob Halford’s vocal range?	What is Rob Halford's vocal range?
368254	How can I know my wife is not cheating?	What should I do knowing as a fact that my wif...
83348	Is Donald Trump's wife a U.S citizen?	What does Vladimir Putin think about the possi...

In [38]:

```
from sklearn.feature_extraction.text import CountVectorizer
# merge texts of questions asked
questions = list(ques_df['question1']) + list(ques_df['question2'])

# if You have Good laptop increase No. of max_features
cv = CountVectorizer(max_features=3000)#creating 3000 here for Question1 7 3000 for Questio
q1_array, q2_array = np.vsplit(cv.fit_transform(questions).toarray(),2)
```

In [39]:

```
temp_data1 = pd.DataFrame(q1_array, index= ques_df.index) # here q1_array back to data fram
temp_data2 = pd.DataFrame(q2_array, index= ques_df.index) # here q2_array back to data fram
temp_data = pd.concat([temp_data1, temp_data2], axis=1) # concating data frames here to mak
temp_data.shape
```

Out[39]:

```
(30000, 6000)
```

In [40]:

```
temp_data['is_duplicate'] = new_df['is_duplicate']
```

In [41]:

```
temp_data.shape
```

Out[41]:

```
(30000, 6001)
```

In [42]:

```
temp_data.head(5)
```

Out[42]:

	0	1	2	3	4	5	6	7	8	9	...	2991	2992	2993	2994	2995	2996	2997	2998	2999	is_duplic
181917	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
280639	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
364988	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
368254	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
83348	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	

5 rows × 6001 columns

In [43]:

```
temp_data.tail(5)
```

Out[43]:

	0	1	2	3	4	5	6	7	8	9	...	2991	2992	2993	2994	2995	2996	2997	2998	2999	is_duplic
139431	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
253042	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
276820	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
165142	0	0	0	0	0	0	0	0	0	0	...	0	0	0	1	0	0	0	0	0	
369318	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	

5 rows × 6001 columns

In [44]:

```
temp_data.sample(5)
```

Out[44]:

	0	1	2	3	4	5	6	7	8	9	...	2991	2992	2993	2994	2995	2996	2997	2998	2999	is_duplic
98683	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
335741	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
100469	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
130737	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
390371	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	0	0	

5 rows × 6001 columns

In [45]:

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
X_train, X_test, y_train, y_test = train_test_split(temp_data.iloc[:,0:-1].values, temp_data
                                                    test_size=0.2,random_state= 42)
```

In [46]:

```
from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier()
rfc.fit(X_train, y_train)
y_pred = rfc.predict(X_test)
accuracy = accuracy_score(y_test,y_pred) * 100
print("Accuracy of Random Forest",accuracy)
```

```
Accuracy of Random Forest 73.86666666666667
```

In [47]:

```
from xgboost import XGBClassifier
xgb = XGBClassifier()
xgb.fit(X_train, y_train)
y_pred = xgb.predict(X_test)
accuracy = accuracy_score(y_test,y_pred) * 100
print("Accuracy of Random Forest",accuracy)
```

```
Accuracy of Random Forest 72.61666666666666
```

In [48]:

```
 '''from sklearn import svm

# Create an SVM classifier
svm_classifier = svm.SVC(kernel='linear')

# Train the model
svm_classifier.fit(X_train, y_train)

# Make predictions on the test set
y_pred = svm_classifier.predict(X_test)

# Evaluate the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
'''
```

Out[48]:

```
'from sklearn import svm\n\n# Create an SVM classifier\nsvm_classifier = svm.S
VC(kernel=\'linear\')\n\n# Train the model\nsvm_classifier.fit(X_train, y_trai
n)\n\n# Make predictions on the test set\ny_pred = svm_classifier.predict(X_te
st)\n\n# Evaluate the accuracy of the model\naccuracy = accuracy_score(y_test,
y_pred)\nprint("Accuracy:", accuracy)\n'
```

In [49]:

```
from sklearn.neural_network import MLPClassifier

# Create an MLP classifier
mlp = MLPClassifier(hidden_layer_sizes=(10, 10), max_iter=1000, random_state=42)

# Train the classifier
mlp.fit(X_train, y_train)

# Make predictions on the test set
y_pred = mlp.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
accuracy = accuracy  * 100
print("Accuracy:", accuracy )
```

```
Accuracy: 67.78333333333333
```

In [50]:

```
from sklearn.neighbors import KNeighborsClassifier

# Create a KNN classifier object
knn = KNeighborsClassifier(n_neighbors=3)

# Train the classifier
knn.fit(X_train, y_train)

# Make predictions on the test set
y_pred = knn.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred) * 100
print("Accuracy:", accuracy)
```

```
Accuracy: 65.96666666666667
```

In [51]:

```
from sklearn.tree import DecisionTreeClassifier

# Create a decision tree classifier
clf = DecisionTreeClassifier()

# Train the classifier
clf.fit(X_train, y_train)

# Make predictions on the testing data to model
y_pred = clf.predict(X_test)

# Calculate the accuracy of the classifier
accuracy = accuracy_score(y_test, y_pred) * 100
print("Accuracy:", accuracy)
```

```
Accuracy: 66.63333333333334
```

In [ ]:

```

```

In [ ]:

```

```

Advance Feature Engineering

In [52]:

```
df =pd.read_csv("train.csv")
```

In [53]:

```
new_df = df.sample(20000,random_state=2)
```

In [54]:

```
new_df.shape
```

Out[54]:

```
(20000, 6)
```

In [55]:

```
new_df.head(5)
```

Out[55]:

	id	qid1	qid2	question1	question2	is_duplicate
398782	398782	496695	532029	What is the best marketing automation tool for...	What is the best marketing automation tool for...	1
115086	115086	187729	187730	I am poor but I want to invest. What should I do?	I am quite poor and I want to be very rich. Wh...	0
327711	327711	454161	454162	I am from India and live abroad. I met a guy f...	T.I.E.T to Thapar University to Thapar Univers...	0
367788	367788	498109	491396	Why do so many people in the U.S. hate the sou...	My boyfriend doesnt feel guilty when he hurts ...	0
151235	151235	237843	50930	Consequences of Bhopal gas tragedy?	What was the reason behind the Bhopal gas trag...	0

In [56]:

```
new_df.tail(5)
```

Out[56]:

	id	qid1	qid2	question1	question2	is_duplicate
36225	36225	66076	66077	Can I find out who is sending my public snap s...	Is Apple still an innovative, strong competito...	0
333984	333984	461132	461133	What do you mean by business world?	What is business world?	1
289725	289725	410890	410891	How we can use waste plastics bags and bottles...	How we can use waste plastic bags and bottles ...	1
342679	342679	470689	470690	Are hackathons a good place to find technical ...	I have no background in programming but have a...	0
245585	245585	358455	358456	Is it safe to visit srinagar in September 2016?	Is it safe to visit Srinagar in this coming Se...	1

In [ ]:

```

```

In [57]:

```
def questions_preprocessing(question):

    question = str(question).lower().strip()

    # Decontracting words
    contractions = {
    "ain't": "am not",
    "aren't": "are not",
    "can't": "can not",
    "can't've": "can not have",
    "'cause": "because",
    "could've": "could have",
    "couldn't": "could not",
    "couldn't've": "could not have",
    "didn't": "did not",
    "doesn't": "does not",
    "don't": "do not",
    "hadn't": "had not",
    "hadn't've": "had not have",
    "hasn't": "has not",
    "haven't": "have not",
    "he'd": "he would",
    "he'd've": "he would have",
    "he'll": "he will",
    "he'll've": "he will have",
    "he's": "he is",
    "how'd": "how did",
    "how'd'y": "how do you",
    "how'll": "how will",
    "how's": "how is",
    "i'd": "i would",
    "i'd've": "i would have",
    "i'll": "i will",
    "i'll've": "i will have",
    "i'm": "i am",
    "i've": "i have",
    "isn't": "is not",
    "it'd": "it would",
    "it'd've": "it would have",
    "it'll": "it will",
    "it'll've": "it will have",
    "it's": "it is",
    "let's": "let us",
    "ma'am": "madam",
    "mayn't": "may not",
    "might've": "might have",
    "mightn't": "might not",
    "mightn't've": "might not have",
    "must've": "must have",
    "mustn't": "must not",
    "mustn't've": "must not have",
    "needn't": "need not",
    "needn't've": "need not have",
    "o'clock": "of the clock",
    "oughtn't": "ought not",
    "oughtn't've": "ought not have",
    "shan't": "shall not",
    "sha'n't": "shall not",
    "shan't've": "shall not have",
    "she'd": "she would",
    "she'd've": "she would have",
    "she'll": "she will",
    "she'll've": "she will have",
    "she's": "she is",
    "should've": "should have",
```

```
    "shouldn't": "should not",
    "shouldn't've": "should not have",
    "so've": "so have",
    "so's": "so as",
    "that'd": "that would",
    "that'd've": "that would have",
    "that's": "that is",
    "there'd": "there would",
    "there'd've": "there would have",
    "there's": "there is",
    "they'd": "they would",
    "they'd've": "they would have",
    "they'll": "they will",
    "they'll've": "they will have",
    "they're": "they are",
    "they've": "they have",
    "to've": "to have",
    "wasn't": "was not",
    "we'd": "we would",
    "we'd've": "we would have",
    "we'll": "we will",
    "we'll've": "we will have",
    "we're": "we are",
    "we've": "we have",
    "weren't": "were not",
    "what'll": "what will",
    "what'll've": "what will have",
    "what're": "what are",
    "what's": "what is",
    "what've": "what have",
    "when's": "when is",
    "when've": "when have",
    "where'd": "where did",
    "where's": "where is",
    "where've": "where have",
    "who'll": "who will",
    "who'll've": "who will have",
    "who's": "who is",
    "who've": "who have",
    "why's": "why is",
    "why've": "why have",
    "will've": "will have",
    "won't": "will not",
    "won't've": "will not have",
    "would've": "would have",
    "wouldn't": "would not",
    "wouldn't've": "would not have",
    "y'all": "you all",
    "y'all'd": "you all would",
    "y'all'd've": "you all would have",
    "y'all're": "you all are",
    "y'all've": "you all have",
    "you'd": "you would",
    "you'd've": "you would have",
    "you'll": "you will",
    "you'll've": "you will have",
    "you're": "you are",
    "you've": "you have",
    "'ve": " have",
    "n't": " not",
    "'re": " are",
    "'ll": " will"
    }

    question_decontracted = []

    for word in question.split():
```

```
        if word in contractions:
            word = contractions[word]

        question_decontracted.append(word)

    question = ' '.join(question_decontracted)


    # Replace certain special characters with their string equivalents
    question = question.replace('%', ' percent')
    question = question.replace('$', ' dollar ')
    question = question.replace('₹', ' rupee ')
    question = question.replace('€', ' euro ')
    question = question.replace('@', ' at ')
    question = question.replace('R$',  'Brazilian Real')
    question = question.replace('S$',  'Singapore Dollar')
    question = question.replace('NZ$', 'New Zealand Dollar')
    question = question.replace('HK$', 'Hong Kong Dollar')
    question = question.replace('₩',   'South Korean Won')
    question = question.replace('₺', 'Turkish Lira')
    question = question.replace('₽',  'Russian Ruble')
    question = question.replace('zł',  'Polish Zloty')
    question = question.replace('Kč',  'Czech Koruna')
    question = question.replace('₪',   'Israeli Shekel')
    question = question.replace('¥',   'Chinese Yuan')
    question = question.replace('₣',   'Swiss Franc')


    # The pattern '[math]' appears around 900 times in the whole dataset.
    question = question.replace('[math]', '')

    # Replacing some numbers with string equivalents (not perfect, can be done better to ac
    question = question.replace(',000,000,000 ', 'b ')
    question = question.replace(',000,000 ', 'm ')
    question = question.replace(',000 ', 'k ')

    # re is regualar Experssion
    question = re.sub(r'([0-9]+)000000000', r'\1b', question)
    question = re.sub(r'([0-9]+)000000', r'\1m', question)
    question = re.sub(r'([0-9]+)000', r'\1k', question)
    # Removing HTML tags
    question = BeautifulSoup(question)
    question = question.get_text()

    # Remove punctuations
    pattern = re.compile('\W')
    question = re.sub(pattern, ' ', question).strip()


    return question
```

In [58]:

```
questions_preprocessing("That's Great <b>done</b>?")
```

Out[58]:

```
'that is great done'
```

In [59]:

```
new_df['question1'] = new_df['question1'].apply(questions_preprocessing)
new_df['question2'] = new_df['question2'].apply(questions_preprocessing)
```

In [60]:

```
new_df.shape
```

Out[60]:

```
(20000, 6)
```

In [61]:

```
new_df.sample(5)
```

Out[61]:

	id	qid1	qid2	question1	question2	is_duplicate
71195	71195	122562	42155	what are the options for an average student af...	what is best career plan after completing grad...	0
40425	40425	73146	73147	how can i write seo content fast	which is a great seo content generator	1
392669	392669	525376	525377	what does seeing the number 44 everywhere mean	what does it mean to see the number 44 everywhere	1
300164	300164	44282	110047	what is the last thing you would like to do be...	what is the one thing that you want to do befo...	1
355006	355006	389915	484202	what are the best ways to do marketing in real...	how do i analyze real estate markets	0

In [62]:

```
new_df.head(5)
```

Out[62]:

	id	qid1	qid2	question1	question2	is_duplicate
398782	398782	496695	532029	what is the best marketing automation tool for...	what is the best marketing automation tool for...	1
115086	115086	187729	187730	i am poor but i want to invest what should i do	i am quite poor and i want to be very rich wh...	0
327711	327711	454161	454162	i am from india and live abroad i met a guy f...	t i e t to thapar university to thapar univers...	0
367788	367788	498109	491396	why do so many people in the u s hate the sou...	my boyfriend doesnt feel guilty when he hurts ...	0
151235	151235	237843	50930	consequences of bhopal gas tragedy	what was the reason behind the bhopal gas tragedy	0

In [63]:

```
new_df.tail(5)
```

Out[63]:

	id	qid1	qid2	question1	question2	is_duplicate
36225	36225	66076	66077	can i find out who is sending my public snap s...	is apple still an innovative strong competito...	0
333984	333984	461132	461133	what do you mean by business world	what is business world	1
289725	289725	410890	410891	how we can use waste plastics bags and bottles...	how we can use waste plastic bags and bottles ...	1
342679	342679	470689	470690	are hackathons a good place to find technical ...	i have no background in programming but have a...	0
245585	245585	358455	358456	is it safe to visit srinagar in september 2016	is it safe to visit srinagar in this coming se...	1

In [64]:

```
new_df['chars_in_q1'] = new_df['question1'].str.len()
new_df['chars_in_q2'] = new_df['question2'].str.len()
```

In [65]:

```
new_df.shape
```

Out[65]:

```
(20000, 8)
```

In [66]:

```
new_df.sample(5)
```

Out[66]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2
97089	97089	161610	161611	why do people feel pride	why do people feel shame	0	24	24
232060	232060	342009	342010	can a person suffered by sickle cell disease b...	can a person with myopia disease be a ias officer	0	92	49
389542	389542	522031	1772	how can i increase my height fast	how can you increase your height	0	33	32
216063	216063	322109	322110	how do i describe a manifold such as a cylinde...	what is flat manifold and non flat manifold	0	78	43
229706	229706	339079	339080	how did alan rickman die	what alan rickman movie did you like the most	0	24	45

In [67]:

```
new_df.head(5)
```

Out[67]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2
398782	398782	496695	532029	what is the best marketing automation tool for...	what is the best marketing automation tool for...	1	75	76
115086	115086	187729	187730	i am poor but i want to invest what should i do	i am quite poor and i want to be very rich wh...	0	48	56
327711	327711	454161	454162	i am from india and live abroad i met a guy f...	t i e t to thapar university to thapar univers...	0	104	119
367788	367788	498109	491396	why do so many people in the u s hate the sou...	my boyfriend doesnt feel guilty when he hurts ...	0	58	145
151235	151235	237843	50930	consequences of bhopal gas tragedy	what was the reason behind the bhopal gas tragedy	0	34	49

In [68]:

```
new_df.tail(5)
```

Out[68]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2
36225	36225	66076	66077	can i find out who is sending my public snap s...	is apple still an innovative strong competito...	0	137	112
333984	333984	461132	461133	what do you mean by business world	what is business world	1	34	22
289725	289725	410890	410891	how we can use waste plastics bags and bottles...	how we can use waste plastic bags and bottles ...	1	62	65
342679	342679	470689	470690	are hackathons a good place to find technical ...	i have no background in programming but have a...	0	143	140
245585	245585	358455	358456	is it safe to visit srinagar in september 2016	is it safe to visit srinagar in this coming se...	1	46	53

In [ ]:

```

```

In [69]:

```
new_df['words_no_words_in_q1'] = new_df['question1'].apply(lambda row: len(row.split(" ")))
new_df['words_no_words_in_q2'] = new_df['question2'].apply(lambda row: len(row.split(" ")))
```

In [70]:

```
new_df.shape
```

Out[70]:

```
(20000, 10)
```

In [71]:

```
new_df.head(5)
```

Out[71]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_r
398782	398782	496695	532029	what is the best marketing automation tool for...	what is the best marketing automation tool for...	1	75	76	
115086	115086	187729	187730	i am poor but i want to invest what should i do	i am quite poor and i want to be very rich wh...	0	48	56	
327711	327711	454161	454162	i am from india and live abroad i met a guy f...	t i e t to thapar university to thapar univers...	0	104	119	
367788	367788	498109	491396	why do so many people in the u s hate the sou...	my boyfriend doesnt feel guilty when he hurts ...	0	58	145	
151235	151235	237843	50930	consequences of bhopal gas tragedy	what was the reason behind the bhopal gas tragedy	0	34	49	

In [72]:

```
new_df.tail(5)
```

Out[72]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_n
36225	36225	66076	66077	can i find out who is sending my public snap s...	is apple still an innovative strong competito...	0	137	112	
333984	333984	461132	461133	what do you mean by business world	what is business world	1	34	22	
289725	289725	410890	410891	how we can use waste plastics bags and bottles...	how we can use waste plastic bags and bottles ...	1	62	65	
342679	342679	470689	470690	are hackathons a good place to find technical ...	i have no background in programming but have a...	0	143	140	
245585	245585	358455	358456	is it safe to visit srinagar in september 2016	is it safe to visit srinagar in this coming se...	1	46	53	

In [73]:

```
new_df.tail(5)
```

Out[73]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_n
36225	36225	66076	66077	can i find out who is sending my public snap s...	is apple still an innovative strong competito...	0	137	112	
333984	333984	461132	461133	what do you mean by business world	what is business world	1	34	22	
289725	289725	410890	410891	how we can use waste plastics bags and bottles...	how we can use waste plastic bags and bottles ...	1	62	65	
342679	342679	470689	470690	are hackathons a good place to find technical ...	i have no background in programming but have a...	0	143	140	
245585	245585	358455	358456	is it safe to visit srinagar in september 2016	is it safe to visit srinagar in this coming se...	1	46	53	

In [74]:

```
def common_words_in_questions(row):
    word1 = set(map(lambda word: word.lower().strip(), row['question1'].split(" ")))
    word2 = set(map(lambda word: word.lower().strip(), row['question2'].split(" ")))
    length = len(word1 & word2)
    return length
```

In [75]:

```
new_df['common_words_qs'] = new_df.apply(common_words_in_questions, axis=1)
```

In [76]:

```
new_df.shape
```

Out[76]:

```
(20000, 11)
```

In [77]:

```
new_df.head(5)
```

Out[77]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_i
398782	398782	496695	532029	what is the best marketing automation tool for...	what is the best marketing automation tool for...	1	75	76	
115086	115086	187729	187730	i am poor but i want to invest what should i do	i am quite poor and i want to be very rich wh...	0	48	56	
327711	327711	454161	454162	i am from india and live abroad i met a guy f...	t i e t to thapar university to thapar univers...	0	104	119	
367788	367788	498109	491396	why do so many people in the u s hate the sou...	my boyfriend doesnt feel guilty when he hurts ...	0	58	145	
151235	151235	237843	50930	consequences of bhopal gas tragedy	what was the reason behind the bhopal gas tragedy	0	34	49	

In [78]:

```
new_df.sample(5)
```

Out[78]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_no
29814	29814	55122	55123	what was the age of rama and sita when they go...	i am a muslim female and i want to marry a hin...	0	55	94	
136098	136098	54919	157084	what is the best book about digital marketing	which are the best books on digital marketing	1	45	45	
170592	170592	263739	138872	what field of engineering pays the highest	which field of engineering gives the highest s...	1	42	51	
339989	339989	143283	467685	what function does the nose has in the respira...	what is the function of the respiratory system...	0	57	60	
162692	162692	253229	253230	will missing mothers surname in certificates c...	age barrier for commercial pilot training in usa	0	111	48	

In [79]:

```
new_df.tail(5)
```

Out[79]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_n
36225	36225	66076	66077	can i find out who is sending my public snap s...	is apple still an innovative strong competito...	0	137	112	
333984	333984	461132	461133	what do you mean by business world	what is business world	1	34	22	
289725	289725	410890	410891	how we can use waste plastics bags and bottles...	how we can use waste plastic bags and bottles ...	1	62	65	
342679	342679	470689	470690	are hackathons a good place to find technical ...	i have no background in programming but have a...	0	143	140	
245585	245585	358455	358456	is it safe to visit srinagar in september 2016	is it safe to visit srinagar in this coming se...	1	46	53	

In [80]:

```
def total_words_in_questions(row):
    word1 = set(map(lambda word: word.lower().strip(), row['question1'].split(" ")))
    word2 = set(map(lambda word: word.lower().strip(), row['question2'].split(" ")))
    length = (len(word1) + len(word2))
    return length
```

In [81]:

```
new_df['total_words_in_questions'] = new_df.apply(total_words_in_questions, axis=1)
```

In [82]:

```
new_df.shape
```

Out[82]:

```
(20000, 12)
```

In [83]:

```
new_df.sample(5)
```

Out[83]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_no_v
186976	186976	203765	285078	what are the novels you can suggest that are w...	can you suggest me some good novels to read	1	58	43	
248076	248076	184449	121560	do you like anime	why do you hate animals	0	17	23	
237037	237037	102177	206436	how bad is it to swallow bleach	what happens if you drink bleach	1	31	32	
112282	112282	22524	183742	what is the spirit of quora	what is the spirit	0	27	18	
355893	355893	244584	55038	what are the best ways to get rid of boredom	how do you get out of boredom	1	44	29	

In [84]:

```
new_df.head(5)
```

Out[84]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_r
398782	398782	496695	532029	what is the best marketing automation tool for...	what is the best marketing automation tool for...	1	75	76	
115086	115086	187729	187730	i am poor but i want to invest what should i do	i am quite poor and i want to be very rich wh...	0	48	56	
327711	327711	454161	454162	i am from india and live abroad i met a guy f...	t i e t to thapar university to thapar univers...	0	104	119	
367788	367788	498109	491396	why do so many people in the u s hate the sou...	my boyfriend doesnt feel guilty when he hurts ...	0	58	145	
151235	151235	237843	50930	consequences of bhopal gas tragedy	what was the reason behind the bhopal gas tragedy	0	34	49	

In [85]:

```
new_df.tail(5)
```

Out[85]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_n
36225	36225	66076	66077	can i find out who is sending my public snap s...	is apple still an innovative strong competito...	0	137	112	
333984	333984	461132	461133	what do you mean by business world	what is business world	1	34	22	
289725	289725	410890	410891	how we can use waste plastics bags and bottles...	how we can use waste plastic bags and bottles ...	1	62	65	
342679	342679	470689	470690	are hackathons a good place to find technical ...	i have no background in programming but have a...	0	143	140	
245585	245585	358455	358456	is it safe to visit srinagar in september 2016	is it safe to visit srinagar in this coming se...	1	46	53	

In [86]:

```
new_df['shared_words_in_questions'] = round(new_df['common_words_qs']/new_df['total_words_i
```

In [87]:

```
new_df.shape
```

Out[87]:

```
(20000, 13)
```

In [88]:

```
new_df.head(5)
```

Out[88]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_i
398782	398782	496695	532029	what is the best marketing automation tool for...	what is the best marketing automation tool for...	1	75	76	
115086	115086	187729	187730	i am poor but i want to invest what should i do	i am quite poor and i want to be very rich wh...	0	48	56	
327711	327711	454161	454162	i am from india and live abroad i met a guy f...	t i e t to thapar university to thapar univers...	0	104	119	
367788	367788	498109	491396	why do so many people in the u s hate the sou...	my boyfriend doesnt feel guilty when he hurts ...	0	58	145	
151235	151235	237843	50930	consequences of bhopal gas tragedy	what was the reason behind the bhopal gas tragedy	0	34	49	

In [89]:

```
new_df.tail(5)
```

Out[89]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_n
36225	36225	66076	66077	can i find out who is sending my public snap s...	is apple still an innovative strong competito...	0	137	112	
333984	333984	461132	461133	what do you mean by business world	what is business world	1	34	22	
289725	289725	410890	410891	how we can use waste plastics bags and bottles...	how we can use waste plastic bags and bottles ...	1	62	65	
342679	342679	470689	470690	are hackathons a good place to find technical ...	i have no background in programming but have a...	0	143	140	
245585	245585	358455	358456	is it safe to visit srinagar in september 2016	is it safe to visit srinagar in this coming se...	1	46	53	

In [90]:

```
new_df.sample(5)
```

Out[90]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_i
257584	257584	372932	372933	can you tell me about myself just by reading t...	can you tell me something about myself just by...	0	58	68	
150742	150742	237175	237176	is mississippi or alabama the most racist stat...	why is mississippi the most backward state bo...	0	57	76	
319307	319307	444725	444726	what is a sensible methodology benchmark to pr...	what is a sensible methodology benchmark to pr...	1	135	143	
306972	306972	430590	430591	what are some examples of chess players collud...	if jamaica had been a white colony with a mino...	0	89	114	
315387	315387	440275	440276	why do most people only think of karma as futu...	what are some of deep learning models that can...	0	145	110	

In [91]:

```
# Advanced Feature adding
from nltk.corpus import stopwords

def token_features_fetching_from_questions(row):

    question1 = row['question1']
    question2 = row['question2']

    SAFE_DIV = 0.0000001

    STOP_WORDS = stopwords.words("english")

    token_features = [0.0]*8 # bcz of 8 features 0-7

    # Converting the Sentence into Tokens:
    question1_tokens = question1.split()
    question2_tokens = question2.split()

    if len(question1_tokens) == 0 or len(question2_tokens) == 0:
        return token_features

    # Get the non-stopwords in Questions
    question1_words = set([word for word in question1_tokens if word not in STOP_WORDS])
    question2_words = set([word for word in question2_tokens if word not in STOP_WORDS])

    #Get the stopwords in Questions
    question1_stops = set([word for word in question1_tokens if word in STOP_WORDS])
    question2_stops = set([word for word in question2_tokens if word in STOP_WORDS])

    # Get the common non-stopwords from Question pair
    common_word_count = len(question1_words.intersection(question2_words))

    # Get the common stopwords from Question pair
    common_stop_count = len(question1_stops.intersection(question2_stops))

    # Get the common Tokens from Question pair
    common_token_count = len(set(question1_tokens).intersection(set(question2_tokens)))

    token_features[0] = common_word_count / (min(len(question1_words), len(question2_words)
    token_features[1] = common_word_count / (max(len(question1_words), len(question2_words)
    token_features[2] = common_stop_count / (min(len(question1_stops), len(question2_stops)
    token_features[3] = common_stop_count / (max(len(question1_stops), len(question2_stops)
    token_features[4] = common_token_count / (min(len(question1_tokens), len(question2_toke
    token_features[5] = common_token_count / (max(len(question1_tokens), len(question2_toke

    # Last word of Q1 AND Q2 is SAME or NOT
    token_features[6] = int(question1_tokens[-1] == question2_tokens[-1])

    # First word of Q1 AND Q2 is AME or NOT
    token_features[7] = int(question1_tokens[0] == question2_tokens[0])

    return token_features
```

In [92]:

```
token_features = new_df.apply(token_features_fetching_from_questions, axis=1)

new_df["common_words_count_min"]       = list(map(lambda x: x[0], token_features))
new_df["common_words_count_max"]       = list(map(lambda x: x[1], token_features))
new_df["common_stopwords_count_min"]       = list(map(lambda x: x[2], token_features))
new_df["common_stopwords_count_max"]       = list(map(lambda x: x[3], token_features))
new_df["common_token_count_min"]       = list(map(lambda x: x[4], token_features))
new_df["common_token_count_max"]       = list(map(lambda x: x[5], token_features))
new_df["last_word_matching"]  = list(map(lambda x: x[6], token_features))
new_df["first_word_matching"] = list(map(lambda x: x[7], token_features))
```

In [93]:

```
new_df.shape
```

Out[93]:

```
(20000, 21)
```

In [94]:

```
new_df.sample(5)
```

Out[94]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	word
198950	198950	300282	300283	how can i log in to textplus on a computer	why does my computer keep logging off on its own	0	42	48	
382301	382301	514140	514141	why did sandor clegane leave command in the ba...	would stannis baratheon have won the battle of...	0	68	109	
393546	393546	526360	526361	what are three essential characteristics of go...	what are the characteristics of a good communi...	1	62	52	
135772	135772	216763	216764	is oh no ok go album available on spotify	is radiohead s new album on spotify yet	0	43	39	
2294	2294	4561	4562	what does honors means in graduation	i have trouble getting my claim amount form ap...	0	36	64	

5 rows × 21 columns

In [95]:

```
new_df.head(5)
```

Out[95]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_i
398782	398782	496695	532029	what is the best marketing automation tool for...	what is the best marketing automation tool for...	1	75	76	
115086	115086	187729	187730	i am poor but i want to invest what should i do	i am quite poor and i want to be very rich wh...	0	48	56	
327711	327711	454161	454162	i am from india and live abroad i met a guy f...	t i e t to thapar university to thapar univers...	0	104	119	
367788	367788	498109	491396	why do so many people in the u s hate the sou...	my boyfriend doesnt feel guilty when he hurts ...	0	58	145	
151235	151235	237843	50930	consequences of bhopal gas tragedy	what was the reason behind the bhopal gas tragedy	0	34	49	

5 rows × 21 columns

In [96]:

```
new_df.tail(5)
```

Out[96]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_n
36225	36225	66076	66077	can i find out who is sending my public snap s...	is apple still an innovative strong competito...	0	137	112	
333984	333984	461132	461133	what do you mean by business world	what is business world	1	34	22	
289725	289725	410890	410891	how we can use waste plastics bags and bottles...	how we can use waste plastic bags and bottles ...	1	62	65	
342679	342679	470689	470690	are hackathons a good place to find technical ...	i have no background in programming but have a...	0	143	140	
245585	245585	358455	358456	is it safe to visit srinagar in september 2016	is it safe to visit srinagar in this coming se...	1	46	53	

5 rows × 21 columns

In [97]:

```
import distance

def fetch_length_features(row):

    question1 = row['question1']
    question2 = row['question2']

    length_features = [0.0]*3

    # Converting the Sentence into Tokens:
    question1_tokens = question1.split()
    question2_tokens = question2.split()

    if len(question1_tokens) == 0 or len(question2_tokens) == 0:
        return length_features


    #Average Token Length of both Questions
    length_features[0] = (len(question1_tokens) + len(question2_tokens))/2

    # Absolute length features
    length_features[1] = abs(len(question1_tokens) - len(question2_tokens))

    strs = list(distance.lcsubstrings(question1, question2))
    length_features[2] = len(strs[0]) / (min(len(question1), len(question2)) + 1)

    return length_features
```

In [98]:

```
length_features = new_df.apply(fetch_length_features, axis=1)

new_df['average_length_of_question'] = list(map(lambda x: x[0], length_features))
new_df['absolute_difference_of_qs_length'] = list(map(lambda x: x[1], length_features))
new_df['longest_substring_ratio'] = list(map(lambda x: x[2], length_features))
```

In [99]:

```
new_df.shape
```

Out[99]:

```
(20000, 24)
```

In [100]:

```
new_df.head(5)
```

Out[100]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_i
398782	398782	496695	532029	what is the best marketing automation tool for...	what is the best marketing automation tool for...	1	75	76	
115086	115086	187729	187730	i am poor but i want to invest what should i do	i am quite poor and i want to be very rich wh...	0	48	56	
327711	327711	454161	454162	i am from india and live abroad i met a guy f...	t i e t to thapar university to thapar univers...	0	104	119	
367788	367788	498109	491396	why do so many people in the u s hate the sou...	my boyfriend doesnt feel guilty when he hurts ...	0	58	145	
151235	151235	237843	50930	consequences of bhopal gas tragedy	what was the reason behind the bhopal gas tragedy	0	34	49	

5 rows × 24 columns

In [101]:

```
new_df.sample(5)
```

Out[101]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_no_v
103882	103882	94517	171598	why should i become a doctor	what should i do if i can not become a doctor	0	28	45	
351268	351268	480097	480098	what is the iupac name for ch3_ch_ ch_ ch3 ch3...	what is the iupac name of ch3ch ch3 ccl c ch3...	0	50	57	
231307	231307	341081	341082	why was the cgi for supreme leader snoke so bad	why was supreme leader snoke made as a cgi cha...	0	47	113	
244302	244302	356908	356909	how many calories are in a serving of bread wi...	how many calories are considered a lot	0	100	38	
41638	41638	75156	75157	is it ever okay to slap a child in the face	is it okay to hit my child if he is over eighteen	0	43	49	

5 rows × 24 columns

In [102]:

```
new_df.tail(5)
```

Out[102]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_n
36225	36225	66076	66077	can i find out who is sending my public snap s...	is apple still an innovative strong competito...	0	137	112	
333984	333984	461132	461133	what do you mean by business world	what is business world	1	34	22	
289725	289725	410890	410891	how we can use waste plastics bags and bottles...	how we can use waste plastic bags and bottles ...	1	62	65	
342679	342679	470689	470690	are hackathons a good place to find technical ...	i have no background in programming but have a...	0	143	140	
245585	245585	358455	358456	is it safe to visit srinagar in september 2016	is it safe to visit srinagar in this coming se...	1	46	53	

5 rows × 24 columns

In [103]:

```
# Fuzzy Features
from fuzzywuzzy import fuzz

def fetch_fuzzy_features_from_questions(row):

    question1 = row['question1']
    question2 = row['question2']

    fuzzy_features = [0.0]*4


    # fuzz_partial_ratio btw question 1 and Question 2
    fuzzy_features[0] = fuzz.partial_ratio(question1, question2)

    # fuzz_ratio btw question 1 and Question 2
    fuzzy_features[1] = fuzz.QRatio(question1, question2)

    # token_sort_ratio btw question 1 and Question 2
    fuzzy_features[2] = fuzz.token_sort_ratio(question1, question2)

    # token_set_ratio btw question 1 and Question 2
    fuzzy_features[3] = fuzz.token_set_ratio(question1, question2)

    return fuzzy_features
```

In [104]:

```
fuzzy_features = new_df.apply(fetch_fuzzy_features_from_questions, axis=1)

# Creating new feature columns for fuzzy features in already given data set
new_df['fuzz_partial_ratio'] = list(map(lambda x: x[0], fuzzy_features))
new_df['fuzz_ratio'] = list(map(lambda x: x[1], fuzzy_features))
new_df['token_sort_ratio'] = list(map(lambda x: x[2], fuzzy_features))
new_df['token_set_ratio'] = list(map(lambda x: x[3], fuzzy_features))
```

In [108]:

```
new_df.shape
```

Out[108]:

```
(20000, 28)
```

In [109]:

```
new_df.head(5)
```

Out[109]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_i
398782	398782	496695	532029	what is the best marketing automation tool for...	what is the best marketing automation tool for...	1	75	76	
115086	115086	187729	187730	i am poor but i want to invest what should i do	i am quite poor and i want to be very rich wh...	0	48	56	
327711	327711	454161	454162	i am from india and live abroad i met a guy f...	t i e t to thapar university to thapar univers...	0	104	119	
367788	367788	498109	491396	why do so many people in the u s hate the sou...	my boyfriend doesnt feel guilty when he hurts ...	0	58	145	
151235	151235	237843	50930	consequences of bhopal gas tragedy	what was the reason behind the bhopal gas tragedy	0	34	49	

5 rows × 28 columns

In [110]:

```
new_df.sample(5)
```

Out[110]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_no_
360069	360069	63836	158795	what is the funniest movie to watch	what is the funniest movie that you have ever ...	1	35	53	
323593	323593	449573	449574	find the equation to the circle passin through...	consultancy job after engineering	0	112	33	
161269	161269	21200	113942	how do a junior high school students make mone...	how can a high school student make money	1	62	40	
164589	164589	62406	109956	what are prospects and challenges of pulses in...	what are the prospects for pulses for sustaina...	1	60	63	
278751	278751	175092	34326	what should be done to improve problem solving	how can i improve me problem solving skills	1	46	43	

5 rows × 28 columns

In [111]:

```
new_df.tail(5)
```

Out[111]:

	id	qid1	qid2	question1	question2	is_duplicate	chars_in_q1	chars_in_q2	words_n
36225	36225	66076	66077	can i find out who is sending my public snap s...	is apple still an innovative strong competito...	0	137	112	
333984	333984	461132	461133	what do you mean by business world	what is business world	1	34	22	
289725	289725	410890	410891	how we can use waste plastics bags and bottles...	how we can use waste plastic bags and bottles ...	1	62	65	
342679	342679	470689	470690	are hackathons a good place to find technical ...	i have no background in programming but have a...	0	143	140	
245585	245585	358455	358456	is it safe to visit srinagar in september 2016	is it safe to visit srinagar in this coming se...	1	46	53	

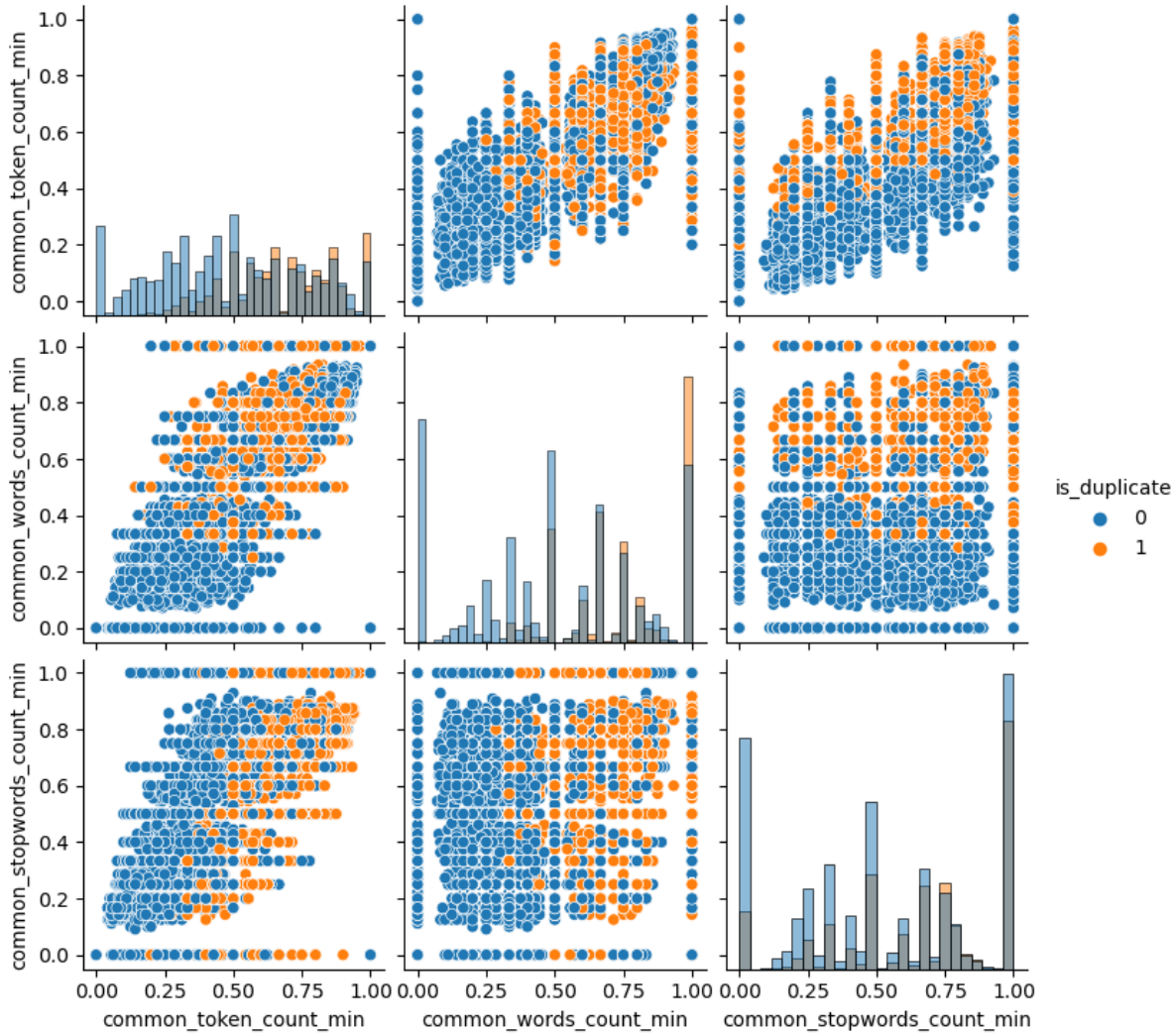
5 rows × 28 columns

In [112]:

```
sns.pairplot(new_df[['common_token_count_min', 'common_words_count_min', 'common_stopwords_
```

Out[112]:

```
<seaborn.axisgrid.PairGrid at 0x7f84fb4eca90>
```

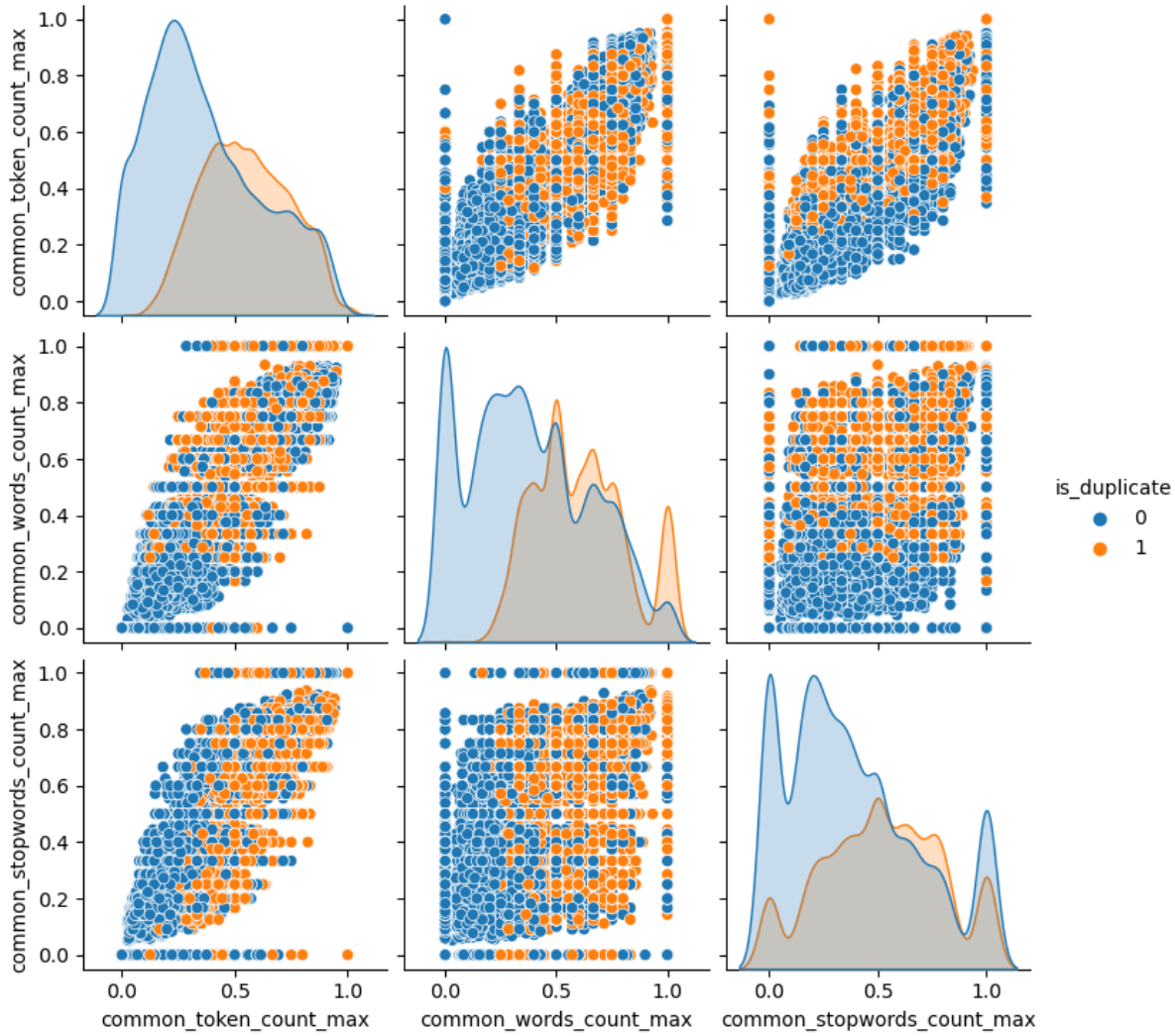


In [113]:

```
sns.pairplot(new_df[['common_token_count_max', 'common_words_count_max', 'common_stopwords_
```

Out[113]:

```
<seaborn.axisgrid.PairGrid at 0x7f8501d91a90>
```

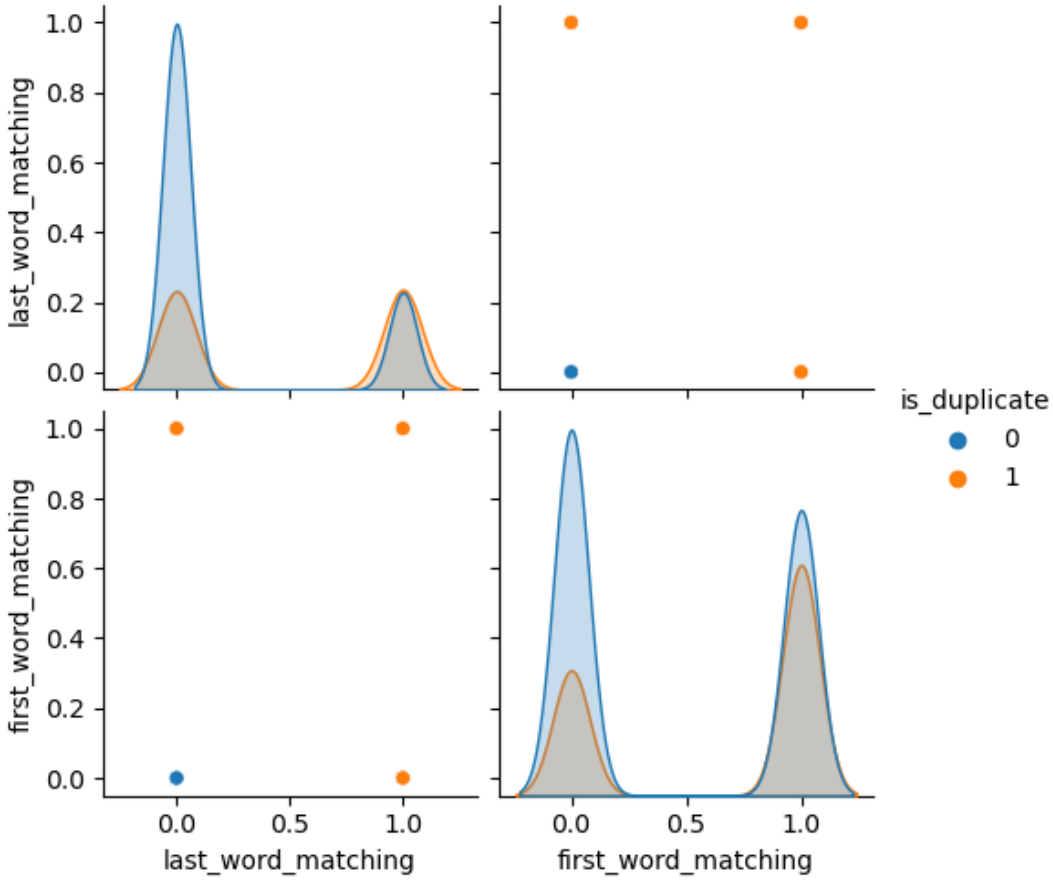


In [114]:

```
sns.pairplot(new_df[['last_word_matching', 'first_word_matching', 'is_duplicate']],hue='is_
```

Out[114]:

```
<seaborn.axisgrid.PairGrid at 0x7f838ace1a90>
```

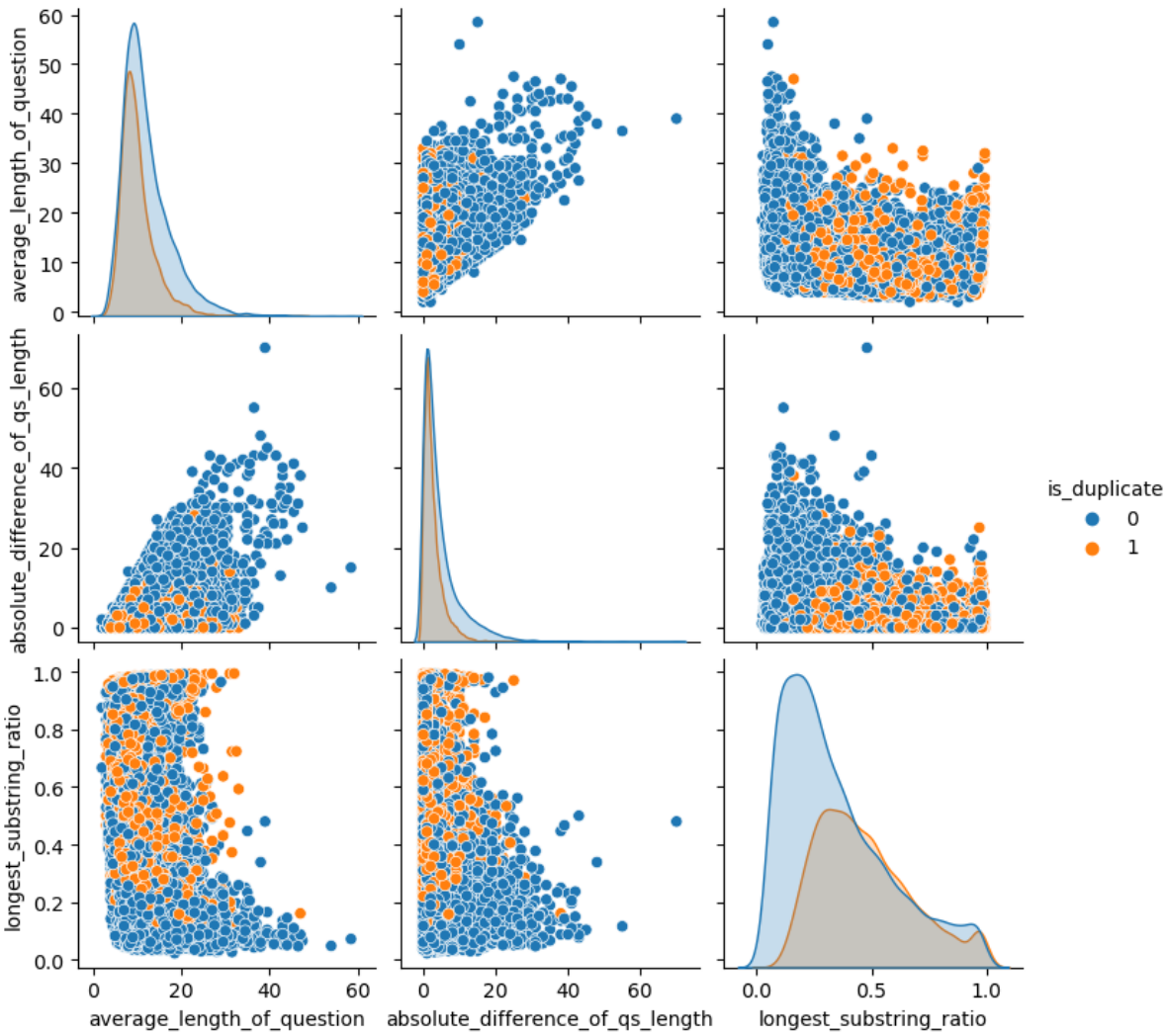


In [115]:

```
sns.pairplot(new_df[['average_length_of_question', 'absolute_difference_of_qs_length','longe
```

Out[115]:

```
<seaborn.axisgrid.PairGrid at 0x7f838a9e8450>
```

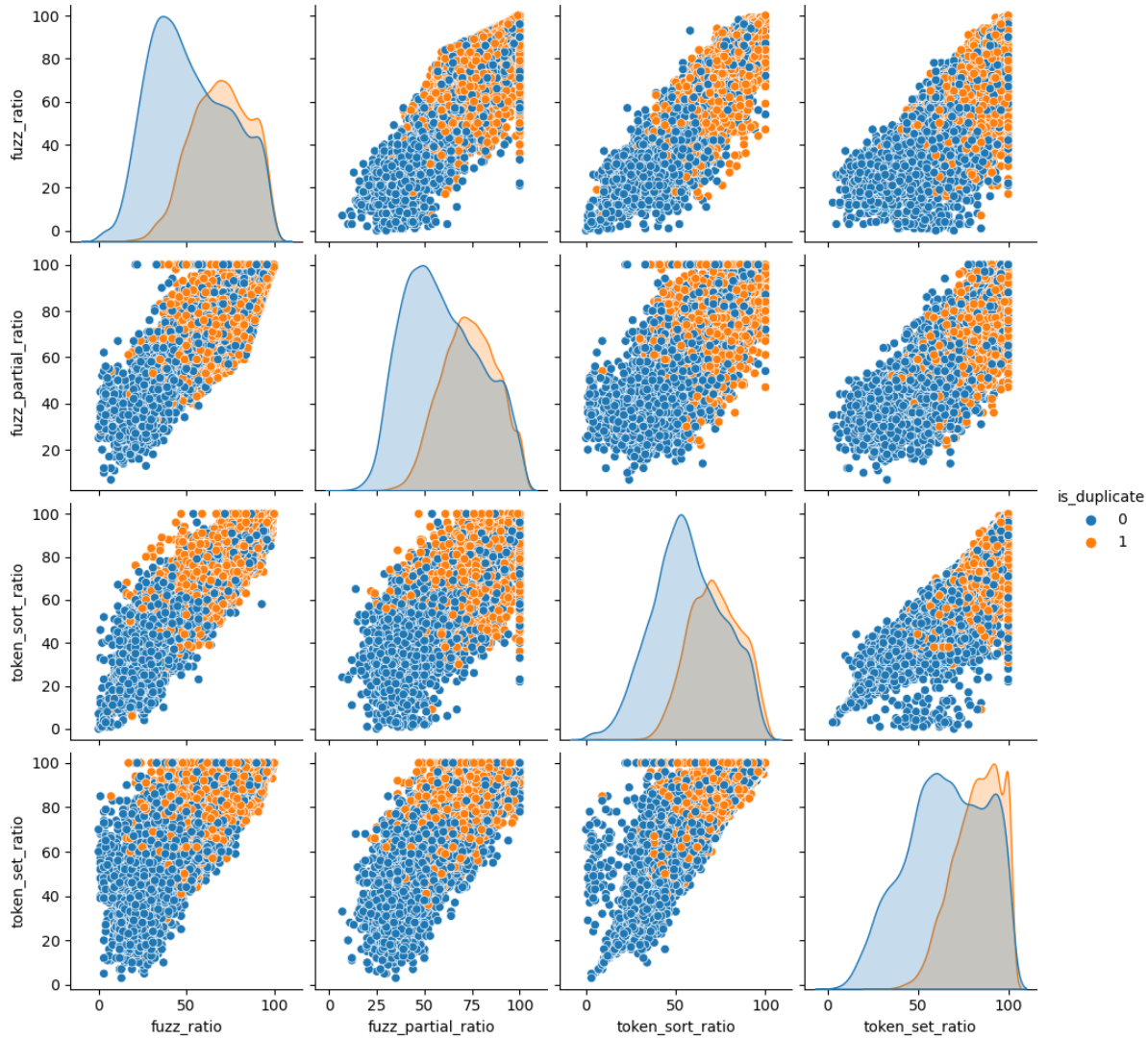


In [116]:

```
sns.pairplot(new_df[['fuzz_ratio', 'fuzz_partial_ratio','token_sort_ratio','token_set_ratio
```

Out[116]:

```
<seaborn.axisgrid.PairGrid at 0x7f838a406e90>
```



In [117]:

```
from sklearn.preprocessing import MinMaxScaler

X = MinMaxScaler().fit_transform(new_df[['common_stopwords_count_min', 'common_stopwords_co
y = new_df['is_duplicate'].values
```

In [118]:

```
# Using TSNE for Dimentionality reduction for 15 Features(Generated after cleaning the data

from sklearn.manifold import TSNE

tsne2d = TSNE(
    n_components=2,init='random',random_state=42 ,method='barnes_hut', n_iter=2000,verbose=1
).fit_transform(X)
```

```
[t-SNE] Computing 91 nearest neighbors...
[t-SNE] Indexed 20000 samples in 0.049s...
[t-SNE] Computed neighbors for 20000 samples in 3.785s...
[t-SNE] Computed conditional probabilities for sample 1000 / 20000
[t-SNE] Computed conditional probabilities for sample 2000 / 20000
[t-SNE] Computed conditional probabilities for sample 3000 / 20000
[t-SNE] Computed conditional probabilities for sample 4000 / 20000
[t-SNE] Computed conditional probabilities for sample 5000 / 20000
[t-SNE] Computed conditional probabilities for sample 6000 / 20000
[t-SNE] Computed conditional probabilities for sample 7000 / 20000
[t-SNE] Computed conditional probabilities for sample 8000 / 20000
[t-SNE] Computed conditional probabilities for sample 9000 / 20000
[t-SNE] Computed conditional probabilities for sample 10000 / 20000
[t-SNE] Computed conditional probabilities for sample 11000 / 20000
[t-SNE] Computed conditional probabilities for sample 12000 / 20000
[t-SNE] Computed conditional probabilities for sample 13000 / 20000
[t-SNE] Computed conditional probabilities for sample 14000 / 20000
[t-SNE] Computed conditional probabilities for sample 15000 / 20000
[t-SNE] Computed conditional probabilities for sample 16000 / 20000
[t-SNE] Computed conditional probabilities for sample 17000 / 20000
[t-SNE] Computed conditional probabilities for sample 18000 / 20000
[t-SNE] Computed conditional probabilities for sample 19000 / 20000
[t-SNE] Computed conditional probabilities for sample 20000 / 20000
[t-SNE] Mean sigma: 0.098581
[t-SNE] Computed conditional probabilities in 0.886s
[t-SNE] Iteration 50: error = 104.4964066, gradient norm = 0.0344090 (50 itera
tions in 7.335s)
[t-SNE] Iteration 100: error = 85.5217056, gradient norm = 0.0091386 (50 itera
tions in 5.543s)
[t-SNE] Iteration 150: error = 81.9046173, gradient norm = 0.0049740 (50 itera
tions in 5.422s)
[t-SNE] Iteration 200: error = 80.3937607, gradient norm = 0.0037552 (50 itera
tions in 6.198s)
[t-SNE] Iteration 250: error = 79.5396271, gradient norm = 0.0026927 (50 itera
tions in 5.457s)
[t-SNE] KL divergence after 250 iterations with early exaggeration: 79.539627
[t-SNE] Iteration 300: error = 3.0753605, gradient norm = 0.0087454 (50 iterat
ions in 5.530s)
[t-SNE] Iteration 350: error = 2.4951191, gradient norm = 0.0085896 (50 iterat
ions in 6.544s)
[t-SNE] Iteration 400: error = 2.1915593, gradient norm = 0.0079488 (50 iterat
ions in 5.831s)
[t-SNE] Iteration 450: error = 2.0060658, gradient norm = 0.0074130 (50 iterat
ions in 6.253s)
[t-SNE] Iteration 500: error = 1.8798679, gradient norm = 0.0069671 (50 iterat
ions in 5.501s)
[t-SNE] Iteration 550: error = 1.7885555, gradient norm = 0.0065800 (50 iterat
ions in 5.680s)
[t-SNE] Iteration 600: error = 1.7199062, gradient norm = 0.0061970 (50 iterat
ions in 6.087s)
[t-SNE] Iteration 650: error = 1.6668059, gradient norm = 0.0057816 (50 iterat
ions in 5.482s)
[t-SNE] Iteration 700: error = 1.6248685, gradient norm = 0.0053931 (50 iterat
ions in 5.451s)
[t-SNE] Iteration 750: error = 1.5912298, gradient norm = 0.0050280 (50 iterat
ions in 6.022s)
[t-SNE] Iteration 800: error = 1.5639437, gradient norm = 0.0046949 (50 iterat
ions in 5.415s)
[t-SNE] Iteration 850: error = 1.5416846, gradient norm = 0.0042429 (50 iterat
ions in 5.438s)
[t-SNE] Iteration 900: error = 1.5233259, gradient norm = 0.0038492 (50 iterat
ions in 6.252s)
[t-SNE] Iteration 950: error = 1.5083070, gradient norm = 0.0035239 (50 iterat
ions in 5.399s)
[t-SNE] Iteration 1000: error = 1.4955441, gradient norm = 0.0031974 (50 itera
tions in 5.908s)
[t-SNE] Iteration 1050: error = 1.4847164, gradient norm = 0.0029392 (50 itera
```

```
tions in 5.554s)
[t-SNE] Iteration 1100: error = 1.4755893, gradient norm = 0.0026778 (50 itera
tions in 5.426s)
[t-SNE] Iteration 1150: error = 1.4677985, gradient norm = 0.0023920 (50 itera
tions in 6.095s)
[t-SNE] Iteration 1200: error = 1.4610808, gradient norm = 0.0022377 (50 itera
tions in 5.397s)
[t-SNE] Iteration 1250: error = 1.4552263, gradient norm = 0.0020366 (50 itera
tions in 5.416s)
[t-SNE] Iteration 1300: error = 1.4501349, gradient norm = 0.0018731 (50 itera
tions in 6.068s)
[t-SNE] Iteration 1350: error = 1.4455937, gradient norm = 0.0017068 (50 itera
tions in 5.420s)
[t-SNE] Iteration 1400: error = 1.4415125, gradient norm = 0.0015493 (50 itera
tions in 5.815s)
[t-SNE] Iteration 1450: error = 1.4378382, gradient norm = 0.0014798 (50 itera
tions in 6.246s)
[t-SNE] Iteration 1500: error = 1.4343441, gradient norm = 0.0013774 (50 itera
tions in 5.586s)
[t-SNE] Iteration 1550: error = 1.4309189, gradient norm = 0.0013995 (50 itera
tions in 5.418s)
[t-SNE] Iteration 1600: error = 1.4275470, gradient norm = 0.0013875 (50 itera
tions in 6.044s)
[t-SNE] Iteration 1650: error = 1.4243829, gradient norm = 0.0013456 (50 itera
tions in 5.422s)
[t-SNE] Iteration 1700: error = 1.4214325, gradient norm = 0.0012944 (50 itera
tions in 6.085s)
[t-SNE] Iteration 1750: error = 1.4187495, gradient norm = 0.0012517 (50 itera
tions in 6.688s)
[t-SNE] Iteration 1800: error = 1.4161776, gradient norm = 0.0012242 (50 itera
tions in 5.423s)
[t-SNE] Iteration 1850: error = 1.4137573, gradient norm = 0.0011374 (50 itera
tions in 6.084s)
[t-SNE] Iteration 1900: error = 1.4114553, gradient norm = 0.0011214 (50 itera
tions in 5.428s)
[t-SNE] Iteration 1950: error = 1.4093946, gradient norm = 0.0010452 (50 itera
tions in 5.434s)
[t-SNE] Iteration 2000: error = 1.4074937, gradient norm = 0.0009159 (50 itera
tions in 6.128s)
[t-SNE] KL divergence after 2000 iterations: 1.407494
```

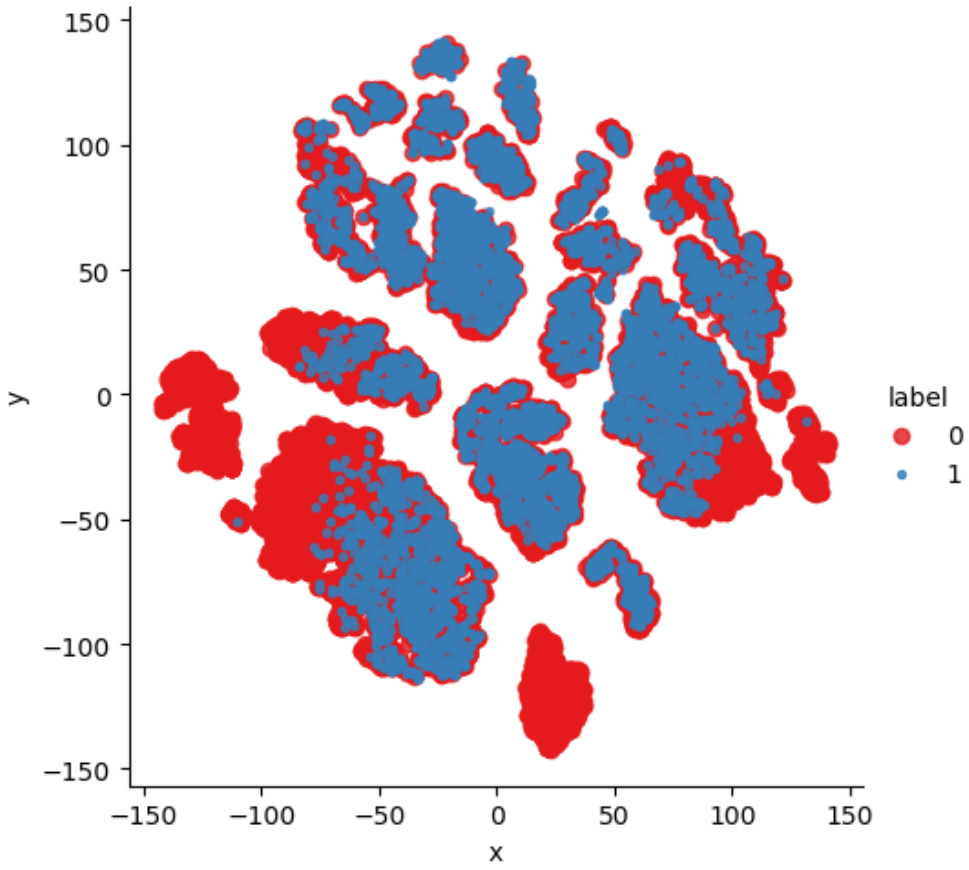
In [119]:

```
new_df1 = pd.DataFrame({'x':tsne2d[:,0], 'y':tsne2d[:,1] ,'label':y})

# draw the plot in appropriate place in the grid
sns.lmplot(data=new_df1, x='x', y='y', hue='label', fit_reg=False,palette="Set1",markers=['
```

Out[119]:

```
<seaborn.axisgrid.FacetGrid at 0x7f8389eb3290>
```



In [120]:

```
tsne3d = TSNE(
    n_components=3, init='random', random_state=42, method='barnes_hut', n_iter=1000, verbo
    angle=0.5
).fit_transform(X)
```

```
[t-SNE] Computing 91 nearest neighbors...
[t-SNE] Indexed 20000 samples in 0.047s...
[t-SNE] Computed neighbors for 20000 samples in 3.615s...
[t-SNE] Computed conditional probabilities for sample 1000 / 20000
[t-SNE] Computed conditional probabilities for sample 2000 / 20000
[t-SNE] Computed conditional probabilities for sample 3000 / 20000
[t-SNE] Computed conditional probabilities for sample 4000 / 20000
[t-SNE] Computed conditional probabilities for sample 5000 / 20000
[t-SNE] Computed conditional probabilities for sample 6000 / 20000
[t-SNE] Computed conditional probabilities for sample 7000 / 20000
[t-SNE] Computed conditional probabilities for sample 8000 / 20000
[t-SNE] Computed conditional probabilities for sample 9000 / 20000
[t-SNE] Computed conditional probabilities for sample 10000 / 20000
[t-SNE] Computed conditional probabilities for sample 11000 / 20000
[t-SNE] Computed conditional probabilities for sample 12000 / 20000
[t-SNE] Computed conditional probabilities for sample 13000 / 20000
[t-SNE] Computed conditional probabilities for sample 14000 / 20000
[t-SNE] Computed conditional probabilities for sample 15000 / 20000
[t-SNE] Computed conditional probabilities for sample 16000 / 20000
[t-SNE] Computed conditional probabilities for sample 17000 / 20000
[t-SNE] Computed conditional probabilities for sample 18000 / 20000
[t-SNE] Computed conditional probabilities for sample 19000 / 20000
[t-SNE] Computed conditional probabilities for sample 20000 / 20000
[t-SNE] Mean sigma: 0.098581
[t-SNE] Computed conditional probabilities in 0.885s
[t-SNE] Iteration 50: error = 105.6581268, gradient norm = 0.0209919 (50 itera
tions in 26.600s)
[t-SNE] Iteration 100: error = 82.9533463, gradient norm = 0.0051237 (50 itera
tions in 15.562s)
[t-SNE] Iteration 150: error = 80.2690430, gradient norm = 0.0024012 (50 itera
tions in 13.085s)
[t-SNE] Iteration 200: error = 79.2720108, gradient norm = 0.0015626 (50 itera
tions in 12.268s)
[t-SNE] Iteration 250: error = 78.7223587, gradient norm = 0.0011782 (50 itera
tions in 12.588s)
[t-SNE] KL divergence after 250 iterations with early exaggeration: 78.722359
[t-SNE] Iteration 300: error = 2.6937220, gradient norm = 0.0045005 (50 iterat
ions in 15.096s)
[t-SNE] Iteration 350: error = 2.1314061, gradient norm = 0.0035331 (50 iterat
ions in 17.822s)
[t-SNE] Iteration 400: error = 1.8505986, gradient norm = 0.0027935 (50 iterat
ions in 17.338s)
[t-SNE] Iteration 450: error = 1.6873705, gradient norm = 0.0023103 (50 iterat
ions in 17.409s)
[t-SNE] Iteration 500: error = 1.5809751, gradient norm = 0.0019538 (50 iterat
ions in 17.707s)
[t-SNE] Iteration 550: error = 1.5073045, gradient norm = 0.0016843 (50 iterat
ions in 18.028s)
[t-SNE] Iteration 600: error = 1.4538633, gradient norm = 0.0014781 (50 iterat
ions in 20.018s)
[t-SNE] Iteration 650: error = 1.4145585, gradient norm = 0.0012282 (50 iterat
ions in 17.505s)
[t-SNE] Iteration 700: error = 1.3872871, gradient norm = 0.0009881 (50 iterat
ions in 17.864s)
[t-SNE] Iteration 750: error = 1.3680828, gradient norm = 0.0007372 (50 iterat
ions in 21.582s)
[t-SNE] Iteration 800: error = 1.3542881, gradient norm = 0.0006061 (50 iterat
ions in 25.147s)
[t-SNE] Iteration 850: error = 1.3437173, gradient norm = 0.0005047 (50 iterat
ions in 18.941s)
[t-SNE] Iteration 900: error = 1.3348768, gradient norm = 0.0004386 (50 iterat
ions in 17.976s)
[t-SNE] Iteration 950: error = 1.3277340, gradient norm = 0.0003758 (50 iterat
ions in 17.603s)
[t-SNE] Iteration 1000: error = 1.3218975, gradient norm = 0.0003145 (50 itera
```

```
tions in 17.530s)
[t-SNE] KL divergence after 1000 iterations: 1.321898
```

In [121]:

```
import plotly.graph_objs as pgo
import plotly.tools as ptls
import plotly.offline as po
po.init_notebook_mode(connected=True)

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

data=[trace1]
layout=dict(height=1000, width=1000, title='3D embedding with engineered features')
fig=dict(data=data, layout=layout)
po.iplot(fig, filename='3DBubble')
```

3D embedding with engineered features

In [122]:

```
ques_df = new_df[['question1','question2']]
```

In [123]:

```
ques_df.shape
```

Out[123]:

(20000, 2)

In [124]:

```
ques_df.head(5)
```

Out[124]:

	question1	question2
398782	what is the best marketing automation tool for...	what is the best marketing automation tool for...
115086	i am poor but i want to invest what should i do	i am quite poor and i want to be very rich wh...
327711	i am from india and live abroad i met a guy f...	t i e t to thapar university to thapar univers...
367788	why do so many people in the u s hate the sou...	my boyfriend doesnt feel guilty when he hurts ...
151235	consequences of bhopal gas tragedy	what was the reason behind the bhopal gas tragedy

In [125]:

```
ques_df.sample(5)
```

Out[125]:

	question1	question2
45408	how will trump s presidency affect indian stud...	does trump s victory effect indian students fo...
357369	what is the meaning of life	what is the meaning of this life
1020	what are some interesting things to do when bored	what should i do if i am badly bored
186300	what is the ideal way of loosing weight	how should i lose weight
404268	why do not we still do great music like in the...	should i raise my young child on 80 s music

In [126]:

```
ques_df.tail(5)
```

Out[126]:

	question1	question2
36225	can i find out who is sending my public snap s...	is apple still an innovative strong competito...
333984	what do you mean by business world	what is business world
289725	how we can use waste plastics bags and bottles...	how we can use waste plastic bags and bottles ...
342679	are hackathons a good place to find technical ...	i have no background in programming but have a...
245585	is it safe to visit srinagar in september 2016	is it safe to visit srinagar in this coming se...

In [127]:

```
final_data = new_df.drop(columns=['id','qid1','qid2','question1','question2'])
```

In [128]:

```
final_data.shape
```

Out[128]:

```
(20000, 23)
```

In [129]:

```
final_data.head(5)
```

Out[129]:

	is_duplicate	chars_in_q1	chars_in_q2	words_no_words_in_q1	words_no_words_in_q2	common_w
398782	1	75	76	13	13	
115086	0	48	56	13	16	
327711	0	104	119	28	21	
367788	0	58	145	14	32	
151235	0	34	49	5	9	

5 rows × 23 columns

In [130]:

```
final_data.sample(5)
```

Out[130]:

	is_duplicate	chars_in_q1	chars_in_q2	words_no_words_in_q1	words_no_words_in_q2	common_w
1527	0	40	39	9	10	
338705	1	69	49	12	9	
301318	0	39	32	10	8	
331852	0	21	48	5	11	
289874	0	39	88	8	15	

5 rows × 23 columns

In [131]:

```
final_data.tail(5)
```

Out[131]:

	is_duplicate	chars_in_q1	chars_in_q2	words_no_words_in_q1	words_no_words_in_q2	common_w
36225	0	137	112	28	20	
333984	1	34	22	7	4	
289725	1	62	65	11	12	
342679	0	143	140	23	28	
245585	1	46	53	9	10	

5 rows × 23 columns

In [132]:

```
from sklearn.feature_extraction.text import CountVectorizer
# merge texts
questions = list(ques_df['question1']) + list(ques_df['question2'])

cv = CountVectorizer(max_features=3000)
q1_array, q2_array = np.vsplit(cv.fit_transform(questions).toarray(),2)
```

In [133]:

```
temp_df1 = pd.DataFrame(q1_array, index= ques_df.index)
temp_df2 = pd.DataFrame(q2_array, index= ques_df.index)
temp_df = pd.concat([temp_df1, temp_df2], axis=1)
```

In [134]:

```
temp_df.shape
```

Out[134]:

```
(20000, 6000)
```

In [135]:

```
temp_data.sample(5)
```

Out[135]:

	0	1	2	3	4	5	6	7	8	9	...	2991	2992	2993	2994	2995	2996	2997	2998	2999	is_duplic
52075	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
392155	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
324837	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
401781	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	
308581	0	0	0	0	0	0	0	0	0	0	...	1	0	0	0	0	0	0	0	0	

5 rows × 6001 columns

In [ ]:

```
```

In [136]:

```
final_data = pd.concat([final_data, temp_df], axis=1)
```

In [137]:

```
final_data.shape
```

Out[137]:

```
(20000, 6023)
```

In [138]:

```
final_data.sample(5)
```

Out[138]:

	is_duplicate	chars_in_q1	chars_in_q2	words_no_words_in_q1	words_no_words_in_q2	common_w
258506	1	38	40	6	7	
320189	0	65	79	12	15	
250705	1	55	33	10	7	
239731	0	130	133	20	21	
138779	0	83	53	11	7	

5 rows × 6023 columns

In [139]:

```
from sklearn.model_selection import train_test_split

X_train,X_test,y_train,y_test = train_test_split(final_data.iloc[:,1:].values,final_data.il
a = final_data.iloc[:,1:]
```

In [140]:

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
rfc = RandomForestClassifier()
rfc.fit(X_train,y_train)
y_pred = rfc.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.7865
```

In [141]:

```
"""from sklearn import svm

svm_classifier = svm.SVC(kernel='linear')

# Train the model
svm_classifier.fit(X_train, y_train)

# Make predictions on the test set
y_pred = svm_classifier.predict(X_test)

# Evaluate the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
"""
```

Out[141]:

```
'from sklearn import svm\n\nsvm_classifier = svm.SVC(kernel=\'linear\')\n\n# T
rain the model\nsvm_classifier.fit(X_train, y_train)\n\n# Make predictions on
the test set\ny_pred = svm_classifier.predict(X_test)\n\n# Evaluate the accura
cy of the model\naccuracy = accuracy_score(y_test, y_pred)\nprint("Accuracy:",
accuracy)\n'
```

In [142]:

```
from xgboost import XGBClassifier
xgb = XGBClassifier()
xgb.fit(X_train,y_train)
y_pred = xgb.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.789
```

In [143]:

```
from sklearn.tree import DecisionTreeClassifier

# Create a decision tree classifier
clf = DecisionTreeClassifier()

# Train the classifier
clf.fit(X_train, y_train)

# Make predictions on the testing data to model
y_pred = clf.predict(X_test)

# Calculate the accuracy of the classifier
accuracy = accuracy_score(y_test, y_pred) * 100
print("Accuracy:", accuracy)
```

```
Accuracy: 72.95
```

In [144]:

```
from sklearn.neural_network import MLPClassifier

# Create an MLP classifier
mlp = MLPClassifier(hidden_layer_sizes=(10, 10), max_iter=1000, random_state=42)

# Train the classifier
mlp.fit(X_train, y_train)

# Make predictions on the test set
y_pred = mlp.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
accuracy = accuracy  * 100
print("Accuracy:", accuracy )
```

```
Accuracy: 71.45
```

In [145]:

```
from sklearn.neighbors import KNeighborsClassifier

# Create a KNN classifier object
knn = KNeighborsClassifier(n_neighbors=3)

# Train the KNN classifier
knn.fit(X_train, y_train)

# Make predictions on the test set
y_pred = knn.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.697
```

In [146]:

```
from sklearn.metrics import confusion_matrix
```

In [147]:

```
# for random forest model
confusion_matrix(y_test,y_pred)
```

Out[147]:

```
array([[1960,  568],
       [ 644,  828]])
```

In [149]:

```
# for xgboost model
confusion_matrix(y_test,y_pred)
```

Out[149]:

```
array([[1960,  568],
       [ 644,  828]])
```

In [150]:

```
def test_common_words_in_question(ques1,ques2):
    word1 = set(map(lambda word: word.lower().strip(), q1.split(" ")))
    word2 = set(map(lambda word: word.lower().strip(), q2.split(" ")))
    length = len(word1 & word2)
    return length
```

In [168]:

```
def test_total_words(ques1,ques2):
    word1 = set(map(lambda word: word.lower().strip(), ques1.split(" ")))
    word2 = set(map(lambda word: word.lower().strip(), ques2.split(" ")))
    length = (len(word1) + len(word2))
    return length
```

In [178]:

```
# Advanced Feature adding
from nltk.corpus import stopwords

def test_token_features_fetching_from_questions(question1,question2):


    SAFE_DIV = 0.0000001

    STOP_WORDS = stopwords.words("english")

    token_features = [0.0]*8 # bcz of 8 features 0-7

    # Converting the Sentence into Tokens:
    question1_tokens = question1.split()
    question2_tokens = question2.split()

    if len(question1_tokens) == 0 or len(question2_tokens) == 0:
        return token_features

    # Get the non-stopwords in Questions
    question1_words = set([word for word in question1_tokens if word not in STOP_WORDS])
    question2_words = set([word for word in question2_tokens if word not in STOP_WORDS])

    #Get the stopwords in Questions
    question1_stops = set([word for word in question1_tokens if word in STOP_WORDS])
    question2_stops = set([word for word in question2_tokens if word in STOP_WORDS])

    # Get the common non-stopwords from Question pair
    common_word_count = len(question1_words.intersection(question2_words))

    # Get the common stopwords from Question pair
    common_stop_count = len(question1_stops.intersection(question2_stops))

    # Get the common Tokens from Question pair
    common_token_count = len(set(question1_tokens).intersection(set(question2_tokens)))

    token_features[0] = common_word_count / (min(len(question1_words), len(question2_words)
    token_features[1] = common_word_count / (max(len(question1_words), len(question2_words)
    token_features[2] = common_stop_count / (min(len(question1_stops), len(question2_stops)
    token_features[3] = common_stop_count / (max(len(question1_stops), len(question2_stops)
    token_features[4] = common_token_count / (min(len(question1_tokens), len(question2_toke
    token_features[5] = common_token_count / (max(len(question1_tokens), len(question2_toke

    # Last word of Q1 AND Q2 is SAME or NOT
    token_features[6] = int(question1_tokens[-1] == question2_tokens[-1])

    # First word of Q1 AND Q2 is AME or NOT
    token_features[7] = int(question1_tokens[0] == question2_tokens[0])

    return token_features
```

In [179]:

```
import distance

def test_fetch_length_features(question1,question2):

    length_features = [0.0]*3

    # Converting the Sentence into Tokens:
    question1_tokens = question1.split()
    question2_tokens = question2.split()

    if len(question1_tokens) == 0 or len(question2_tokens) == 0:
        return length_features


    #Average Token Length of both Questions
    length_features[0] = (len(question1_tokens) + len(question2_tokens))/2

    # Absolute length features
    length_features[1] = abs(len(question1_tokens) - len(question2_tokens))

    strs = list(distance.lcsubstrings(question1, question2))
    length_features[2] = len(strs[0]) / (min(len(question1), len(question2)) + 1)

    return length_features
```

In [180]:

```
# Fuzzy Features
from fuzzywuzzy import fuzz

def test_fetch_fuzzy_features_from_questions(question1, question2):

    fuzzy_features = [0.0]*4


    # fuzz_partial_ratio btw question 1 and Question 2
    fuzzy_features[0] = fuzz.partial_ratio(question1, question2)

    # fuzz_ratio btw question 1 and Question 2
    fuzzy_features[1] = fuzz.QRatio(question1, question2)

    # token_sort_ratio btw question 1 and Question 2
    fuzzy_features[2] = fuzz.token_sort_ratio(question1, question2)

    # token_set_ratio btw question 1 and Question 2
    fuzzy_features[3] = fuzz.token_set_ratio(question1, question2)

    return fuzzy_features
```

In [181]:

```
def query_point_creator(question1,question2):

    input_query = []

    # preprocess
    question1 = questions_preprocessing(q1)
    question2 = questions_preprocessing(q2)

    # fetch basic features
    input_query.append(len(question1))
    input_query.append(len(question2))

    input_query.append(len(question1.split(" ")))
    input_query.append(len(question2.split(" ")))

    input_query.append(test_common_words_in_question(question1,question2))
    input_query.append(test_total_words(question1,question2))
    input_query.append(round(test_common_words_in_question(question1,question2)/test_total_w

    # fetch token features
    token_features = test_token_features_fetching_from_questions(question1,question2)
    input_query.extend(token_features)

    # fetch length based features
    length_features = test_fetch_length_features(question1,question2)
    input_query.extend(length_features)

    # fetch fuzzy features
    fuzzy_features = test_fetch_fuzzy_features_from_questions(question1,question2)
    input_query.extend(fuzzy_features)

    # bag of words feature for q1
    question1_bow = cv.transform([question1]).toarray()

    # bag oo words feature for q2
    question2_bow = cv.transform([question2]).toarray()



    return np.hstack((np.array(input_query).reshape(1,22),question1_bow,question2_bow))
```

In [ ]:

In [ ]:

In [ ]: