Importing libraries

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

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

# from google.colab import drive
# drive.mount('/content/drive')
train_df =pd.read_csv("../input/aidescalatingdataset/train.csv")
test =pd.read_csv("../input/test-dataset/test.csv")
print("Train shape : ",train_df.shape)

test['label'] = 2
test['indicator'] = 0
train_df['indicator'] = 1
train = pd.concat([test,train_df])
train.head()
```

```
Train shape :  (4437, 27)
```

	link	link_id	page_description	alchemy_
0	http://www.ellesnewenglandkitchen.com/blog/200...	4049	{"title":"Elle s New England Kitchen Elle s Ne...	arts_ent
1	http://www.alternet.org/story/149193/study_con...	3692	{"url":"alternet org story 149193 study confir...	cultu
2	http://www.wiredberries.com/	9739	{"title":" ","body":" ","url":"wiredberries"}	
3	http://www.elements4health.com/cayenne-pepper....	1548	{"title":"The Health Benefits of Cayenne Peppe...	
4	http://www.poorgirleatswell.com/2009/10/recipe...	5574	{"title":"Recipe Hearty Mushroom Potato Soup "...	

5 rows × 28 columns

```
test.shape
```

```
(2958, 28)
```

```
train.columns
```

```
Index(['link', 'link_id', 'page_description', 'alchemy_category',
       'alchemy_category_score', 'avg_link_size', 'common_word_link_ratio_1',
       'common_word_link_ratio_2', 'common_word_link_ratio_3',
       'common_word_link_ratio_4', 'compression_ratio', 'embed_ratio',
       'frame_based', 'frame_tag_ratio', 'has_domain_link', 'html_ratio',
       'image_ratio', 'is_news', 'lengthy_link_domain', 'link_word_score',
       'news_front_page', 'non_markup_alphanumeric_characters',
       'count_of_links', 'number_of_words_in_url', 'parametrized_link_ratio',
       'spelling_mistakes_ratio', 'label', 'indicator'],
      dtype='object')
```

```
train.head()
```

	link	link_id	page_description	alchem
0	http://www.ellesnewenglandkitchen.com/blog/200...	4049	{"title":"Elle s New England Kitchen Elle s Ne...	arts_

```
# Y=train['label']
# Y.value_counts()
```

```
X=train;
X.head()
```

	link	link_id	page_description	alchem
0	http://www.ellesnewenglandkitchen.com/blog/200...	4049	{"title":"Elle s New England Kitchen Elle s Ne...	arts_
1	http://www.alternet.org/story/149193/study_con...	3692	{"url":"alternet org story 149193 study confir...	c
2	http://www.wiredberries.com/	9739	{"title":" ","body":" ","url":"wiredberries"}	
3	http://www.elements4health.com/cayenne-pepper....	1548	{"title":"The Health Benefits of Cayenne Peppe...	
4	http://www.poorgirleatswell.com/2009/10/recipe...	5574	{"title":"Recipe Hearty Mushroom Potato Soup "...	

5 rows × 28 columns

PRE-PROCESSING EDA

checking null values

```
X.isna().sum()
```

```
link                                  0
link_id                               0
page_description                      0
alchemy_category                      0
alchemy_category_score                0
avg_link_size                         0
common_word_link_ratio_1              0
common_word_link_ratio_2              0
common_word_link_ratio_3              0
common_word_link_ratio_4              0
compression_ratio                     0
embed_ratio                           0
frame_based                           0
frame_tag_ratio                       0
has_domain_link                       0
html_ratio                            0
image_ratio                           0
is_news                               0
lengthy_link_domain                   0
link_word_score                       0
news_front_page                       0
non_markup_alphanumeric_characters    0
count_of_links                        0
number_of_words_in_url                0
parametrized_link_ratio               0
spelling_mistakes_ratio               0
label                                 0
indicator                             0
dtype: int64
```

```
(X == "?").sum()
```

```
link                                     0
link_id                                  0
page_description                         0
alchemy_category                      2342
alchemy_category_score                2342
avg_link_size                            0
common_word_link_ratio_1                 0
common_word_link_ratio_2                 0
```

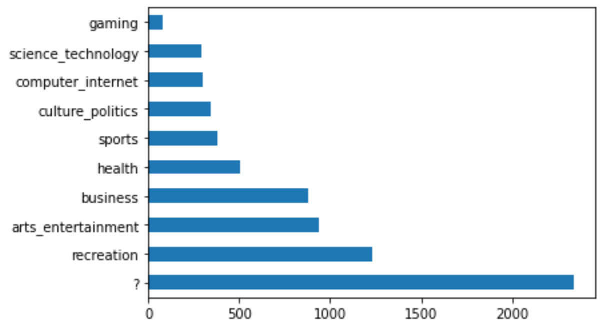
```
common_word_link_ratio_3                0
common_word_link_ratio_4                0
compression_ratio                       0
embed_ratio                             0
frame_based                             0
frame_tag_ratio                         0
has_domain_link                         0
html_ratio                              0
image_ratio                             0
is_news                              2843
lengthy_link_domain                     0
link_word_score                         0
news_front_page                      1248
non_markup_alphanumeric_characters      0
count_of_links                          0
number_of_words_in_url                  0
parametrized_link_ratio                 0
spelling_mistakes_ratio                 0
label                                   0
indicator                               0
dtype: int64
```

```
X['alchemy_category'].value_counts()
```

```
?                     2342
recreation            1229
arts_entertainment     941
business               880
health                 506
sports                 380
culture_politics       343
computer_internet      296
science_technology     289
gaming                  76
religion                72
law_crime               31
unknown                  6
weather                  4
Name: alchemy_category, dtype: int64
```

```
train['alchemy_category'].value_counts()[:10].plot(kind='barh')
```

<AxesSubplot:>




```
X['alchemy_category'].value_counts
```

```
<bound method IndexOpsMixin.value_counts of 0       arts_entertainment
1         culture_politics
2                        ?
3                        ?
4               recreation
               ...
4432                sports
4433                     ?
4434      culture_politics
4435      culture_politics
4436                sports
Name: alchemy_category, Length: 7395, dtype: object>
```

```
# replacing ? values with random value
X['alchemy_category'] = X['alchemy_category'].replace(to_replace ="?",value =random.choice(X['alchemy_category'].values.tolist()))
X['alchemy_category']
```

```
0       arts_entertainment
1         culture_politics
2                        ?
3                        ?
4               recreation
               ...
4432                sports
4433                     ?
4434      culture_politics
```

```
4435      culture_politics
4436                sports
Name: alchemy_category, Length: 7395, dtype: object
```

```
X.columns
```

```
Index(['link', 'link_id', 'page_description', 'alchemy_category',
       'alchemy_category_score', 'avg_link_size', 'common_word_link_ratio_1',
       'common_word_link_ratio_2', 'common_word_link_ratio_3',
       'common_word_link_ratio_4', 'compression_ratio', 'embed_ratio',
       'frame_based', 'frame_tag_ratio', 'has_domain_link', 'html_ratio',
       'image_ratio', 'is_news', 'lengthy_link_domain', 'link_word_score',
       'news_front_page', 'non_markup_alphanumeric_characters',
       'count_of_links', 'number_of_words_in_url', 'parametrized_link_ratio',
       'spelling_mistakes_ratio', 'label', 'indicator'],
      dtype='object')
```

```
X['alchemy_category_score'] = X['alchemy_category_score'].replace(to_replace ="?",value = (X['alchemy_category_score']!='?').mean())
X['alchemy_category_score'] = X['alchemy_category_score'].astype('float')
```

```
print('median: ',X['alchemy_category_score'].median())
print('mean : ',X['alchemy_category_score'].mean())
```

```
median:  0.6832995267072346
mean :  0.6286593365041708
```

```
X['news_front_page'] = X['news_front_page'].replace(to_replace ="?",value =0)
X['is_news'] = X['is_news'].replace(to_replace ="?",value =1)
```

```
X.dtypes
```

```
link                                    object
link_id                                  int64
page_description                        object
alchemy_category                        object
alchemy_category_score                 float64
avg_link_size                          float64
common_word_link_ratio_1               float64
common_word_link_ratio_2               float64
common_word_link_ratio_3               float64
common_word_link_ratio_4               float64
compression_ratio                      float64
embed_ratio                            float64
frame_based                              int64
frame_tag_ratio                        float64
has_domain_link                          int64
html_ratio                             float64
image_ratio                            float64
is_news                                 object
lengthy_link_domain                      int64
link_word_score                          int64
news_front_page                         object
non_markup_alphanumeric_characters       int64
count_of_links                           int64
number_of_words_in_url                   int64
parametrized_link_ratio                float64
spelling_mistakes_ratio                float64
label                                    int64
indicator                                int64
dtype: object
```

```
X['is_news']=X['is_news'].astype('float')
X['news_front_page']=X['news_front_page'].astype('float')
```

```
(X == "?").sum()
```

```
link                                     0
link_id                                  0
page_description                         0
alchemy_category                      2342
alchemy_category_score                   0
avg_link_size                            0
common_word_link_ratio_1                 0
common_word_link_ratio_2                 0
common_word_link_ratio_3                 0
common_word_link_ratio_4                 0
compression_ratio                        0
embed_ratio                              0
frame_based                              0
frame_tag_ratio                          0
has_domain_link                          0
html_ratio                               0
image_ratio                              0
is_news                                  0
```

```
lengthy_link_domain                  0
link_word_score                      0
news_front_page                      0
non_markup_alphanumeric_characters   0
count_of_links                       0
number_of_words_in_url               0
parametrized_link_ratio              0
spelling_mistakes_ratio              0
label                                0
indicator                            0
dtype: int64
```

```
X.head()
```

	link	link_id	page_description	alchem
0	http://www.ellesnewenglandkitchen.com/blog/200...	4049	{"title":"Elle s New England Kitchen Elle s Ne...	arts_
1	http://www.alternet.org/story/149193/study_con...	3692	{"url":"alternet org story 149193 study confir...	c
2	http://www.wiredberries.com/	9739	{"title":" ","body":" ","url":"wiredberries"}	
3	http://www.elements4health.com/cayenne-pepper....	1548	{"title":"The Health Benefits of Cayenne Peppe...	
4	http://www.poorgirleatswell.com/2009/10/recipe...	5574	{"title":"Recipe Hearty Mushroom Potato Soup "...	

5 rows × 28 columns

One hot encoding

[ ] ↳ *1 cell hidden*

dropping column with all 0 values

[ ] ↳ *5 cells hidden*

Removing outliers

The interquartile range (IQR), also called the midspread or middle 50%, or technically H-spread, is a measure of statistical dispersion, being equal to the difference between 75th and 25th percentiles, or between upper and lower quartiles, IQR = Q3 − Q1.

In other words, the IQR is the first quartile subtracted from the third quartile; these quartiles can be clearly seen on a box plot on the data.

It is a measure of the dispersion similar to standard deviation or variance, but is much more robust against outliers.

[ ] ↳ *24 cells hidden*

NLP Pre Processsing

```
from urllib.parse import urlparse
X.link=X.link.apply(lambda x:urlparse(x).netloc)
X.link
```

```
0        www.ellesnewenglandkitchen.com
1                      www.alternet.org
2                  www.wiredberries.com
3              www.elements4health.com
4              www.poorgirleatswell.com
                      ...
4432                  newsfeed.time.com
4433                   tastykitchen.com
4434                         ecoble.com
4435             www.huffingtonpost.com
4436                    www.bromygod.com
Name: link, Length: 7395, dtype: object
```

```
X.head()
```

	link	link_id	page_description	alchemy_category_s
0	www.ellesnewenglandkitchen.com	4049	{"title":"Elle s New England Kitchen Elle s Ne...	0.36
1	www.alternet.org	3692	{"url":"alternet org story 149193 study confir...	0.87
2	www.wiredberries.com	9739	{"title":" ","body":" ","url":"wiredberries"}	0.68
3	www.elements4health.com	1548	{"title":"The Health Benefits of Cayenne Peppe...	0.68
4	www.poorgirleatswell.com	5574	{"title":"Recipe Hearty Mushroom Potato Soup "...	0.74

5 rows × 39 columns

```
X['page_description'].replace(to_replace=r'"title":', value="",inplace=True,regex=True)
X['page_description'].replace(to_replace=r'"url":',value="",inplace=True,regex=True)
X['page_description'].replace(to_replace=r'"body":',value="",inplace=True,regex=True)
X['page_description'].replace(to_replace=r'{|}',value="",inplace=True,regex=True)
X['page_description'].head()
```

```
0    "Elle s New England Kitchen Elle s New England...
1    "alternet org story 149193 study confirms that...
2                                 " "," ","wiredberries"
3    "The Health Benefits of Cayenne Pepper ","Brie...
4    "Recipe Hearty Mushroom Potato Soup ","If you ...
Name: page_description, dtype: object
```

```
X.head()
```

	link	link_id	page_description	alchemy_category_s
0	www.ellesnewenglandkitchen.com	4049	"Elle s New England Kitchen Elle s New England...	0.36
1	www.alternet.org	3692	"alternet org story 149193 study confirms that...	0.87
2	www.wiredberries.com	9739	" "," ","wiredberries"	0.68
3	www.elements4health.com	1548	"The Health Benefits of Cayenne Pepper ","Brie...	0.68
4	www.poorgirleatswell.com	5574	"Recipe Hearty Mushroom Potato Soup ","If you ...	0.74

5 rows × 39 columns

1. The **isalpha()** method returns True if all the characters are alphabet letters (a-z).Example of characters that are not alphabet letters: (space)!#%&? etc.
2. **Stop Words:** A stop word is a commonly used word (such as “the”, “a”, “an”, “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. We would not want these words to take up space in our database, or taking up valuable processing time. For this, we can remove them easily, by storing a list of words that you consider to stop words.
3. **word_tokenize:** In Natural Language Processing, tokenization divides a string into a list of tokens. Tokens come in handy when finding valuable patterns and helping to replace sensitive data components with non-sensitive ones. word_tokenize is a function in Python that splits a given sentence into words using the NLTK library.

```
import nltk
nltk.download('punkt')
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
wordnet = WordNetLemmatizer()
from nltk.corpus import stopwords
nltk.download('stopwords')

def textCleaning(df,column_name):
    cleanList = list()
```

```
    lines = df[column_name].values.tolist()
    for text in lines:
        text = text.lower()
        words = word_tokenize(text)
        stop_words = set(stopwords.words("english"))
        words = [w for w in words if not w in stop_words]
        words = [w for w in words if w.isalpha()]
        words = ' '.join(words)
        cleanList.append(words)
    return cleanList
```

```
[nltk_data] Downloading package punkt to /usr/share/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /usr/share/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
pageDescription = textCleaning(X,"page_description")
pageDescription[0:2]
```

```
['elle new england kitchen elle new england kitchen weeks ago sarah homemade asked like start making recipes together
would blog experiences course said yes sarah sweetheart like baking friend kitchen except west coast east coast one two
month stay tuned first one decided try pita bread great recipe dough dream work needed add flour kneading quite sticky
get texture right good time consuming part set dough aside rise got minute rise minute rest another minute rest baking
take minutes bake makes right even better rip one open law legally required rip open piece bread hot oven even causes
acute pain digits rip one open get aroma yeasty heavenly steamy bread even need butter enjoy although pretty sure law
somewhere hot bread butter going together come think also law making blog posts ramble recipe found brown eyed baker
used kitchenaid stand mixer let kneading ten minutes addition cup flour dough perfect doubled recipe get pita breads
family every one kids pita bread butter wait two sons butter little hoodlums breaking warm bread butter law early age
dinner see eight enough husband dinner burger recipe upcoming post try one recipe blog burgers would top favorites
mmmmm coming end ramblings pita breads easy make would really fun kids kinda fun watch puff oven thanks sarah lot fun
whole family enjoyed bread even picky ones kids picky bread bet picky bread like butter either ellesnewenglandkitchen
blog pita make pita bread html',
 'alternet org story study confirms fox news makes stupid study confirms fox news makes stupid study confirms fox news
makes stupid new survey american voters shows fox news viewers significantly misinformed consumers news sources
december yet another study released proving watching fox news detrimental intelligence world public opinion project
managed program international policy attitudes university maryland conducted survey american voters shows fox news
viewers significantly misinformed consumers news sources study shows greater exposure fox news increases misinformation
watch less know precise think know actually false study corroborates previous pipa study focused iraq war similar
results nbc wall street journal poll demonstrated break reality part fox viewers regard health care body evidence fox
news nothing propaganda machine dedicated lies growing day eight nine questions fox news placed first percentage
misinformed placed second question tarp pretty high batting average journalistic fraud list fox news viewers believe
aint percent believe stimulus legislation lost jobs percent believe health reform law increase deficit percent believe
economy getting worse percent believe climate change occurring percent believe income taxes gone percent believe
stimulus legislation include tax cuts percent believe obama initiated gm chrysler bailout percent believe republicans
opposed tarp percent believe obama born u unclear conclusion inescapable fox news deliberately misinforming viewers
reason every issue one republican party vested interest gop benefited ignorance fox news helped proliferate results
apparent election last month voters based decisions demonstrably false information fed fox news way rest media
blameless cnn broadcast network news operations fared slightly better many cases even msnbc best record accurately
informing viewers ways go brag conclusions study need disseminated broadly possible fox competitors need report results
produce ad campaigns featuring newspapers magazines need publish study across country big news critical nation advised
major news enterprise poisoning minds isolated review fox performance corroborated time time fact fox news blatantly
dishonest effects dishonesty become ingrained electorate purposefully deceived needs made known every american
democracy function voters making choices based lies evidence fox tilting scales must make certain corporate owners get
away mark howard artist author publisher news corpse independent media alternative news media activism drug war new
survey american voters shows fox news viewers significantly misinformed consumers news sources']
```

TF-IDF vectorization

TF-IDF stands for Term Frequency — Inverse Document Frequency and is a statistic that aims to better define how important a word is for a document, while also taking into account the relation to other documents from the same corpus.

The rationale behind this is the following:

- a word that frequently appears in a document has more relevancy for that document, meaning that there is higher probability that the document is about or in relation to that specific word
- a word that frequently appears in more documents may prevent us from finding the right document in a collection; the word is relevant either for all documents or for none. Either way, it will not help us filter out a single document or a small subset of documents from the whole set.

TF-IDF is a score which is applied to every word in every document in our dataset. And for every word, the TF-IDF value increases with every appearance of the word in a document, but is gradually decreased with every appearance in other documents.

WHY TD-IDF over BoW?

the initial step of bag-of-words acts as a downside because it emphasizes words only based on counts. To overcome this, a simple twist to bag-of-words introduces the tf-idf approach.

Unlike, bag-of-words, tf-idf creates a normalized count where each word count is divided by the number of documents this word appears in.

> **bow(w, d) = # times word w appears in document d.**

> ***tf-idf(w, d) = bow(w, d) x N / (# documents in which word w appears)***

min_df is used for removing terms that appear too infrequently. For example:

min_df = 0.01 means "ignore terms that appear in less than 1% of the documents".

min_df = 1 means "ignore terms that appear in less than 1 documents".

```
from sklearn.feature_extraction.text import TfidfVectorizer
TV = TfidfVectorizer(min_df=1)
```

```
def chkNonzero(df,col):
    for i in df[col+'_0']: # checking non null values for words in document 1
      if(i != 0.00):
        print(i)
```

```
Z = TV.fit_transform(pageDescription).toarray()
arrayCols = len(Z[0])
print('Shape : ',np.shape(Z),'\n')
columns = [f'pageDescription_{num}' for num in range(arrayCols)]
df_pageDescription =  pd.DataFrame(Z, columns=columns)
chkNonzero(df_pageDescription,'pageDescription')
```

```
Shape :  (7395, 78185)

0.0025915282530577324
0.04105453632902744
0.002596059771451365
0.05913979252219915
0.02675493944400716
0.012818520847319492
0.015283500466449879
0.018359562050246428
0.09084697249475217
0.017480213818118352
0.02585804919241791
0.03645746146791632
0.07530509381003035
0.03264670223163986
0.035524425891513264
0.08208210230742576
0.29206593872035475
```

```
df_pageDescription.shape
```

```
(7395, 78185)
```

feature scaling and joining vectorize data with other feature columns

```
X.head()
```

	link	link_id	page_description	alchemy_category_s
0	www.ellesnewenglandkitchen.com	4049	"Elle s New England Kitchen Elle s New England...	0.36
1	www.alternet.org	3692	"alternet org story 149193 study confirms that...	0.87
2	www.wiredberries.com	9739	" "," ","wiredberries"	0.68
3	www.elements4health.com	1548	"The Health Benefits of Cayenne Pepper ","Brie...	0.68
4	www.poorgirleatswell.com	5574	"Recipe Hearty Mushroom Potato Soup ","If you ...	0.74

5 rows × 39 columns

```
len(X.columns)
```

```
39
```

```
X.columns
```

```
Index(['link', 'link_id', 'page_description', 'alchemy_category_score',
       'avg_link_size', 'common_word_link_ratio_1', 'common_word_link_ratio_2',
       'common_word_link_ratio_3', 'common_word_link_ratio_4',
       'compression_ratio', 'embed_ratio', 'frame_tag_ratio',
       'has_domain_link', 'html_ratio', 'image_ratio', 'lengthy_link_domain',
       'link_word_score', 'news_front_page',
       'non_markup_alphanumeric_characters', 'count_of_links',
       'number_of_words_in_url', 'parametrized_link_ratio',
       'spelling_mistakes_ratio', 'label', 'indicator', 'alchemy_category_?',
       'alchemy_category_arts_entertainment', 'alchemy_category_business',
       'alchemy_category_computer_internet',
       'alchemy_category_culture_politics', 'alchemy_category_gaming',
       'alchemy_category_health', 'alchemy_category_law_crime',
       'alchemy_category_recreation', 'alchemy_category_religion',
       'alchemy_category_science_technology', 'alchemy_category_sports',
       'alchemy_category_unknown', 'alchemy_category_weather'],
      dtype='object')
```

```
# dropping few unrelated columns
X.drop(axis="columns", labels="link_id", inplace=True)
X.drop(axis="columns", labels="page_description", inplace=True)
X.drop(axis="columns", labels="link", inplace=True)
X.drop(axis="columns", labels="label", inplace=True)
X.drop(axis="columns", labels="indicator", inplace=True)
```

```
X.head()
```

	alchemy_category_score	avg_link_size	common_word_link_ratio_1	common_w
0	0.365831	1.217617	0.261307	
1	0.876315	3.814208	0.589744	
2	0.683300	1.793103	0.402299	
3	0.683300	2.083333	0.636364	
4	0.747449	1.845815	0.676856	

5 rows × 34 columns

When data contains outliers, StandardScaler can often be mislead. In such cases, it is better to use a scaler that is robust against outliers.

```
X.head()
```

	alchemy_category_score	avg_link_size	common_word_link_ratio_1	common_w
0	0.365831	1.217617	0.261307	
1	0.876315	3.814208	0.589744	
2	0.683300	1.793103	0.402299	
3	0.683300	2.083333	0.636364	
4	0.747449	1.845815	0.676856	

5 rows × 34 columns

```
#robust scaling is used to handle outliers
import pandas as pd
from sklearn.preprocessing import RobustScaler
scaler = RobustScaler()
X = pd.DataFrame(scaler.fit_transform(X), columns=X.columns)
```

```
X.head()
```

	alchemy_category_score	avg_link_size	common_word_link_ratio_1	common_w
0	-1.872733	-0.849062	-0.797059	
1	1.138590	1.683237	0.391921	
2	0.000000	-0.287824	-0.286650	
3	0.000000	-0.004781	0.560692	
4	0.378415	-0.236418	0.707278	

5 rows × 34 columns

we are concatenating all the columns obtained after the TD-IDF

```
X.reset_index(inplace=True, drop=True)
```

```
horizontal_concat = pd.concat([df_pageDescription,X], axis=1)
```

```
horizontal_concat.shape
```

```
(7395, 78219)
```

```
horizontal_concat.tail()
```

	pageDescription_0	pageDescription_1	pageDescription_2	pageDescripti
7390	0.0	0.0	0.0	
7391	0.0	0.0	0.0	
7392	0.0	0.0	0.0	
7393	0.0	0.0	0.0	
7394	0.0	0.0	0.0	

5 rows × 78219 columns

Train-Test Split

```
# import numpy as np
# from sklearn.model_selection import train_test_split
```

```
# X_train, X_test, y_train, y_test = train_test_split(horizontal_concat, Y, test_size=0.33, random_state=42)
```

```
# print("Shape of new dataframes - {} , {}".format(X_train.shape, X_test.shape, y_train.shape, y_test.shape))
```

```
# y_train
```

```
X_test_df = horizontal_concat.iloc[:2958,:]
X_train_df  = horizontal_concat.iloc[2958:,:]
print("Shape of new dataframes - {} , {}".format(X_test_df.shape, X_train_df.shape))

Y_train = train_df['label']
print("Y_train_df shape : ",Y_train.shape)
```

```
Shape of new dataframes - (2958, 78219) , (4437, 78219)
Y_train_df shape :  (4437,)
```

LOGISTIC REGRESSION

```
"""using logistic regression"""
from sklearn.linear_model import LogisticRegression

# instantiate the model (using the default parameters)
logreg = LogisticRegression(max_iter = 1500)

# fit the model with data
logreg.fit(X_train_df, Y_train)

#[:,1] this is applied to take positive probablities
y_pred_logreg=logreg.predict_proba(X_test_df)[:,1]
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/linear_model/_logistic.py:818: ConvergenceWarning: lbfgs failed to conver
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
```

```
# y_pred
y_pred_logreg
```

```
array([0.91561575, 0.14913372, 0.31642757, ..., 0.35342764, 0.28920958,
       0.33833196])
```

```
sample_sum=pd.read_csv("../input/aid-escalating-internet-coverage/sample_submission.csv")
sample_sum["label"]=y_pred_logreg
sample_sum.to_csv("./sum.csv",index=False)
```