LING 131A Final Project

Pengcheng Xu, Yumeng Zhu, Zhiqi Chen

Introduction to Dataset and Methods:

This dataset contains around 200k news information from the year 2012 to 2018 pulled from HuffPost with 6 variables. The six variables are:
category: Category article belongs to
headline: Headline of the article
authors: Person authored the article
link: Link to the post
short_description: Short description of the article
date: Date the article was published

We plan to include five sections in the project:

1) In the beginning we will include some statistical analysis of the dataset, including word frequency, bigram frequency and trigram frequency.
2) In addition to the first section, we will include some visualizations to further demonstrate the analytical results.
3) In this section, we will use Naive Bayes, Logistic Regression and SGD methods to train six models in total to classify news categories based on raw counts or TFIDF counts.
4) In this section, we use a pairwise_distance function to calculate the distance between all articles with the target article, and recommend similar articles based on shortest distance.
5) Finally, we extract features of writing style of authors, who have at least 1,000 pieces of news contained in the dataset and infer authorship based on the features.

Section 1: Data Cleaning and Statistical Analyses

We examined all the categories and found some categories are very similar, so we decided to merge them, such as “THE WORLDPOST” and “WORLDPOST”. After combining similar categories, the total number of categories was reduced from 41 to 32. And we print out all categories in the following chart, with number of news pieces in each respective category. As we can see, categories such as politics and parents contain a lot of pieces of news (more than 10k) while some categories contain much fewer pieces of news (fewer than 3k) such as religion and fifty.

```
total categories: 32
category
ARTS                3878
BLACK VOICES        4528
BUSINESS            7644
COMEDY              5175
CRIME               3405
DIVORCE             3426
EDUCATION           2148
ENTERTAINMENT      16058
ENVIRONMENT         3945
FIFTY               1401
FOOD & DRINK        6226
GOOD NEWS           1398
HEALTHY LIVING      6694
HOME & LIVING       4195
IMPACT              3459
LATINO VOICES       1129
MEDIA               2815
PARENTS            12632
POLITICS           32739
QUEER VOICES        6314
RELIGION            2556
SPORTS              4884
STYLE              11903
TASTE               2096
TECH & SCIENCE      4260
TRAVEL              9887
WEDDINGS            3651
WEIRD NEWS          2670
WELLNESS           17827
WOMEN               3490
WORLD NEWS          2177
WORLDPOST           6243
dtype: int64
```

Meanwhile, we created a new variable ‘text’ combining headline and short description, so that we have more words to work with, and we believe by feeding this new variable into the machine learning model instead of either headline or short description variable, the model can generate more accurate results.

We also generated some statistical analyses on our variables. There are 200,853 short descriptions, and the average length of descriptions is 23. As to the sentences that form the short descriptions, there are 280,300 sentences in total, and the average number of sentences of a description is 1.40, which means one short description usually has one to two sentences.

We also got most frequent single words, bigrams and trigrams. The most frequent single words include one, time, people, etc. And the frequent bigrams are more interesting, such as New York,

Donald Trump, White House, Hillary Clinton, health care and climate change. We can see New York is the center of news, politics news is very popular, and people are concerned about their health and the environment. Besides, the Affordable Care Act is in the frequent trigrams, which is designed to reduce the cost of health insurance coverage for people who qualify. It is signed into law by President Barack Obama in March 2010, but it was still being cited frequently from 2012 to 2018.

25 most frequent single words:
[('one', 11060), ('time', 8840), ('people', 8603), ('like', 8160), ('new', 6886), ('us', 6869), ('said', 6414), ('would', 6055), ('get', 5869), ('life', 5846), ('make', 5399), ('know', 5320), ('Trump', 5159), ('many', 5081), ('years', 4909), ('first', 4876), ('way', 4839), ('world', 4814), ('could', 4771), ('may', 4574), ('year', 4466), ('day', 4366), ('want', 4252), ('even', 4160), ('need', 4025)]

25 most frequent bigrams:
[(('New', 'York'), 1689), (('Donald', 'Trump'), 1650), (('United', 'States'), 894), (('years', 'ago'), 739), (('HuffPost', 'Style'), 699), (('White', 'House'), 698), (('first', 'time'), 605), (('Hillary', 'Clinton'), 573), (('health', 'care'), 527), (('last', 'week'), 518), (('last', 'year'), 434), (('York', 'City'), 429), (('social', 'media'), 421), (('every', 'day'), 416), (('climate', 'change'), 411), (('Supreme', 'Court'), 409), (('feel', 'like'), 403), (('many', 'people'), 391), (('one', 'thing'), 372), (('President', 'Obama'), 368), (('Los', 'Angeles'), 349), (('new', 'study'), 326), (('make', 'sure'), 298), (('New', 'Year'), 289), (('high', 'school'), 285)]

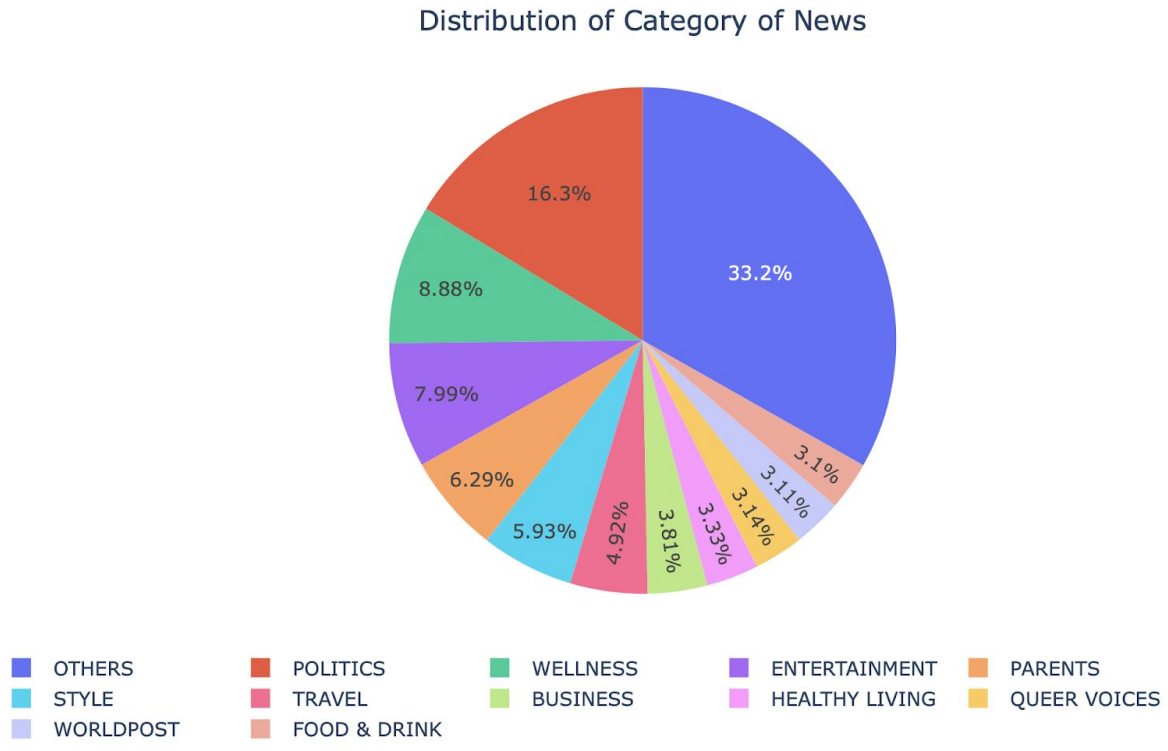
25 most frequent trigrams:
[(('New', 'York', 'City'), 429), (('New', 'York', 'Times'), 254), (('HuffPost', 'Rise', 'Morning'), 193), (('Rise', 'Morning', 'Newsbrief'), 193), (('President', 'Donald', 'Trump'), 164), (('President', 'Barack', 'Obama'), 132), (('Affordable', 'Care', 'Act'), 122), (('political', 'news', 'every'), 108), (('news', 'every', 'evening'), 107), (('home', 'story', 'idea'), 105), (('Saturday', 'Night', 'Live'), 96), (('New', 'York', 'Fashion'), 96), (('PR', 'pitches', 'sent'), 95), (('York', 'Fashion', 'Week'), 90), (('need', 'help', 'maintaining'), 79), (('personal', 'spiritual', 'practice'), 79), (('Kids', 'may', 'say'), 69), (('HuffPost', 'Style', 'beauty'), 59), (('Style', 'beauty', 'content'), 58), (('Fox', 'News', 'host'), 55), (('Mother', 'Nature', 'Network'), 53), (('popular', 'YouTube', 'videos'), 50), (('World', 'War', 'II'), 49), (('Twitter', 'never', 'fail'), 49), (('new', 'study', 'suggests'), 49)]

Section 2: Data Visualizations

After we cleaned the dataset, we conducted some statistical analysis on the dataset and visualized the results to have a general idea of the landscape of the dataset.

1. Distribution of Category of News

We plotted the distribution of news categories. We firstly merged 21 categories into one called OTHERS, because each of them accounts for less than 3%. The graph shows politics news occupies the most proportion of HuffPost, followed by wellness, entertainment, parents, style, etc. It could be that HuffPost is a really politicalized newspaper, or it is because the creator of the dataset pulled a large portion of politics news from HuffPost. In either case, the politics category seems an interesting category to probe into.

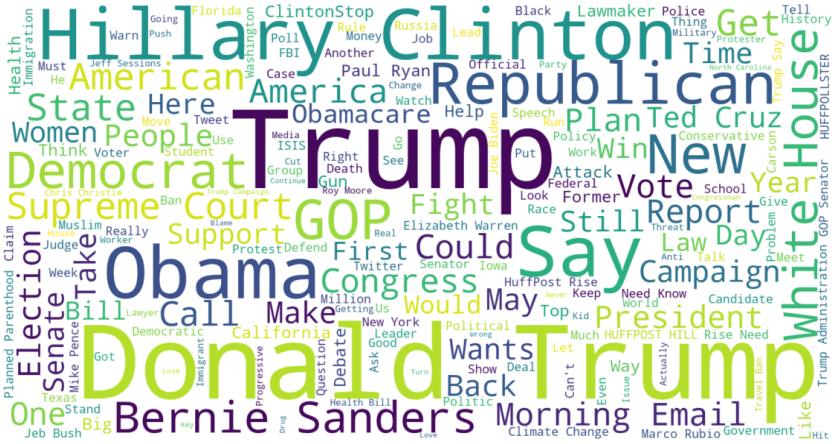


2. Word Cloud of Popular News Categories

For the popular news categories, we generated word clouds of their headlines, to find out which words or phrases are the most popular. Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance.

1) Politics news

As we can see, the most frequent words in politics news include Donald Trump, Hillary Clinton, Obama, White House, Republican, and Democrat, which are very reasonable. It can be concluded that HuffPost really features a lot of domestic news in the politics category, since almost no international politics related words appear in the word cloud. Moreover, as expected, Trump is a super star in the news. Both his last name alone and his full name are the most frequent words mentioned in politics news. In fact, news in the dataset spans from 2012 to 2018, and year 2016 was the election year. It is no surprise that Hillary Clinton also features in the world cloud, with some other names such as: Jeb Bush, Ted Cruz and etc.

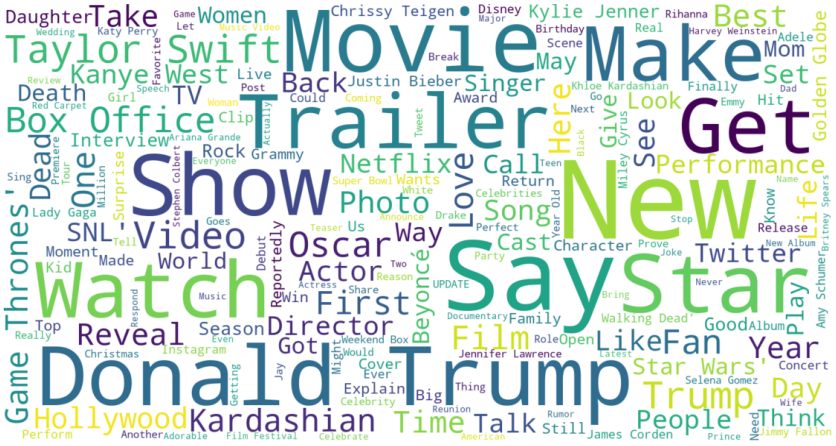


2) Wellness news

When it comes to wellness, people will think about life, health, food, sleep, love and time. We first did not quite understand what wellness news tell about, however, after we generated the word cloud, it became clear that wellness is synonym to health and lifestyle. It's funny that 'food' comes in much bigger letters than 'lost pounds' or 'diet', though we previously thought people might care more about losing weight than cuisines.

3) Entertainment news

The entertainment news is about movie, trailer, star, show, etc. The frequent names in the entertainment category include Taylor Swift, Kanye West, and Kardashian, and they are all super stars. Among all the TV series from 2012 to 2018, Game of Thrones is the most popular one. The interesting thing is “Donald Trump”, a politician and businessman, occurs frequently in the entertainment news. He is really good at catching people’s eyeballs!



4) Parenting news

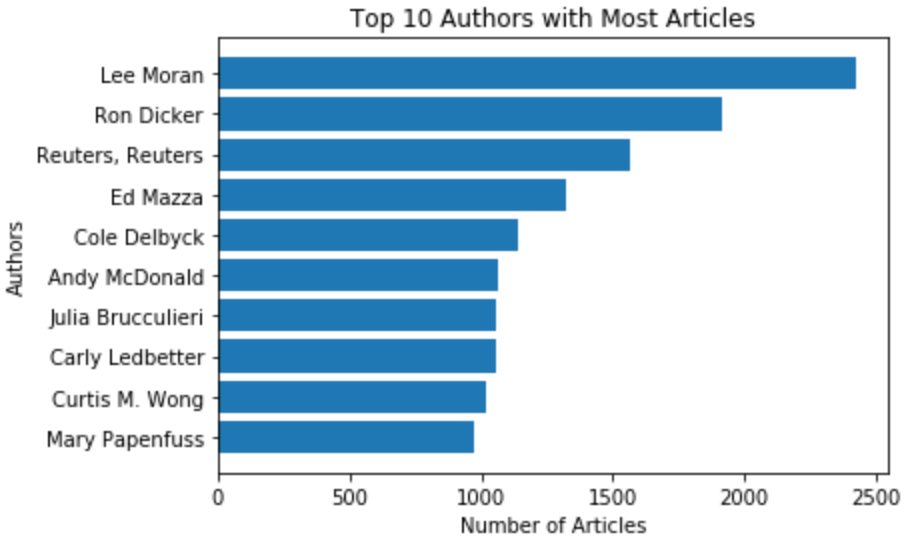
Parenting news is about everything from infancy through the teen years. The frequent words in this category are parent, mom, kid, children, daughter, baby, etc. It’s sad to see that autism appears in the word cloud as well.

5) Style news

Photo is the most frequent word in the style news. Fashion Week, Style Evolution and Red Carpet are popular too. Some names occur repeatedly in the style news, such as Michelle Obama, Kate Middleton and Rihanna.

3. Top 10 Authors with Most Articles

Because we would use news headlines and short descriptions to infer authorship in a later section, we generated a bar plot to show the top 10 authors with the most articles. Lee Moran, who is the most productive author, has written nearly 2,500 articles.



Section 3: Classification of News

In this section, we used raw count and TF-IDF method to convert individual tokens of the text variable into features, and used Multinomial Naive Bayes, Decision Tree, Logistic Regression and SGD Classifier methods to train models using the two features respectively. During the process, we considered using bigrams instead of individual tokens as well, and generated the Naive Bayes models accuracy rates for bigrams features using raw count and TF-IDF methods respectively. The names of these two models are: bi_bayes_bow and bi_bayes_tfidf. However, the accuracy rate of these two models are 0.42 and 0.41, which are considerably lower than Naive Bayes models using individual tokens. As a result, we stopped using bigram features at all for other models, assuming that feature engineering using individual tokens captures enough information of the text variable and suffixes for classification of news. The two Decision Tree models took more than 5 minutes to train, so we abandoned them. In the end, we had 8 models, including the two models using bigram features, and pickled all of them out so that they can be used later for any piece of news later. We also built argparse functions in the file so that the train function can be accessed directly from the terminal.

When the following code is fed into the terminal:

```
$ python3 NLP\ Final\ Project.py --train
```

The TRAIN function in the final project file is set to run and should be able to generate results as shown in the graph below:

```
Creating Bayes classifier in classifiers/uni_bayes_bow.pkl
Accuracy: 0.60
Time:  1.75
Creating Bayes classifier in classifiers/uni_bayes_tfidf.pkl
Accuracy: 0.56
Time:  1.64
Creating Bayes classifier in classifiers/bi_bayes_bow.pkl
Accuracy: 0.42
Time:  1.6
Creating Bayes classifier in classifiers/bi_bayes_tfidf.pkl
Accuracy: 0.41
Time:  1.6
/Users/xupech/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWa
rning: Default solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning
.
  FutureWarning)
/Users/xupech/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:460: FutureWa
rning: Default multi_class will be changed to 'auto' in 0.22. Specify the multi_class option to si
lence this warning.
  "this warning.", FutureWarning)
Creating logistic regression in classifiers/uni_logistic_bow.pkl
Accuracy: 0.62
Time:  107.69
Creating logistic regression in classifiers/uni_logistic_tfidf.pkl
Accuracy: 0.63
Time:  51.47
/Users/xupech/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/stochastic_gradient.py:16
6: FutureWarning: max_iter and tol parameters have been added in SGDClassifier in 0.19. If both ar
e left unset, they default to max_iter=5 and tol=None. If tol is not None, max_iter defaults to ma
x_iter=1000. From 0.21, default max_iter will be 1000, and default tol will be 1e-3.
  FutureWarning)
Creating sdg classifier in classifiers/uni_sgd_bow.pkl
Accuracy: 0.61
Time:  5.03
Creating sdg classifier in classifiers/uni_sgd_tfidf.pkl
Accuracy: 0.60
Time:  5.78
```

The total run time shall take around 2 minutes, since the program will first combine all 200k rows of headlines and short descriptions into one variable: text and generate features based on the variable, however, the runtime to train models do not take too long. In fact, most models take fewer than 10 seconds to train.

The models are:
uni_bayes_bow: Multinomial Naive Bayes using raw count unigram features
uni_bayes_tfidf : Multinomial Naive Bayes using TF-IDF unigram features
bi_bayes_bow: Multinomial Naive Bayes using raw count bigram features
bi_bayes_tfidf: Multinomial Naive Bayes using TF-IDF bigram features
uni_logistic_bow: Logistic Regression using raw count unigram features
uni_logistic_tfidf: Logistic Regression using TF-IDF unigram features

uni_sgd_bow: SGD Classifier using raw count unigram features
uni_sgd_tfidf: SGD Classifier using TF-IDF unigram features

Most models have an accuracy rate of more than 60% (test accuracy rate). Given that there are 32 categories in total, the accuracy rate is not bad. Using TF-IDF method does not make too much difference with using raw count method in feature engineering. Meanwhile, there is no distinctly best model among the three models: Multinomial Naive Bayes, Logistic Regression or SGD Classifier. All three models give more or less the same result (around 60%).

In addition, prediction is also embedded in the program, and when the following code is executed in the terminal, a prediction of the news category will be given:

```
$ python3 NLP\ Final\ Project.py --run bow data/1.txt
```

The first argument of the line is always 'run', the second argument specifies the way of feature engineering, with two options to choose: bow is raw counts, tfidf is TF-IDF method, the third argument is the file directory.

```
Choose a model:
1 - unigram_bayes
2 - unigram_logistic_regression
3 - sgd classifier
Type a number:
```

After that, the program will ask for the model to train for prediction. 1 is the unigram Multinomial Naive Bayes model, 2 is the unigram Logistic Regression model, and 3 is SGD model. After specifying the model, the program will generate a category prediction for the news piece.

For example, we would like to categorize the following news piece about the soccer star Christiano Ronaldo, the prediction is sports, which meets our expectations.

'Real Madrid star Ronaldo became the top scorer for the club in 2016, surpassing Raul. However, he made a transfer to Juventus in 2017 summer after he failed to reach an agreement with his former club on his contract renewal terms, in which he demanded a crazy amount of salary. Any football club on earth will not agree on such a great amount, despite the success Ronaldo brought to the club.'

Section 4: Recommendation of News

Another thing we work on the dataset is building 4 recommendation systems for the piece of news read. We again used two featuring methods, one is raw count, the other is TF-IDF. Moreover, for each featuring method, we had two ways to recommend news: one takes category and author of the news into consideration, the other gives weight to text only. The basic method for all models is to calculate the distance between each other news piece and the targeted news piece, and recommend 10 news pieces with the shortest distance to the targeted news piece. The weight we gave to author, category and text are 30%, 30% and 40% respectively.

The recommendation system works only for news in the dataset. It takes the index of that piece of news as input and generates as many recommended news pieces as the user desires. This recommendation feature can also be accessed through terminal as following:

```
$ python3 NLP\ Final\ Project.py --recommend 10 10
```

The line takes three arguments. The first argument ‘recommend’ is always there, the second argument is the target news piece, it takes the index of the news as input, the third argument is the desired number of recommendations.

After this line is executed, the program will ask the reader to pick a recommendation model: 1 is the TFIDF model using only the text variable, 2 is the raw count model using only text variable, 3 is the TFIDF model using text, author and category variables, and 4 is the raw count model using three variables.

```
Choose how you want to recommend the news, dear reader:
1 - text based only in tfidf
2 - text based only using raw count
3 - text based in tfidf plus giving weight to author and category
4 - text based using raw count plus giving weight to author and category
Type a number:
```

After a model is picked, the program will print out the target news and the desired number of recommendation news pieces.

```
 News Just Read:
Sebastian Murdock | 2018-05-26 00:00:00 | ENTERTAINMENT | Justin Timberlake Visits Texas School Shooting Victims The
 pop star also wore a "Santa Fe Strong" shirt at his show in Houston.

 Recommended News For You:
Lily Karlin | 2015-01-27 00:00:00 | ENTERTAINMENT | This Could Be The Next Shonda Rhimes Show


Rev. Peter E. Bauer, ContributorUnited Church of Christ minister | 2015-04-29 00:00:00 | ENTERTAINMENT | WHO Are You
 Now ?


Irina Dvalidze | 2014-10-12 00:00:00 | ENTERTAINMENT | The Last Hurrah?


Jessica Toomer | 2014-06-02 00:00:00 | ENTERTAINMENT | R.I.P.


 | 2014-06-14 00:00:00 | ENTERTAINMENT | OUT IN FRONT


Rob Taub, ContributorWriter, Humorist & Television Commentator | 2014-04-18 00:00:00 | ENTERTAINMENT | The Name's Ma
met, Clara Mamet


Tom Klein, ContributorChair of Animation program, LMU School of Film and Television | 2016-06-18 00:00:00 | ENTERTAI
NMENT | Warcraft and PIXARcraft


Fiona Finn, ContributorCancer Survivor, Author & Keynote Speaker | 2014-07-06 00:00:00 | HEALTHY LIVING | No Butts A
bout It: Get a Colonoscopy!


Lisa K. Brown, ContributorFreelance writer and bemused 50-year-old | 2014-07-02 00:00:00 | FIFTY | Over


Yasmine Hafiz | 2014-06-30 00:00:00 | RELIGION | What Is A Caliphate?
```

The above is an exemplary result print out of 10 recommended news for news piece no.10 using TFIDF method taking consideration of all three variables. As we can see from the result, the 10 pieces of recommendations align well with the target news piece in terms of category, however, several pieces of recommendations are too short to make sense at all. Maybe before we dive into building the recommendation system, we should further clean the dataset by removing news that have fewer than 5 words combining headline and short description. After some consideration, we decided to retain these news pieces, since the original news is unavailable to us. We only have access to the dataset uploaded on Kaggle, and have no idea how it might differ from the original news. In this case, we thought it would be better to trust the creator of the dataset, as if s/he gathers all information from the original news. It will make no sense then to remove some pieces of news from the dataset. Rather, it is better to leave the whole picture of the dataset unchanged.

Section 5: Identify Authorship

This section aims to identify authorship based on the title, short description, and category of the news as we believe different writers have different writing styles. Since there are more than 20,000 authors which is too complicated for a classification problem, we only chose the authors who wrote more than 1000 articles. In order to achieve that, we grouped the data by authors, counted the number of news articles of each author, and sorted the data in descending order. As a result, 9 authors are selected and 9 different labels are created. Similar to the previous section, the title, the short description, and the category are joined together to form a string for each news article. Then we used raw counts or binary features to create the feature sets. After creating the feature set and the label set, we split them into the train and test dataset, and feed them into 4 different classification models - MultinomialNB, BernoulliNB, LogisticRegression, and DecisionTree. It was worth noting that, for the Naive Bayes models, raw count features are fed into MultinomialNB and binary features are fed into BernoulliNB. The following table is the accuracies and the elapsed time of each model.

raw_count_binary	vectorizer	classifier	accuracy	elapsed_time
raw_count	Tfidf	LR	0.64	2.41
binary	Tfidf	LR	0.64	2.55
raw_count	Count	LR	0.62	6.19
binary	Count	LR	0.62	5.90
raw_count	Count	DT	0.57	83.94
raw_count	Count	MNB	0.57	4.07
binary	Count	DT	0.55	87.13
binary	Tfidf	DT	0.54	63.26
raw_count	Tfidf	DT	0.53	55.91
raw_count	Tfidf	MNB	0.48	1.95
binary	Count	BNB	0.46	7.94
binary	Tfidf	BNB	0.46	4.49

As shown in the table, the average accuracy is around 55%, which meets our expectations since only titles, short descriptions, and categories, rather than the full articles are provided in the dataset and there are 9 authors. The logistic regression models generally have the highest accuracies which are above 0.60. The Decision Tree models come to the next and the two Naive Bayes models are the least accurate. Besides, whether using raw_count approach or binary approach to convert words to features, or whether using TfidfVectorizer and CountVectorizer do not make significant differences to accuracies. In terms of the runtime, the Decision Tree models are much slower than the others, which may be caused by its complexity. As a whole, the tfidf logistic regression models perform the best, because their average accuracy is the highest among 12 models and the runtime is shorter than the others.