REPORT

Title of project: **Choosing the best way for Text Classification in Python3**

Group name: Mamamia

Group members: Nan Wang, z5238059
Wenbin Wang, z5239391
Zhaokun Su, z5235878
Yangyang Liu, z5244484
Jieyun Xing, z5234434

Introduction

The problem we want to solve is to judge the topic of one article using the training set. Our work is trying to make computers adapt their actions so that these actions can get more accurate. In the process, 9500 articles are used as a training set and 500 articles are used as a test set.

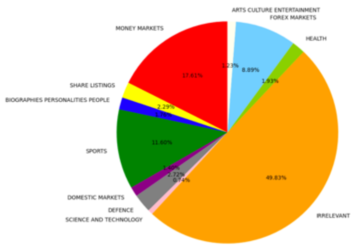
Note:

1. The distribution of articles over topics for training data set and testing data set is not uniform. It means that maybe some of topics (labels) gain more popularity than other topics.
2.The amount of data for each data unequal.
3. Articles only have single words not sentences, so they are lack sentence meanings, which may cause a low accuracy.
4.The label of 'FOREX MARKETS' and 'MONEY MARKETS' have very similar articles, so it is difficult to distinguish these two labels.

Exploratory data analysis

- The amount of data for each label

Topic name	Number
IRRELEVANT	4734
MONEY MARKETS	1673
SPORTS	1102
FOREX MARKETS	845
DEFENCE	258
SHARE LISTINGS	218
HEALTH	183
BIOGRAPHIES PERSONALITIES PEOPLE	167
DOMESTIC MARKETS	133
ARTS CULTURE ENTERTAINMENT	117
SCIENCE AND TECHNOLOGY	70

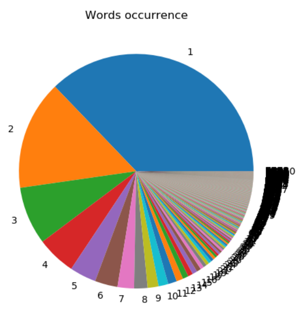


- It can be found easily that the number of articles contained in different topics are significantly different.
- And nearly half of the articles are irrelevant.
- Even ignoring irrelevant articles, the sizes of these topic sets have huge difference.

(just for articles concluded in training set)

- Words occurrence

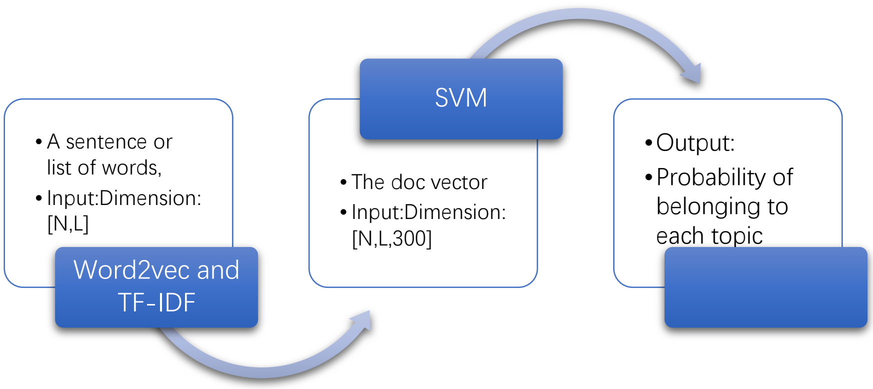
	Words occurrence
count	857.000000
mean	41.738623
std	509.090486
min	1.000000
25%	1.000000
50%	2.000000
75%	4.000000
max	13300.000000



- More than 50% of words only occur no more than twice.
- The pre-train data contains most of the words.
- For this, 2 methods can be applied: one is keeping them and the other one is to delete them.

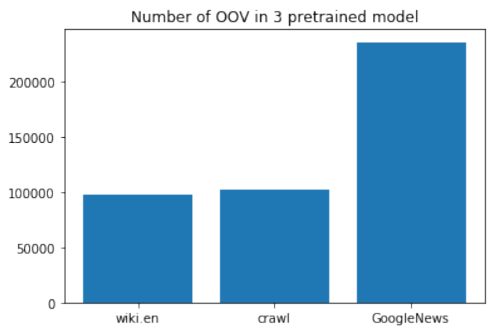
Methodology

- **Method 1: word2vec + TF-IDF+SVM**


• A sentence or list of words,
• Input:Dimension: [N,L]
Word2vec and TF-IDF
SVM
• The doc vector
• Input:Dimension: [N,L,300]
• Output:
• Probability of belonging to each topic

a. Word2vec

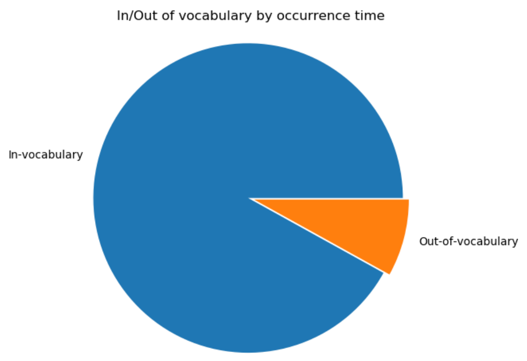
Because the training data are not complete sentences, pre-trained word2vec data should work well in this project because it contains context relationship in it. we choose the vector data trained by Fasttext with skip-gram model[1] (which can be download here: https://fasttext.cc/docs/en/pretrained-vectors.html , English-text: wiki.en.vec) which has been trained on Wikipedia and the vector in dimension 300, and we compare 3 models, this one can cover most training-set words.

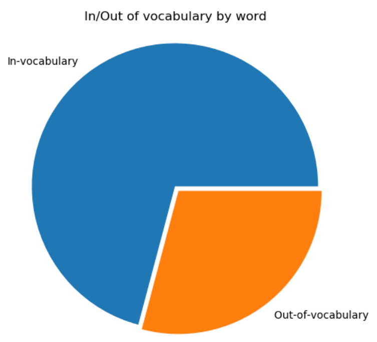


Number of OVV in 3 model (count by occurrence time)	
Wiki.en	97172
Crawl	101901
GoogleNews	235486

In word2vec, semantically similar words are very close to each other in spatial coordinates, while semantically unrelated words are far apart. This property can be used for a more generalized analysis of words and sentences.

By checking the words in, I found there are too many OVVs, so I just delete them in the training set, because these words cannot improve the model precision and it will cause useless dimensions.





In/Out of vocabulary			
In-vocabulary by occurrence time	1110632	In-vocabulary by word	25352
Out-of-vocabulary by occurrence time	97172	Out-of-vocabulary by word	10418
Total occurrence time	1207804	Total word	35770

b. TF-IDF

TF-IDF[2] (Term Frequency-Invers Document Frequency) calculates the importance of a word in the entire corpus based on the number of times the word appears in the text and the frequency of the document that appears in the entire corpus.

$$TFIDF_{i,j} = TF_{i,j} * IDF_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} * log\frac{|D|}{1+|D_{t_i}|}$$

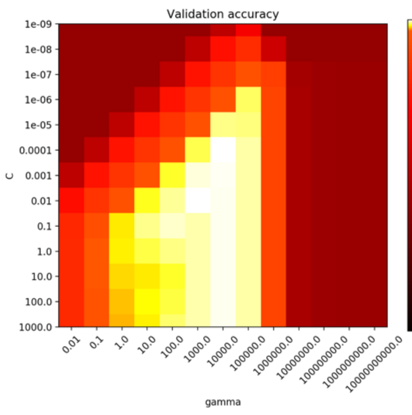
$n_{i,j}$ is the number of times feature word t_i appears in text d_j, $\sum_k n_{k,j}$ is the number of all feature words in the text d_j, $TF_{i,j}$ is the word frequency for a characteristic word.

$|\mathrm{D}|$ is the total number of texts in the corpus, $|\mathrm{D}_{\mathrm{t_i}}|$ indicating the number of feature words $\mathrm{t_i}$ in the text.

We can count each doc vector with pre trained word2vec model and TF-IDF as weights.

c. SVM

After calculated doc vector, each train or test case can be represented by 300-dim vector. The SVM theory has been learned on lecture and will not be repeated here. And two (c and gamma) parameters were tested using GCV.



According to the result, we choose gamma = 1000 and c = 0.001, because the parameters (10000, 0.0001) and (1000, 0.1) are overfitting.

- **Method 2: Multinomial Naïve Bayes Model**

  a. **Data cleaning and Data pre-processing**

     1. Remove IRRELEVANT articles
     2. Remove all "stop words" (which means they are not meaningful for our classification)

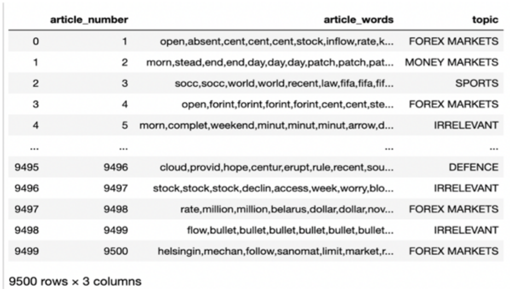
     Our original data contains all categorical data (including class "IRRELEVANT"), this class is not relevant to our objective classes.

     it can be seen that there are too many words in some specific article. We are supposed to distinguish which part of words is meaningful and which part is not. "stop words" is all about topics in our daily life.

     Therefore, these two parts can be deleted.

(Here is the comparison of before and after the processing)

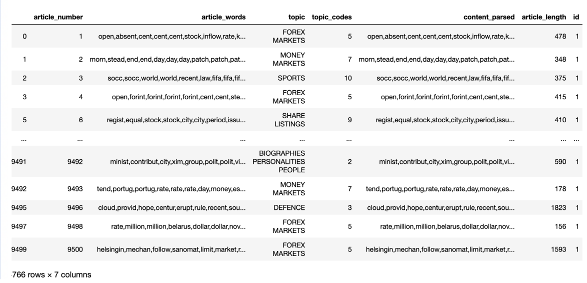
Before



	article_number	article_words	topic
0	1	open,absent,cent,cent,cent,stock,inflow,rate,k...	FOREX MARKETS
1	2	morn,stead,end,end,day,day,day,patch,patch,pat...	MONEY MARKETS
2	3	socc,socc,world,world,recent,law,fifa,fifa,fif...	SPORTS
3	4	open,forint,forint,forint,forint,cent,cent,ste...	FOREX MARKETS
4	5	morn,complet,weekend,minut,minut,minut,arrow,d...	IRRELEVANT
...	...	...	...
9495	9496	cloud,provid,hope,centur,erupt,rule,recent,sou...	DEFENCE
9496	9497	stock,stock,stock,declin,access,week,worry,blo...	IRRELEVANT
9497	9498	rate,million,million,belarus,dollar,dollar,nov...	FOREX MARKETS
9498	9499	flow,bullet,bullet,bullet,bullet,bullet,bullet...	IRRELEVANT
9499	9500	helsingin,mechan,follow,sanomat,limit,market,r...	FOREX MARKETS

9500 rows × 3 columns

After



	article_number	article_words	topic	topic_codes	content_parsed	article_length	id
0	1	open,absent,cent,cent,cent,stock,inflow,rate,k...	FOREX MARKETS	5	open,absent,cent,cent,cent,stock,inflow,rate,k...	478	1
1	2	morn,stead,end,end,day,day,day,patch,patch,pat...	MONEY MARKETS	7	morn,stead,end,end,day,day,day,patch,patch,pat...	348	1
2	3	socc,socc,world,world,recent,law,fifa,fifa,fif...	SPORTS	10	socc,socc,world,world,recent,law,fifa,fifa,fif...	375	1
3	4	open,forint,forint,forint,forint,cent,cent,ste...	FOREX MARKETS	5	open,forint,forint,forint,forint,cent,cent,ste...	415	1
5	6	regist,equal,stock,stock,city,city,period,issu...	SHARE LISTINGS	9	regist,equal,stock,stock,city,city,period,issu...	410	1
...	...	...	...	...	...	...	...
9491	9492	minist,contribut,city,xim,group,polit,polit,vi...	BIOGRAPHIES PERSONALITIES PEOPLE	2	minist,contribut,city,xim,group,polit,polit,vi...	590	1
9492	9493	tend,portug,portug,rate,rate,rate,day,money,es...	MONEY MARKETS	7	tend,portug,portug,rate,rate,rate,day,money,es...	178	1
9495	9496	cloud,provid,hope,centur,erupt,rule,recent,sou...	DEFENCE	3	cloud,provid,hope,centur,erupt,rule,recent,sou...	1823	1
9497	9498	rate,million,million,belarus,dollar,dollar,nov...	FOREX MARKETS	5	rate,million,million,belarus,dollar,dollar,nov...	156	1
9499	9500	helsingin,mechan,follow,sanomat,limit,market,r...	FOREX MARKETS	5	helsingin,mechan,follow,sanomat,limit,market,r...	1593	1

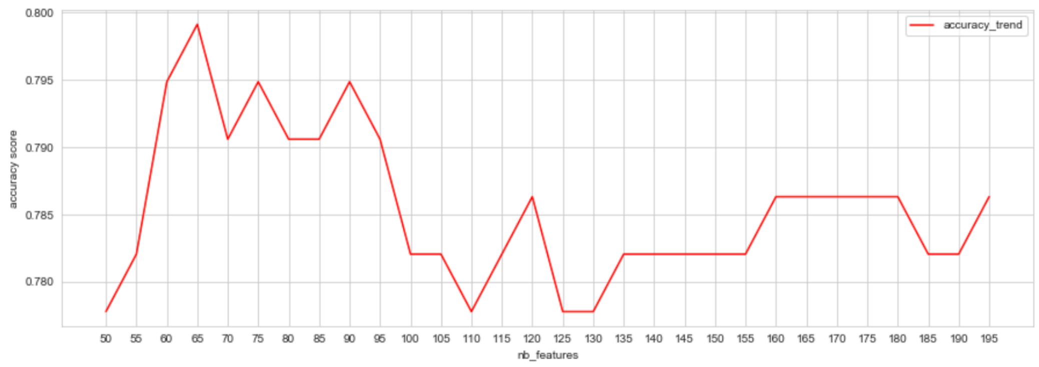
766 rows × 7 columns

  b. **Exploring and defining feature engineering**

     We can choose some of words that have a relatively higher frequency in our article signed with a specific topic as our feature space. There is no doubt that each distinct word should be treated as a feature. And we can utilize the same approach to all articles for each topic. And then, we acquire 10 bags of the most frequent words occurred in articles corresponding to each of our topics. And we can combine these words together and form a set of words, finally we treat words in this set as our feature vectors in order to make classifications.

     In addition, we need to find out and explore the most suitable number of feature size. The line chart below shows the accuracy value using Multinomial Naïve Bayes Model over the increasing number of features we chose.

     After analyzing, we get a relatively useful conclusion that choosing the top frequent 65 words works the best by using Multinomial Model compared with others. We intend to choose the top frequent 65 words to decide our feature space.



When we choose top 65 frequent words to discover our feature space, we get the following words as our features eventually:

Number of features: 248

Feature words and details is in below picture:

```
words feature numbers :  248 gives the best performance!
our words feature is :  {'term', 'increas', 'yen', 'match', 'beat', 'tuesday', 'result', 'monet', 'report', 'art', 's
ale', 'season', 'add', 'open', 'head', 'museum', 'point', 'london', 'foreign', 'televis', 'wednesday', 'polit', 'se
t', 'econom', 'final', 'domest', 'sharehold', 'research', 'round', 'expand', 'friday', 'newsroom', 'food', 'late', 's
mok', 'billion', 'author', 'clon', 'peopl', 'produc', 'austral', 'republ', 'summit', 'divid', 'human', 'brazil', 'def
end', 'inflat', 'money', 'financ', 'music', 'women', 'team', 'direct', 'champ', 'polic', 'stat', 'cigaret', 'export',
'year', 'top', 'back', 'currenc', 'quot', 'agree', 'presid', 'europ', 'base', 'demand', 'initial', 'bond', 'exchang',
'weapon', 'compan', 'trad', 'short', 'meet', 'care', 'deal', 'health', 'month', 'gmt', 'leagu', 'rise', 'forc', 'rai
s', 'hit', 'system', 'nation', 'man', 'alli', 'yield', 'make', 'club', 'act', 'govern', 'peso', 'nato', 'european',
'due', 'pow', 'met', 'monday', 'russia', 'bid', 'matur', 'anal', 'countr', 'child', 'buy', 'franc', 'tobacc', 'long',
'commit', 'test', 'socc', 'industr', 'plan', 'list', 'gener', 'problem', 'told', 'fami', 'start', 'reserv', 'heart',
'time', 'clinton', 'dollar', 'home', 'patient', 'titl', 'injur', 'interest', 'capit', 'die', 'kong', 'win', 'cent',
'war', 'secret', 'auction', 'record', 'friend', 'program', 'found', 'show', 'percent', 'part', 'develop', 'unit', 'br
itain', 'stock', 'public', 'cup', 'talk', 'issu', 'day', 'import', 'germ', 'weak', 'fix', 'lead', 'court', 'firm', 'g
ame', 'work', 'americ', 'goal', 'car', 'memb', 'week', 'rate', 'city', 'strong', 'made', 'suspend', 'control', 'mosco
w', 'sell', 'call', 'group', 'thursday', 'russian', 'pictur', 'pct', 'minut', 'bank', 'race', 'hk', 'expect', 'high',
'treat', 'hong', 'level', 'vict', 'run', 'japan', 'launch', 'case', 'stand', 'award', 'sunday', 'bours', 'arm', 'mill
ion', 'mark', 'world', 'clos', 'aver', 'hold', 'german', 'trial', 'scor', 'rule', 'mexic', 'study', 'early', 'marke
t', 'side', 'diseas', 'treasur', 'offic', 'fall', 'total', 'film', 'england', 'army', 'south', 'releas', 'invest', 'b
ill', 'troop', 'secur', 'minist', 'includ', 'overnight', 'janu', 'milit', 'afric', 'shar', 'low', 'end', 'drug', 'pla
y', 'live', 'pric', 'canad'}
```

c. Creating and training models initially and choosing a metric standard

To start with, there is a fact that this is a typical **multi-class problem**, which can be evaluated by the commonly used **multi-class log loss**.

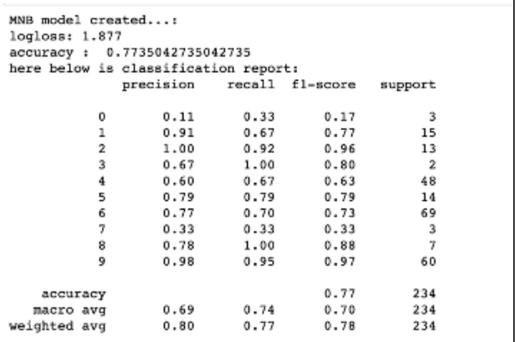
For multiple classification, we can use log loss function (multiple-class version), the formula is:

$$-\sum_{i}^{N}\sum_{j}^{M} y_{ij}log(p_{ij})$$

Where N corresponds to the number of samples or the number of instances input, and i to a specific sample or instance; M represents the number of possible classes of our sample, and j represents a certain label; y_{ij} is the label that belongs to category j for a sample i, and i only belongs to one category; And then p_{ij} is the probability that sample i is predicted to be classified as j.

From the expression, we could obtain some information that log loss is intended to punish misclassification, and for a good multi-classification model, the estimating value of log loss should be as low as possible.

Once we determine our metric method, we begin to create our initial models:



```
MNB model created...:
logloss: 1.877
accuracy :  0.7735042735042735
here below is classification report:
              precision    recall  f1-score   support

           0       0.11      0.33      0.17         3
           1       0.91      0.67      0.77        15
           2       1.00      0.92      0.96        13
           3       0.67      1.00      0.80         2
           4       0.60      0.67      0.63        48
           5       0.79      0.79      0.79        14
           6       0.77      0.70      0.73        69
           7       0.33      0.33      0.33         3
           8       0.78      1.00      0.88         7
           9       0.98      0.95      0.97        60

    accuracy                           0.77       234
   macro avg       0.69      0.74      0.70       234
weighted avg       0.80      0.77      0.78       234
```

- As we can see, the initial MultinomialNB model does not work so well in terms of the value of logloss (logloss value is around 1.8). By the way, in the process of establishing initial model, we can assign a set of parameters. The "alpha" term is set to be 1.0 by default.

d. tuning hyper-parameter

Parameter adjustment guide: naive bayesian parameter adjustment generally adjusts alpha, smooth parameter, the smaller the value, the easier overfitting, the larger the value, easy to cause underfitting.

We use param_grid to represent our alpha value, which is set range with 6 different. values: 0.001, 0.01, 0.1, 1, 10 and 100

And we need to figure out which value gives the best performance of our model. That is, we will find the most suitable value which makes log loss being lowest.

The ideal value of alpha is 100.

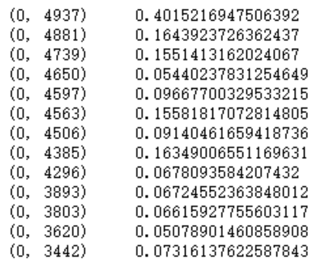
```
Best parameters set:
        nb__alpha: 100
```

Now, we use parameter alpha=100.

We get our log loss around 1.3, which gives a better performance than the model. with alpha=1.0.

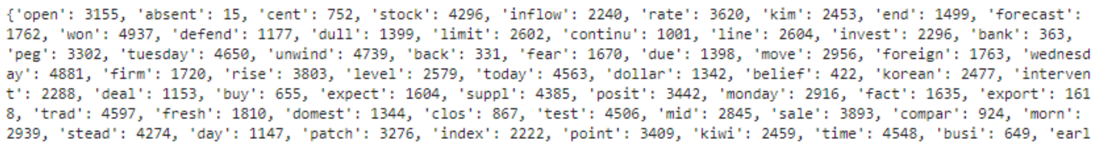
- **Method 3: TF-IDF+SVC**

a. TF-IDF



```
(0, 4937)     0.4015216947506392
(0, 4881)     0.1643923726362437
(0, 4739)     0.1551413162024067
(0, 4650)     0.05440237831254649
(0, 4597)     0.09667700329533215
(0, 4563)     0.15581817072814805
(0, 4506)     0.09140461659418736
(0, 4385)     0.16349006551169631
(0, 4296)     0.0678093584207432
(0, 3893)     0.06724552363848012
(0, 3803)     0.06615927755603117
(0, 3620)     0.05078901460858908
(0, 3442)     0.07316137622587843
```

During the preprocessing step, the texts should be converted to numerical feature vectors. One common approach for extracting features is called TF-IDF.



```
{'open': 3155, 'absent': 15, 'cent': 752, 'stock': 4296, 'inflow': 2240, 'rate': 3620, 'kim': 2453, 'end': 1499, 'forecast':
1762, 'won': 4937, 'defend': 1177, 'dull': 1399, 'limit': 2602, 'continu': 1001, 'line': 2604, 'invest': 2296, 'bank': 363,
'peg': 3302, 'tuesday': 4650, 'unwind': 4739, 'back': 331, 'fear': 1670, 'due': 1398, 'move': 2956, 'foreign': 1763, 'wednesd
ay': 4881, 'firm': 1720, 'rise': 3803, 'level': 2579, 'today': 4563, 'dollar': 1342, 'belief': 422, 'korean': 2477, 'interven
t': 2288, 'deal': 1153, 'buy': 655, 'expect': 1604, 'suppl': 4385, 'posit': 3442, 'monday': 2916, 'fact': 1635, 'export': 161
8, 'trad': 4597, 'fresh': 1810, 'domest': 1344, 'clos': 867, 'test': 4506, 'mid': 2845, 'sale': 3893, 'compar': 924, 'morn':
2939, 'stead': 4274, 'day': 1147, 'patch': 3276, 'index': 2222, 'point': 3409, 'kiwi': 2459, 'time': 4548, 'busi': 649, 'earl
```

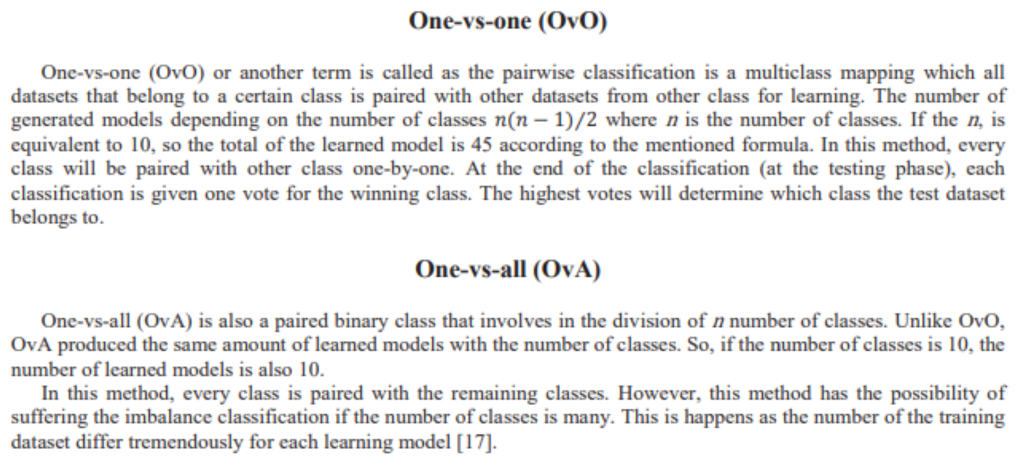
Therefore, we transform the data to vectorized numeric data. These contain for each row a list of unique integer number and its associated importance as calculated by TF-IDF.

b. support-vector machines (SVM)

After reading some articles and papers, I found that there are two main methods. about SVM that can achieve multiple article classifications:

1. One- Versus -One SVM for Multi-Class Classification (OVO)

2. One-Versus-All SVM for Multi-Class Classification (OVA)[1]



One-vs-one (OvO)

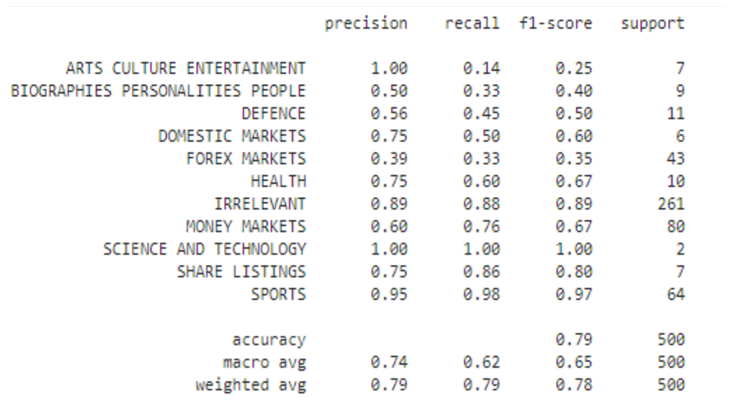
One-vs-one (OvO) or another term is called as the pairwise classification is a multiclass mapping which all datasets that belong to a certain class is paired with other datasets from other class for learning. The number of generated models depending on the number of classes $n(n-1)/2$ where n is the number of classes. If the n, is equivalent to 10, so the total of the learned model is 45 according to the mentioned formula. In this method, every class will be paired with other class one-by-one. At the end of the classification (at the testing phase), each classification is given one vote for the winning class. The highest votes will determine which class the test dataset belongs to.

One-vs-all (OvA)

One-vs-all (OvA) is also a paired binary class that involves in the division of n number of classes. Unlike OvO, OvA produced the same amount of learned models with the number of classes. So, if the number of classes is 10, the number of learned models is also 10.

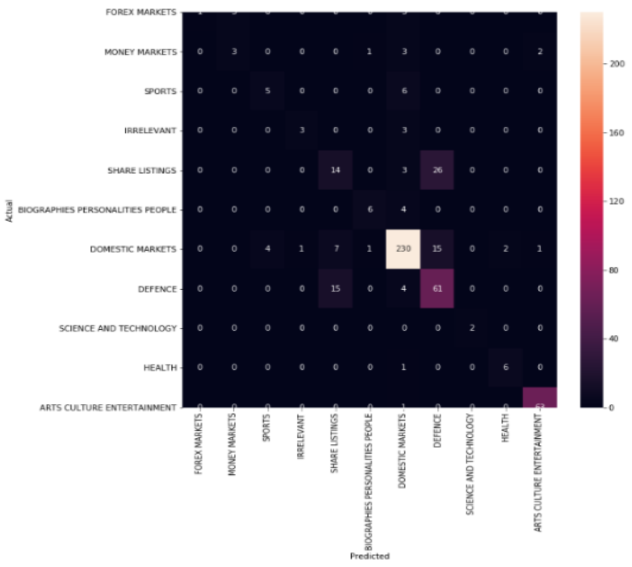
In this method, every class is paired with the remaining classes. However, this method has the possibility of suffering the imbalance classification if the number of classes is many. This is happens as the number of the training dataset differ tremendously for each learning model [17].

In order to implementation of OvO, I choose the Support Vector Classification: SVC(kernel=’ linear’), because the classification speed of the linear classifier is faster than that of the non-linear one. Then it can be optimized in the subsequent hyperparameter tuning to improve its accuracy.



```
                               precision    recall  f1-score   support

     ARTS CULTURE ENTERTAINMENT       1.00      0.14      0.25         7
BIOGRAPHIES PERSONALITIES PEOPLE      0.50      0.33      0.40         9
                        DEFENCE       0.56      0.45      0.50        11
               DOMESTIC MARKETS       0.75      0.50      0.60         6
                  FOREX MARKETS       0.39      0.33      0.35        43
                         HEALTH       0.75      0.60      0.67        10
                     IRRELEVANT       0.89      0.88      0.89       261
                  MONEY MARKETS       0.60      0.76      0.67        80
         SCIENCE AND TECHNOLOGY       1.00      1.00      1.00         2
                 SHARE LISTINGS       0.75      0.86      0.80         7
                         SPORTS       0.95      0.98      0.97        64

                       accuracy                           0.79       500
                      macro avg       0.74      0.62      0.65       500
                   weighted avg       0.79      0.79      0.78       500
```

c. Cross-validated grid-search

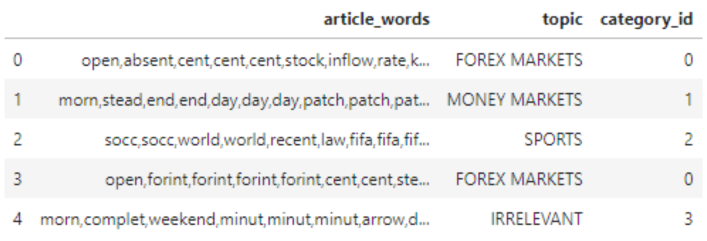


Now we get the classification report and the confusion matrix on the development set. After that, I use the cross-validated grid-search over a parameter grid to optimize the hyperparameters:

```
Best score for training data: 0.7812222222222223

Best C: 1

Best Kernel: linear

Best Gamma: auto_deprecated

Best decision_function_shape: ovo
```

However, considering the imbalance of data distribution just mentioned, OvO may lead to such problem more serious. Therefore, I also attempt the OvA(OvR) method by LinearSVC(). The difference between SVC(kernel=’ linear’) and LinearSVC() is their implementation. SVC implemented in terms of libsvm, but LinearSVC adapts to liblinear. In fact, for text classification problems, the number of samples and features are very large. LinearSVC implemented with liblinear maybe a better choice compared to another method in this project.

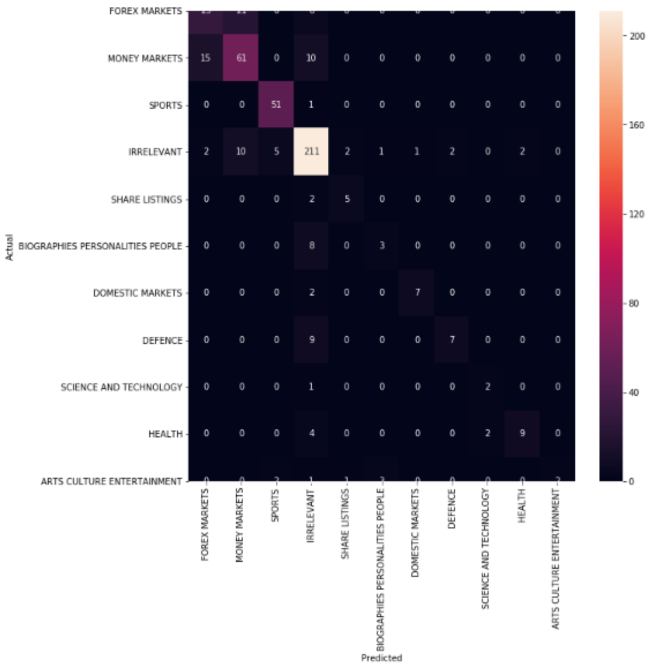
Before fitting a new model, I combined previous experience to reprocess the. model. Firstly, adding a column encoding the ‘topic’ as an integer because categorical. Variables are often better represented by integers than strings, and then create a couple of dictionaries for future use.



	article_words	topic	category_id
0	open,absent,cent,cent,cent,stock,inflow,rate,k...	FOREX MARKETS	0
1	morn,stead,end,end,day,day,day,patch,patch,pat...	MONEY MARKETS	1
2	socc,socc,world,world,recent,law,fifa,fifa,fif...	SPORTS	2
3	open,forint,forint,forint,forint,cent,cent,ste...	FOREX MARKETS	0
4	morn,complet,weekend,minut,minut,minut,arrow,d...	IRRELEVANT	3

In next step, extracting features by TF-IDF again.

Then, evaluating model and plotting the confusion matrix.



The vast majority of the predictions end up on the diagonal (predicted label = actual label), where we want them to be. Through there are a bit differences between the confusion matrix of SVC(kernel=’ linear’) and LinearSVC(), they have similar performance of accuracy (about 0.78)on the development set, probably because there is no large sample data.

However, considering that SVC has been tuned, SVC(kernel=’ linear’) was finally selected.

- **Method 4: TF-IDF + SGDClassifier**

a. Feature extracting

We used CountVectorizer to map terms to features. It converts a collection of text documents to a matrix of token counts. This implementation produces a sparse representation of the counts using scipy.sparse.csr_matrix. Their ngram_range is set to (1,1) since the dataset provided does not take the order of words into consideration. Then use TfidfTransformer to transform the count matrix to a normalized tf-idf representation.The goal of using TF-IDF instead of the raw frequencies of occurrence of a token in a given document is to scale down the impact of tokens that occur very frequently in a given corpus and that are hence empirically less informative than features that occur in a small fraction of the training corpus. Applying CountVectorizer and TfidfTransformer is just the same with using TfidfVectorizer. (We use them separately just to make the math background more understandable.)

b. **Classifier**

The classifier we used is SGDClassifier , which is also a linear classifiers with SGD training.This estimator implements regularized linear models with stochastic gradient descent (SGD) learning: the gradient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing strength schedule. The loss function is set to 'modified_huber' , which is another smooth loss that brings tolerance to outliers as well as probability estimates.

- **Method 5: CountVectorize + DecisionTree**

a. **CountVectorize**

We use the tool that has been concluded in the sklearn to count words and use it as a feature. In this program, to avoid errors, words in testing articles also have been included.

b. **DecisionTree**

We choose DecisionTree as a classifier, which is easy to implement and use. This model can predict a categorical output from categorical and/or real inputs, or regression.

- **Comparations**

Comparations of Accuracy

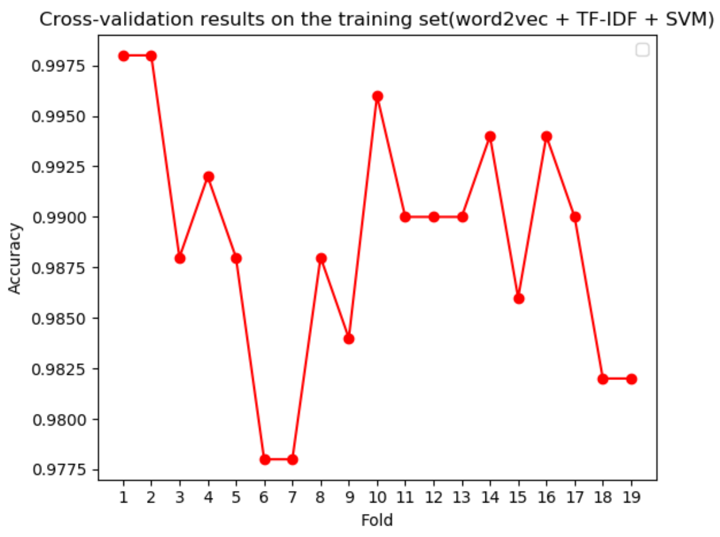
Model name	Accuracy
word2vec + TF-IDF+SVM	0.95
Multinomial Naïve Bayes Model	0.77
TF-IDF + SVC	0.77
TF-IDF + SGDClassifier	0.77
CountVectorize + DecisionTree	0.71

Through the table above, we can easily get that model 1(word2vec + TF-IDF + SVM) has the best accuracy and therefore, we choose this one to get our final result.

Results

- word2vec + TF-IDF+SVM

Cross-validation results



Final results

word2vec + TF-IDF+SVM			
Topic name	**Precision**	**Recall**	**F1**
ARTS CULTURE ENTERTAINMENT	1.00	1.00	1.00
BIOGRAPHIES PERSONALITIES PEOPLE	1.00	0.94	0.97
DEFENCE	0.92	1.00	0.96
DOMESTIC MARKETS	1.00	1.00	1.00
FOREX MARKETS	0.83	0.82	0.82
HEALTH	0.93	1.00	0.96
MONEY MARKETS	0.88	0.90	0.89
SCIENCE AND TECHNOLOGY	0.67	1.00	0.80
SHARE LISTINGS	1.00	0.78	0.88
SPORTS	1.00	1.00	1.00

Final article recommendations

word2vec + TF-IDF+SVM				
Topic name	**Suggested articles**	**Precision**	**Recall**	**F1**
ARTS CULTURE ENTERTAINMENT	9952 9703 9834	1.00	1.00	1.00
BIOGRAPHIES PERSONALITIES PEOPLE	9940 9758 9854 9878 9983 9768 9581 9988 9645 9526	1.00	0.94	0.97
DEFENCE	9559 9770 9987 9773 9616 9576 9706 9842 9670 9713	0.92	1.00	0.96
DOMESTIC MARKETS	9994 9796	1.00	1.00	1.00
FOREX MARKETS	9961 9965 9718 9725 9530 9588 9975 9551 9977 9837	0.83	0.82	0.82
HEALTH	9873 9661 9947 9810 9887 9621 9807 9735	0.93	1.00	0.96

	9833 9978			
MONEY MARKETS	9618 9516 9769 9998 9820 9828 9835 9691 9967 9860	0.88	0.90	0.89
SCIENCE AND TECHNOLOGY	9722 9929	0.67	1.00	0.80
SHARE LISTINGS	9601 9518 9562 9999 9972 9654 9667 9867 9666	1.00	0.78	0.88
SPORTS	9981 9573 9580 9657 9569 9541 9920 9663 9738 9574	1.00	1.00	1.00

Discussion

- Method 1(word2vec + TF-IDF + SVM) has a higher accuracy over the 3 methods. Compared with Method 3(TF-IDF + SVM). It may be due to the use of word2vec.
  In Method 2(Multinomial Naïve Bayes), it shows the relation of the number of features and the final accuracy and therefore decide the number of features.
- $Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$, $Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$,

  $$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

  From the formula above, we can see F1 is more appropriate. This is because it contains 4 conditions: TP, FP, FN and FP($= Overall - TP - FP - FN$).
- According to the learning in this project, the single classification approach does not perform well in every case, for example, SVM cannot identify categories of words correctly when the texts are in cross zones of multi-categories without word2vec. Therefore, combining the advantages of each model through ensemble learning may significantly improve accuracy, such as SVM-kNN.
  If continuing this project, we can use LSTM (RNN) or CNN instead of SVM, they may get better performance in this project.
  F1 Score might be a better measure to use if we need to seek a balance between Precision and Recall and there is an uneven class distribution (large number of Actual Negatives).

Conclusion

Through 5 methods given above, we can predict any article in the test data belongs to which label. The accuracy of 3 methods are not equal and method 1 has a high accuracy which is more than 90%. Through this project, we have learned the application of many kinds of models and how to improve the accuracy.

Reference

[1] Raziff, A.R.A., Sulaiman, M.N., Mustapha, N. and Perumal, T., 2017, October. Single classifier, OvO, OvA and RCC multiclass classification method in handheld based smartphone gait identification. In AIP Conference Proceedings (Vol. 1891, No. 1, p. 020009). AIP Publishing LLC

[2] Kaggle.com. 2020. SVM For Multiclass Classification. [online] Available at: <https://www.kaggle.com/pranathichunduru/svm-for-multiclass-classification> [Accessed 24 April 2020].

[3] Medium. 2020. Multi-Class Text Classification With Scikit-Learn. [online] Available at: <https://towardsdatascience.com/multi-class-text-classification-with-scikit-learn-12f1e60e0a9f> [Accessed 24 April 2020].

[4] Medium. 2020. Multi-Class Text Classification With Scikit-Learn. [online] Available at: <https://towardsdatascience.com/multi-class-text-classification-with-scikit-learn-12f1e60e0a9f> [Accessed 24 April 2020].

[5] GitHub. 2020. Miguelfzafra/Latest-News-Classifier. [online] Available at: <https://github.com/miguelfzafra/Latest-News-Classifier/blob/master/0.%20Latest%20News%20Classifier/04.%20Model%20Training/09.%20MT%20-%20MultinomialNB.ipynb> [Accessed 26 April 2020]. [6] miguelfzafra/Latest-News-Classifier

[6] Kusner, M., Sun, Y., Kolkin, N. and Weinberger, K., 2015, June. From word. embeddings to document distances. In International conference on machine learning (pp. 957-966).

[7] Martineau, J.C. and Finin, T., 2009, March. Delta tfidf: An improved feature space for sentiment analysis. In Third international AAAI conference on weblogs and social media.

[8] Martineau, J.C. and Finin, T., 2009, March. Delta tfidf: An improved feature space for sentiment analysis. In Third international AAAI conference on weblogs and social media.