Title page:

Title of project: Choosing the best way for Text Classification in Python3
Group name: Mamamia
Group members: Zhaokun Su(z5235878)

Introduction:

We are given a specific dataset and we have already got all labels for 10 different topics, and we can assign them from number 1 to 10.

For training data set: we get 9500 new articles (9500 instances/observations), and some of them (about 48%) are assigned to label “irrelevant”, which means these articles do not have an explicit topic for our training data.

For testing data set: we get 500 new articles that need to be classified correctly.

Note:
1. the distribution of articles over topics for training data set and tesing data set is not uniform. It means that maybe some of topics (labels) gain more popularity than other topics.
2. But from perspective of choosing our data, the distribution of training data and testing data over topics can be treated as similar, which means that they have the same distribution for new articles.

Our aim is to try to classcify those news articles to correct labels using some well-performed models and obtain their performance metrics for each finding model.

First of all, we should do some data cleaning for our data set. We should firstly identify our features, do some data processing and get some useful data to utilise. After that, we select several models to be fitted. By using different metrics and evaluation approaches, we will obtain a comprehensive assessment for our problem and test the best model we chose to make the final decision (suggesting up to 10 new articles per topic from the test set).

Exploratory data analysis:

Since there is one class label not related to our classcification, we need to remove that “IIRELEVANT” label before we do our analysis.

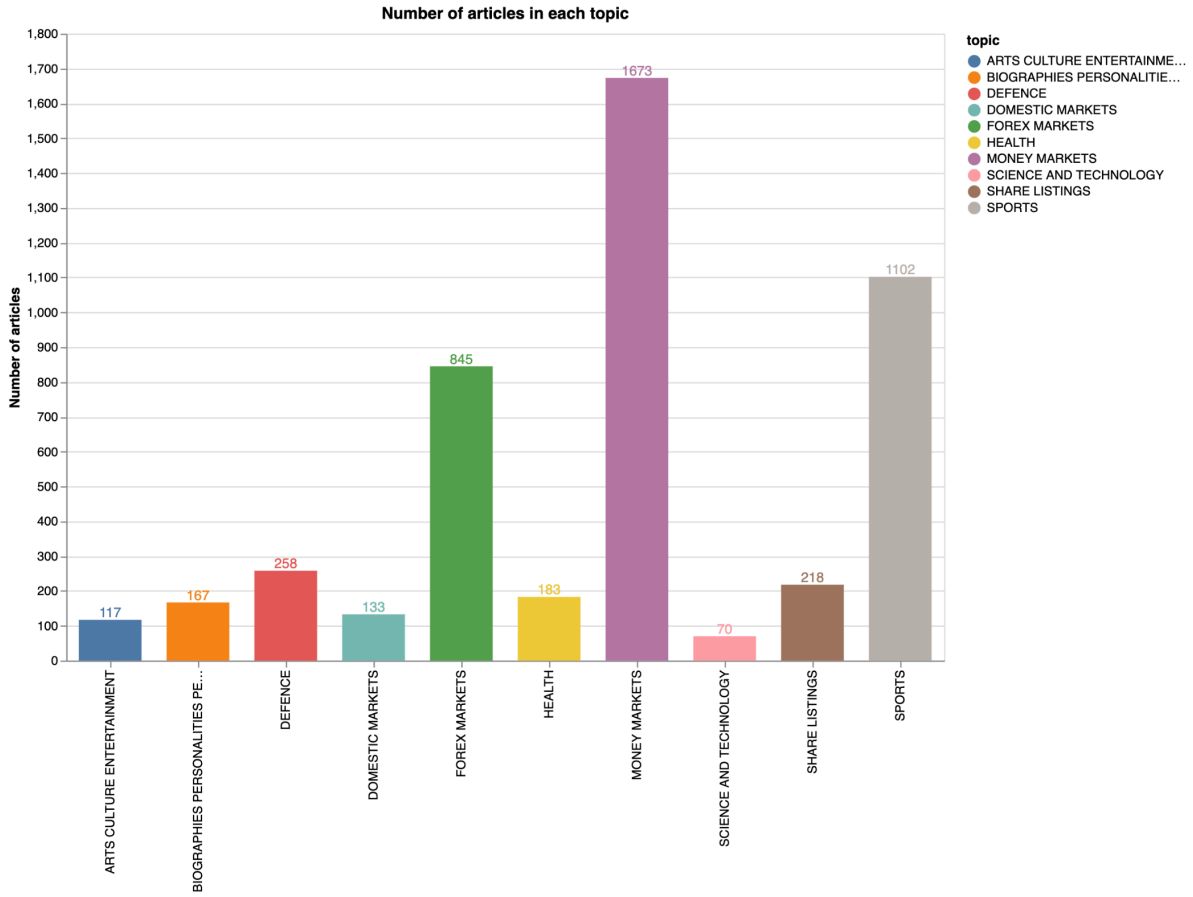
```
df = pd.read_csv('training.csv')
df.head()
```

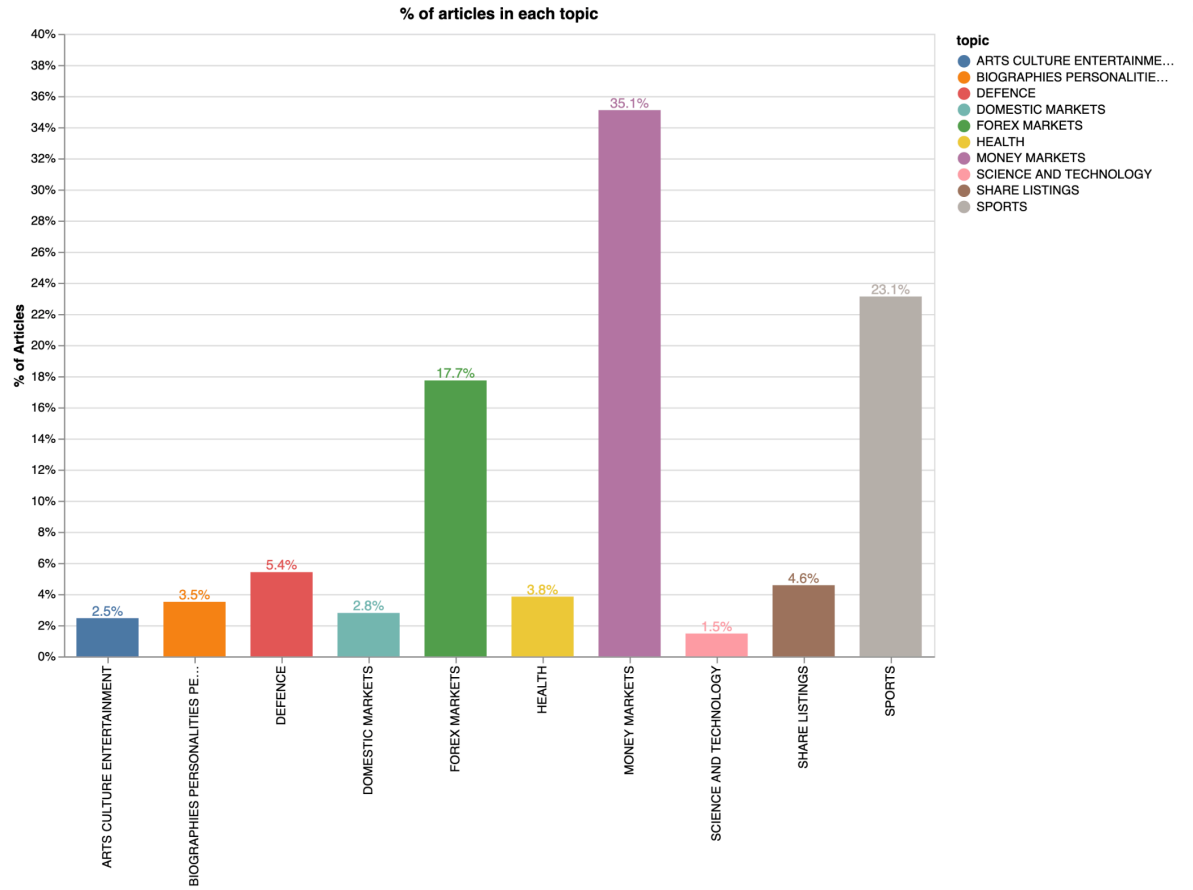
	article_number	article_words	topic
0	1	open,absent,cent,cent,cent,stock,inflow,rate,k...	FOREX MARKETS
1	2	morn,stead,end,end,day,day,day,patch,patch,pat...	MONEY MARKETS
2	3	socc,socc,world,world,recent,law,fifa,fifa,fif...	SPORTS
3	4	open,forint,forint,forint,forint,cent,cent,ste...	FOREX MARKETS
4	5	morn,complet,weekend,minut,minut,minut,arrow,d...	IRRELEVANT

```
df = df.loc[df['topic']!='IRRELEVANT']
df.head()
```

	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
5	6	regist,equal,stock,stock,city,city,period,issu...	SHARE LISTINGS

One thing we need to consider before we develop a classification model is if the classes in the data set is of balance, the meaning of this issue is about whether the original data set has a relatively approximately equal radio for each label. The bar chart below shows the number of new articles of different topics. Once getting this indicator, we can calculate the corresponding percentage of each category.
The charts below show details about the radio of different classes in the training set, and this data is about class prior probabilities for each class


Number of articles in each topic
Number of articles
1,800
1,700
1,600
1,500
1,400
1,300
1,200
1,100
1,000
900
800
700
600
500
400
300
200
100
0
117
167
258
133
845
183
1673
70
218
1102
ARTS CULTURE ENTERTAINMENT
BIOGRAPHIES PERSONALITIES PE...
DEFENCE
DOMESTIC MARKETS
FOREX MARKETS
HEALTH
MONEY MARKETS
SCIENCE AND TECHNOLOGY
SHARE LISTINGS
SPORTS
topic
ARTS CULTURE ENTERTAINME...
BIOGRAPHIES PERSONALITIE...
DEFENCE
DOMESTIC MARKETS
FOREX MARKETS
HEALTH
MONEY MARKETS
SCIENCE AND TECHNOLOGY
SHARE LISTINGS
SPORTS


% of articles in each topic
% of Articles
40%
38%
36%
34%
32%
30%
28%
26%
24%
22%
20%
18%
16%
14%
12%
10%
8%
6%
4%
2%
0%
2.5%
3.5%
5.4%
2.8%
17.7%
3.8%
35.1%
1.5%
4.6%
23.1%
ARTS CULTURE ENTERTAINMENT
BIOGRAPHIES PERSONALITIES PE...
DEFENCE
DOMESTIC MARKETS
FOREX MARKETS
HEALTH
MONEY MARKETS
SCIENCE AND TECHNOLOGY
SHARE LISTINGS
SPORTS
topic
ARTS CULTURE ENTERTAINME...
BIOGRAPHIES PERSONALITIE...
DEFENCE
DOMESTIC MARKETS
FOREX MARKETS
HEALTH
MONEY MARKETS
SCIENCE AND TECHNOLOGY
SHARE LISTINGS
SPORTS

Methodology And Results:

Since this project can be regarded as a text classification problem, so in this problem, I decide to use **Multinomial Naïve Bayes Model** and **Multinomial Logistic Regression Model**. Before we establish our models, first of all, we need to do follow three steps:

Step 1: Data Cleaning and Data Pre-processing

1. our original data contains all categorical data (including class "IRRELEVANT"), "IRRELEVANT" class is not relevant to our objective classes, so we need to clean our data by **removing all records that are "NOT RELATED" to our classes**.

```
# remove irrelevant items
df_train = df_train.loc[df_train['topic'] != 'IRRELEVANT']
df_test = df_test.loc[df_test['topic'] != 'IRRELEVANT']
```

2. after we droping all observations which belong to IRRELEVANT class, we need to **add topic codes for each topic and generate topic codes for each topic**. We generate our topic codes as below: we have 10 objective topics, so we use integer number 1 to 10 to represent our classes.

```
# add topic codes for each topic
# generate topic codes for each topic
topic_codes = {
    'ARTS CULTURE ENTERTAINMENT': 1,
    'BIOGRAPHIES PERSONALITIES PEOPLE': 2,
    'DEFENCE': 3,
    'DOMESTIC MARKETS': 4,
    'FOREX MARKETS': 5,
    'HEALTH': 6,
    'MONEY MARKETS': 7,
    'SCIENCE AND TECHNOLOGY': 8,
    'SHARE LISTINGS': 9,
    'SPORTS': 10
}
```

3. Eventually, we can process our data set to make more sense. Our aim is to classify new articles to correct class. When it comes to the content of our new article, it shows 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. Since it is all about topics in our daily life, we can **remove all "stop words"** ("stop words" means they are not meaningful for our classification model) in advance. We have obtained some "stop words" from specific package in our Python file like this:

```
# get stopwords
nltk.download('stopwords')
stop_words = list(stopwords.words('english'))  # get stop words
```

And we remove them in our function in the first step.

Finally, we can make a comparasion before and after processing our data sets:

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

Step 2: Exploring and defining our feature engineering

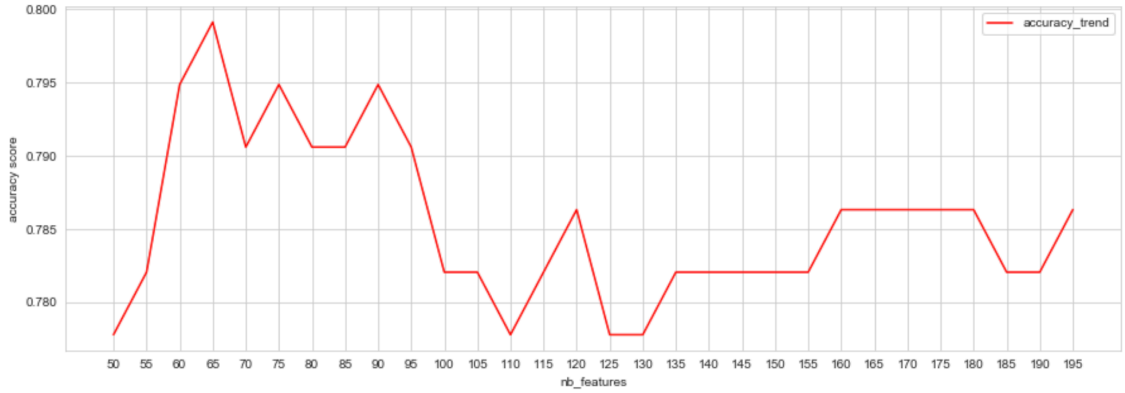
After we have pre-processed our data sets, how should we make a decision about our feature selection?

In step 2, naturally, by our original recognition, there is no doubt that each distinct word should be treated as a feature. But it is not enough, the reason is that we get thousands of words in even one article, so we derive an idea that:

We can choose some of words that have a relatively higher frequency in our article signed with a specific topic as our feature space. And we can utilise 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 analysing, 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'}
```

Step 3: 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 fuction (multiple-class version), the formular is in below:

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

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:
a. For **Multinomial Naïve Bayes Model**:

```
training_process_for_MultinomialNB_model(X_train, y_train, X_test, y_test, alpha=1.0)
# training_process_for_MultinomialLR_model(X_train, y_train, X_test, y_test, C=1.0)
```

```
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.

b. For **Multinomial Logistic Regression Model**:

```
# training_process_for_MultinomialNB_model(X_train, y_train, X_test, y_test, alpha=1.0)
training_process_for_MultinomialLR_model(X_train, y_train, X_test, y_test, C=1.0)
```

```
MLR model created...:
logloss: 0.655
accuracy :  0.7435897435897436
here below is classification report:
              precision    recall  f1-score   support

           0       0.25      0.33      0.29         3
           1       0.82      0.60      0.69        15
           2       1.00      0.92      0.96        13
           3       1.00      1.00      1.00         2
           4       0.54      0.40      0.46        48
           5       0.69      0.79      0.73        14
           6       0.65      0.77      0.70        69
           7       0.25      0.33      0.29         3
           8       0.88      1.00      0.93         7
           9       0.98      0.98      0.98        60

    accuracy                           0.74       234
   macro avg       0.71      0.71      0.70       234
weighted avg       0.74      0.74      0.74       234
```

As we can see, the initial MultinomialNB model works not bad in terms of logloss value (logloss value is around 1.8). By the way, in the process of establishing initial model, we can assign a set of parameters. The “C” term is set to be 1.0 by default.

Step 3: tuning hyper-parameters in order to require the best model
a. tuning parameter for **Multinomial Naïve Bayes Model**:

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.

```
# create score fuction
mll_scorer = metrics.make_scorer(multiclass_logloss, greater_is_better=False, needs_proba=True)
nb_model = MultinomialNB()

# create pipeline
clf = pipeline.Pipeline([('nb', nb_model)])

# search parameters
param_grid = {'nb__alpha': [0.001, 0.01, 0.1, 1, 10, 100]}

# Grid Search Model Initialization
model = GridSearchCV(estimator=clf, param_grid=param_grid, scoring=mll_scorer,
                                verbose=10, n_jobs=-1, iid=True, refit=True, cv=6)

# fit Grid Search Model
model.fit(X_train, y_train)
# print("Best score: %0.3f" % model.best_score_)
print("Best parameters set:")
best_parameters = model.best_estimator_.get_params()
for param_name in sorted(param_grid.keys()):
    print("\t%s: %r" % (param_name, best_parameters[param_name]))
```

We use param_grid to represent our alpha value, which is set range with 6 different values:

i. 0.001

ii. 0.01

iii. 0.1

iv. 1

v. 10

vi. 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, assign this to our model and retrain it:

```
MNB model created...:
logloss: 1.318
accuracy :  0.7478632478632479
here below is classification report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         3
           1       0.78      0.47      0.58        15
           2       0.71      0.92      0.80        13
           3       0.67      1.00      0.80         2
           4       0.57      0.50      0.53        48
           5       0.72      0.93      0.81        14
           6       0.69      0.74      0.71        69
           7       0.00      0.00      0.00         3
           8       0.67      0.86      0.75         7
           9       0.97      1.00      0.98        60

    accuracy                           0.75       234
   macro avg       0.58      0.64      0.60       234
weighted avg       0.73      0.75      0.73       234
```

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

a. tuning parameter for **Multinomial Logistic Regression Model**:
Similarly, we can use the same way to obtain our best C parameter in Multinomial Logistic Regression Model.

```
lr_model = LogisticRegression(solver='lbfgs',multi_class='multinomial')

# create pipeline
clf = pipeline.Pipeline([('lr', lr_model)])

# search parameters
param_grid = {'lr__C': [0.01, 0.1, 1.0, 10, 100]}

# Grid Search Model Initialization
model = GridSearchCV(estimator=clf, param_grid=param_grid, scoring=mll_scorer,
                                 verbose=10, n_jobs=-1, iid=True, refit=True, cv=6)

# fit Grid Search Model
model.fit(X_train, y_train)
print("Best score: %0.3f" % model.best_score_)
print("Best parameters set:")
best_parameters = model.best_estimator_.get_params()
for param_name in sorted(param_grid.keys()):
    print("\t%s: %r" % (param_name, best_parameters[param_name]))
```

The result turns out to be 0.1

```
Best parameters set:
        lr__C: 0.1
```

Now, we use parameter C=0.1, assign this to our model and retrain it:

```
MLR model created...:
logloss: 0.550
accuracy :  0.7435897435897436
here below is classification report:
              precision    recall  f1-score   support

           0       0.20      0.33      0.25         3
           1       0.90      0.60      0.72        15
           2       1.00      0.92      0.96        13
           3       1.00      1.00      1.00         2
           4       0.53      0.35      0.42        48
           5       0.71      0.86      0.77        14
           6       0.64      0.78      0.70        69
           7       0.33      0.33      0.33         3
           8       0.88      1.00      0.93         7
           9       0.98      0.98      0.98        60

    accuracy                           0.74       234
   macro avg       0.72      0.72      0.71       234
weighted avg       0.75      0.74      0.74       234
```

We get our log loss around 0.5, which gives a better performance than the model with C=1.0.

Finally, results are acquired:

TABLE_1 (for model 1):

Topic name	Suggested articles	Precision	Recall	F1
ARTS CULTURE ENTERTAINMENT	9604, 9789, 9834, 9878	0.25	0.33	0.29
BIOGRAPHIES PERSONALITIES	9582, 9645, 9758, 9768, 9783, 9830, 9933, 9983, 9988	0.89	0.53	0.67
DEFENCE	9559, 9576, 9616, 9670, 9706, 9713, 9721, 9739, 9770, 9773, 9987	1.00	0.92	0.96
DOMESTIC MARKETS	9796, 9994	1.00	1.00	1.00
FOREX MARKETS	9506, 9577, 9671, 9711, 9743, 9851, 9852, 9855, 9875, 9893, 9894	0.56	0.38	0.45
HEALTH	9621, 9661, 9703, 9735, 9807, 9810, 9833, 9873, 9887, 9911, 9929	0.67	0.86	0.75
MONEY MARKETS	9516, 9553, 9586, 9589, 9618, 9691, 9737, 9751, 9755, 9769, 9863	0.65	0.80	0.71
SCIENCE AND TECHNOLOGY	9617, 9722, 9982	0.33	0.33	0.33
SHARE LISTINGS	9518, 9562, 9581, 9601, 9654, 9667, 9972, 9999	0.88	1.00	0.93
SPORTS	9568, 9569, 9752, 9760, 9774, 9787, 9832, 9848, 9857, 9922, 9997	0.97	0.98	0.98

TABLE 2 (for model 2):

Topic name	Suggested articles	Precision	Recall	F1
ARTS CULTURE ENTERTAINMENT	9526,9604, 9789	0.25	0.33	0.29
BIOGRAPHIES PERSONALITIES	9604, 9722, 9758, 9768, 9854, 9878, 9940, 9983, 9988	0.78	0.47	0.58
DEFENCE	9559, 9576, 9616, 9706, 9713, 9721, 9739, 9770, 9773, 9783, 9987	0.71	0.92	0.80
DOMESTIC MARKETS	9796, 9833, 9994	0.67	1.00	0.80
FOREX MARKETS	9572, 9584, 9625, 9693, 9704, 9711, 9743, 9748, 9823, 9875, 9902	0.56	0.48	0.52
HEALTH	9621, 9661, 9703, 9735, 9807, 9810, 9873, 9887, 9911, 9929, 9982	0.76	0.93	0.84
MONEY MARKETS	9516, 9553, 9618, 9737, 9755, 9769, 9816, 9835, 9863, 9901, 9967	0.68	0.74	0.71
SCIENCE AND TECHNOLOGY		0.00	0.00	0.00
SHARE LISTINGS	9518, 9562, 9581, 9601, 9667, 9668, 9834, 9972, 9999	0.67	0.86	0.75
SPORTS	9597, 9656, 9695, 9752, 9760, 9774, 9787, 9813, 9848, 9857, 9922	0.97	1.00	0.98

Discussion: (UNFINISHED)

- Compare different methods, their features and their performance. State any general observations.
- Discuss the metrics in the above two tables. Which metric(s) is/are more appropriate and why?
- If you continue this project, how would you improve it, e.g. using of other methods and parameters that have a potential to be useful but not tried yet?

Conclusion: (UNFINISHED)

Give a brief summary of the project and the findings, what have you discovered and learned from this project (if anything).

Reference:

Some reference resources and introduction about methods used can be found in:
https://github.com/miguelfzafra/Latest-News-Classifier/blob/master/0.%20Latest%20News%20Classifier/04.%20Model%20Training/09.%20MT%20-%20MultinomialNB.ipynb
"Introduction to text classification about Multinomial Naïve Bayes Model"

https://github.com/miguelfzafra/Latest-News-Classifier/blob/master/0.%20Latest%20News%20Classifier/04.%20Model%20Training/10.%20MT%20-%20Multinomial%20LogReg.ipynb
"Introduction to text classification about Multinomial Logistic Regression Model"