Text Mining Assignment

Article Popularity Prediction, Classification & Virality of News Articles

OCTOBER 2017

Contents

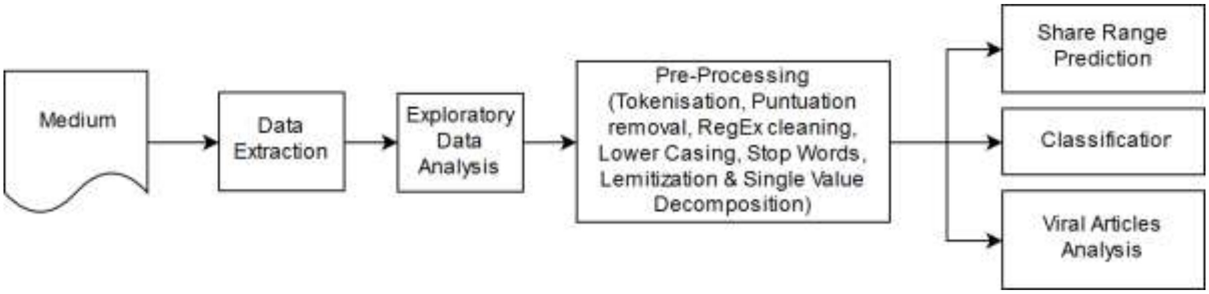
1. Introduction

Medium is an online publishing platform, having a hybrid collection of blog articles from amateur and professional blogger and publications. Medium has a Clap button for every article on its blogging platform. In this exercise we will refer to Claps as the share count, representing the popularity of the article.

The objective of this project is to extract and analyze text data from Medium. In this project, we aim to answer the following questions using textual analysis:

1. Given a set of attributes of an article, what is the range of shares it will get?
2. Based on the text in the articles, how would you classify articles in to different categories?
3. What factors contribute towards making an article viral?

These questions will be answered by extracting training samples of unstructured documents and the results will be projected to new text. Data preparation would include transforming text to numerical format upon which various text mining tasks would be performed by adopting a predictive framework for machine learning.


Medium
Data Extraction
Exploratory Data Analysis
Pre-Processing (Tokenisation, Puntuation removal, RegEx cleaning, Lower Casing, Stop Words, Lemitization & Single Value Decomposition)
Share Range Prediction
Classificatior
Viral Articles Analysis

2. Data Acquisition

2.1 The target data

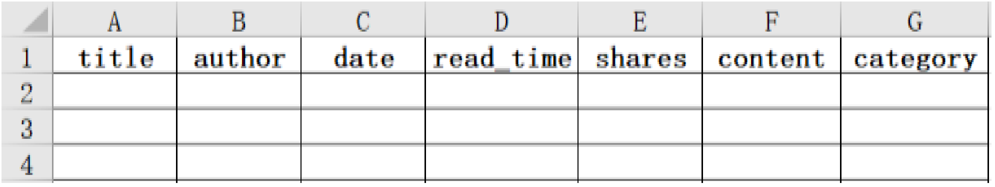
Data was acquired from the Medium website where information in stored in various categories. In Medium's homepage there are categories such as: Home, Collections, Culture, Tech, Self, Politics, Design, Health, Popular and more. Seven categories for chosen for this analysis: Culture, Tech, Self, Politics, Science, Design, Entrepreneur and Popular. A web crawling process was used to extract articles from these chosen categories to obtain content and the number of shares.

2.2 The preparatory work

Two Python scripts were used for this task: one for getting all the links of articles and the other for getting content from all the links. An exe program of Chrome Driver was used to load HTML and to interact with it.

2.3 Data Extraction

The objective of the data extraction step is to obtain data in the following format:



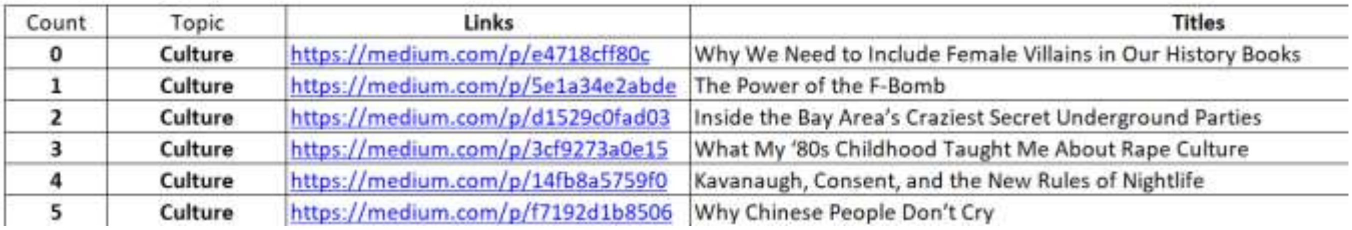
	A	B	C	D	E	F	G
1	title	author	date	read_time	shares	content	category
2							
3							
4							

Step 1: Extracting URLs of all articles

The first step is to get the URL link of articles for the chosen categories. Medium has a page for each category, however the pages do not load all the articles at once. So the task of scrolling down the pages was automated for every category.

- https://medium.com/topic/culture
- https://medium.com/topic/culture
- https://medium.com/topic/politics
- https://medium.com/topic/design
- https://medium.com/topic/popular
- https://medium.com/topic/self
- https://medium.com/topic/science
- https://medium.com/topic/entrepreneur

An initial Python Script was used to extract the topics or categories, titles of articles and the relevant links. The excel file at this stage had 11406 entries.



Count	Topic	Links	Titles
0	Culture	https://medium.com/p/e4718cff80c	Why We Need to Include Female Villains in Our History Books
1	Culture	https://medium.com/p/5e1a34e2abde	The Power of the F-Bomb
2	Culture	https://medium.com/p/d1529c0fad03	Inside the Bay Area's Craziest Secret Underground Parties
3	Culture	https://medium.com/p/3cf9273a0e15	What My '80s Childhood Taught Me About Rape Culture
4	Culture	https://medium.com/p/14fb8a5759f0	Kavanaugh, Consent, and the New Rules of Nightlife
5	Culture	https://medium.com/p/f7192d1b8506	Why Chinese People Don't Cry

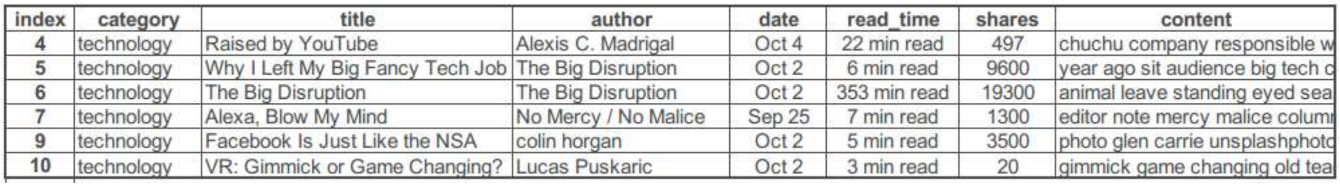
Step 2: Authentication to use Medium website

During the process of extraction of content from each URL, it was discovered that a majority of the articles were not available without a Medium premium account and it was necessary to be connected to view the content and scrap the pages.

In order to solve these issues, we purchased a premium account. The **cookie** and **User-Agent** data was obtained after logging in with the premium account and this data was used as "headers" in our crawler script. Meanwhile the "Connection" parameter was set as "keep alive" in headers to stay connected.

Step 3: Crawl the details of the articles

A python code was written to authenticate, retrieve, and keep the cookies and to extract the content of all the articles. At this stage, an Excel table as shown in the figure below was obtained.



index	category	title	author	date	read_time	shares	content
4	technology	Raised by YouTube	Alexis C. Madrigal	Oct 4	22 min read	497	chuchu company responsible w
5	technology	Why I Left My Big Fancy Tech Job	The Big Disruption	Oct 2	6 min read	9600	year ago sit audience big tech c
6	technology	The Big Disruption	The Big Disruption	Oct 2	353 min read	19300	animal leave standing eyed sea
7	technology	Alexa, Blow My Mind	No Mercy / No Malice	Sep 25	7 min read	1300	editor note mercy malice colum
9	technology	Facebook Is Just Like the NSA	colin horgan	Oct 2	5 min read	3500	photo glen carrie unsplashphoto
10	technology	VR: Gimmick or Game Changing?	Lucas Puskaric	Oct 2	3 min read	20	gimmick game changing old tea

The data dictionary is in the table below:

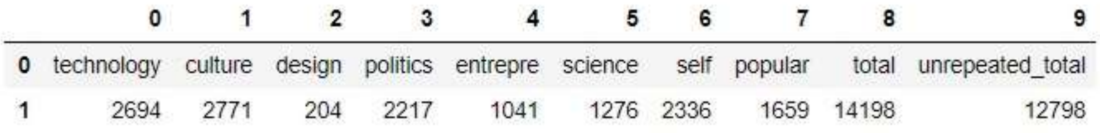
Column name	Description
Index	Serial number of the article
Category	Article categories such as Technology, design, culture, politics, entrepreneur, science, self and popular
title	Title of the article
author	Author of the article
date	In month-day-year format
read_time	Reading time of an article, in minutes
share	Share means share number of one article. The original share number is in xxk format. When it is beyond 1000 the “xxk” data was multiplied by 1000 to simplify this field.
content	The text in an article

2.4 Increasing shares of recent articles

Recently published articles will usually see an increase in the number of shares for a few days. To allow the number of shares to stabilize for recent articles, The links were extracted on 2018/10/6, and the content was extracted on 2018/10/12, giving a six-day period to ensure stability.

3 Exploratory Data Analysis

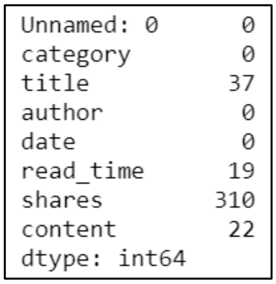
As seen in the table, Technology, Self, Culture, Politics got the maximum count of articles. Popular category has some same articles with other categories is not considered in the further analysis.



	0	1	2	3	4	5	6	7	8	9
0	technology	culture	design	politics	entrepre	science	self	popular	total	unrepeated_total
1	2694	2771	204	2217	1041	1276	2336	1659	14198	12798

3.1 Identifying and treating missing values

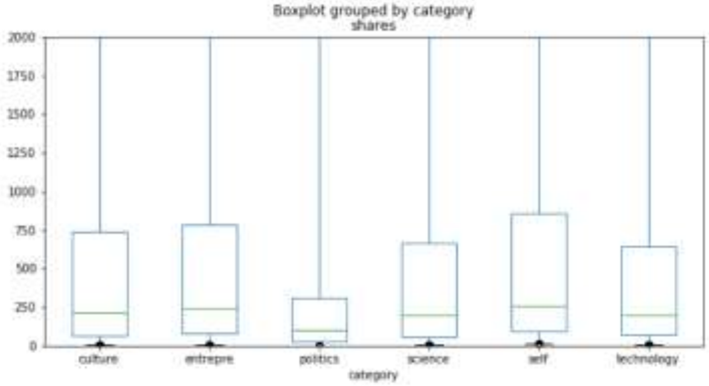
An initial analysis of data for missing values showed the following results. The missing values in the shares columns are the articles which were not shared. The missing data under content are articles which have no text, but only pictures. These records would be removed as they cannot be used for text analysis. The ones without reading time can be imputed using a suitable method.



```
Unnamed: 0      0
category        0
title          37
author          0
date            0
read_time      19
shares        310
content        22
dtype: int64
```

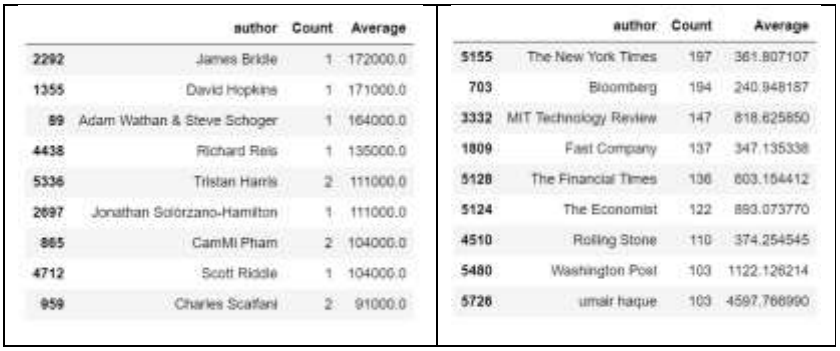
3.2 Content by category

The box plot below shows the count of articles grouped by number of shares. While the median shares for most categories was around 250, articles under popular, technology, self and entrepreneur had quite a few articles with very high shares. The box plot shows that the median of articles in the “self” category is slightly above than that for other categories.



3.3 Top ten authors by average shares and count

The figure on the left shows the top 10 authors based on average number of shares and the one on right sorts the authors on the number of articles published. A clear distinction is seen here: the authors in the top 10 list, based on number of shares, seem to be individual bloggers while the authors who have published more number of articles are bigger news agencies.

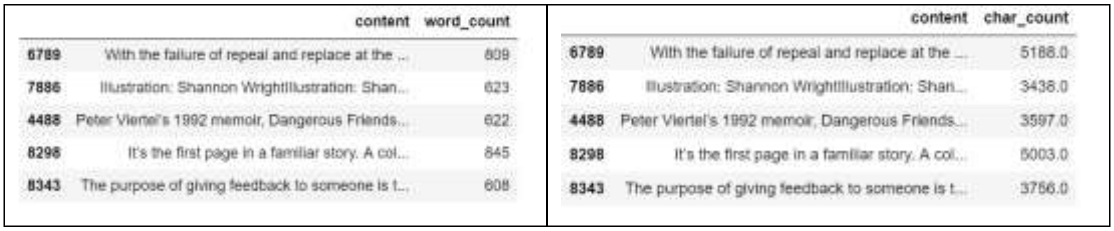


	author	Count	Average
2292	James Bridle	1	172000.0
1355	David Hopkins	1	171000.0
89	Adam Wathan & Steve Schoger	1	164000.0
4438	Richard Reis	1	135000.0
5336	Tristan Harris	2	111000.0
2697	Jonathan Solórzano-Hamilton	1	111000.0
865	CamMi Pham	2	104000.0
4712	Scott Riddle	1	104000.0
959	Charles Scalfani	2	91000.0

	author	Count	Average
5155	The New York Times	197	361.807107
703	Bloomberg	194	240.948187
3332	MIT Technology Review	147	818.625850
1809	Fast Company	137	347.135338
5128	The Financial Times	136	603.154412
5124	The Economist	122	893.073770
4510	Rolling Stone	110	374.254545
5480	Washington Post	103	1122.126214
5726	umair haque	103	4597.766990

3.4 Word count and character count

New features such as word count and character count were extracted based on the number of words and number of characters in each article. These could be used as features for model building.

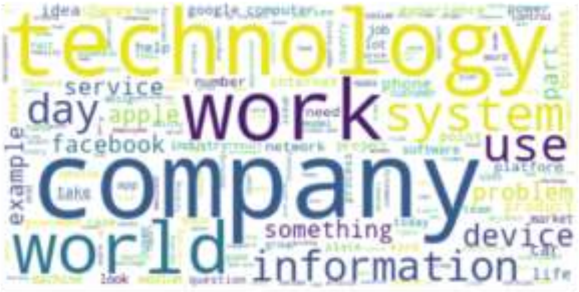


	content	word_count
6789	With the failure of repeal and replace at the ...	809
7886	Illustration: Shannon WrightIllustration: Shan...	623
4488	Peter Viertel's 1992 memoir, Dangerous Friends...	622
8298	It's the first page in a familiar story. A col...	845
8343	The purpose of giving feedback to someone is t...	608

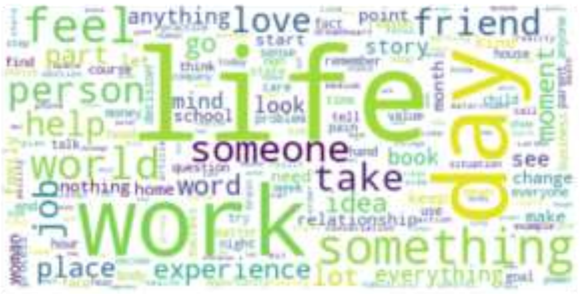
	content	char_count
6789	With the failure of repeal and replace at the ...	5188.0
7886	Illustration: Shannon WrightIllustration: Shan...	3438.0
4488	Peter Viertel's 1992 memoir, Dangerous Friends...	3597.0
8298	It's the first page in a familiar story. A col...	5003.0
8343	The purpose of giving feedback to someone is t...	3756.0

3.5 Word Clouds

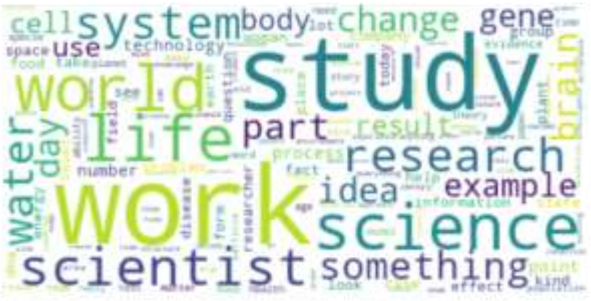
To understand the frequency of words occurring in each category, word clouds were created for all the six individual categories. For obtaining accurate word clouds, nouns and verbs in each category were extracted using Parts-of-speech tags. The categories can be distinguished and identified based on the frequency of words in the word cloud.



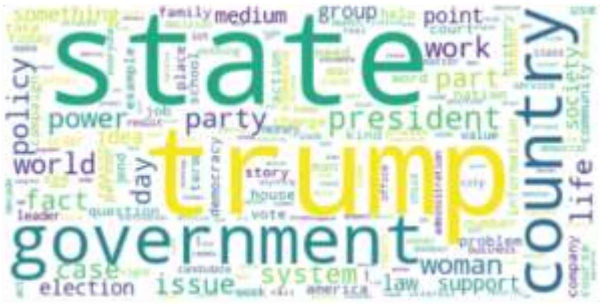
Technology



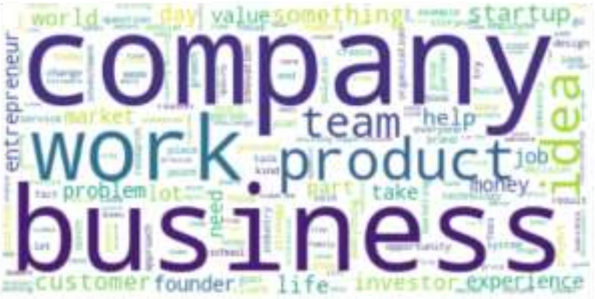
Self



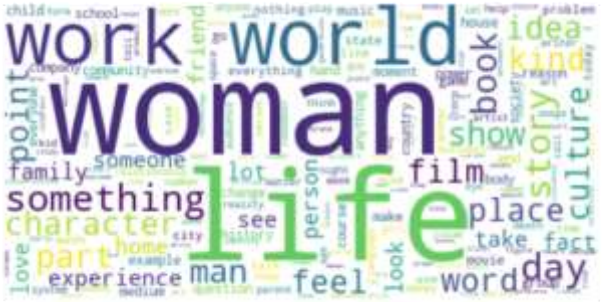
Science



Politics



Entrepreneur



Culture

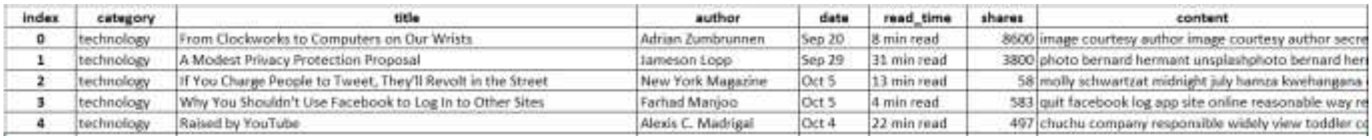
4 Article Popularity Prediction

4.1 Processing data

Following preprocessing steps were performed on the content column of the data frame. Tokenization was done to break down the text content stream into words, terms, symbols or some other meaningful element called a tag. This was followed by lemmatization to combine the variants of words. After the word segmentation, the words will include some stop words, as well as some useless numbers and punctuation. So, in the final step, we remove the stop words, punctuation and numbers.

Common nouns in the content of each article were extracted along with the number of shares. Number of shares were considered as the weights to calculate the weighted average of each common noun. The nouns were then sorted based on the weighted averages.

After pre-processing data, the following columns were obtained:



index	category	title	author	date	read_time	shares	content
0	technology	From Clockworks to Computers on Our Wrists	Adrian Zumbrunnen	Sep 20	8 min read	8600	image courtesy author image courtesy author secre
1	technology	A Modest Privacy Protection Proposal	Jameson Lopp	Sep 29	31 min read	3800	photo bernard hermant unsplashphoto bernard her
2	technology	If You Charge People to Tweet, They'll Revolt in the Street	New York Magazine	Oct 5	13 min read	58	molly schwartzat midnight july hamza kwehangana
3	technology	Why You Shouldn't Use Facebook to Log In to Other Sites	Farhad Manjoo	Oct 5	4 min read	583	quit facebook log app site online reasonable way re
4	technology	Raised by YouTube	Alexis C. Madrigal	Oct 4	22 min read	497	chuchu company responsible widely view toddler co

4.2 Modelling

The data was split into train and test using a 70-30 ratio.
Initially a regression model was fit to the training data to predict the number of shares. The results of the obtained fit were found to be very poor and it was found that there was no correlation between the nouns in the article and the number of shares.
It was therefore decided to predict the popularity of an article by binning the number of shares into 3 different ranges. Articles with 0-100 shares were categorized as “Common”, articles with 100-1000 shares were categorized as “Good” articles with shares >1000 were categorized as “Amazing”.
The categories are as follows:

Segment	Share Count range	No of Articles
common	0-100	4000
good	100-1000	5000
amazing	>1000	2500

4.2.1 Initial Run

Tf-idf Vectorizer was used on the content and accuracy was obtained for the following models: Random Forest Classifier, Decision Tree Classifier, SGDClassifier, Ridge Classifier CV and Logistic Regression models.
As can be seen from the above results, SGDClassifier and Logistic Regression got the best results, about 50%.

Model	Accuracy
Random Forest Classifier	40%
Decision Tree Classifier	41%
Ridge Classifier CV	50 %
SGDClassifier	50.5 %
Logistic Regression	50.9 %

4.2.2 Model optimization

1. Dimensionality reduction

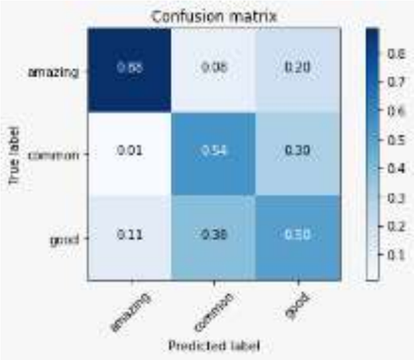
Around 80,000 tokens were obtained from nouns and verbs, therefore the SVD method was used to reduce the dimension of the TfidfVectorizer results, and the explained variance of SVD is 83%.
The results obtained by using dimensionality reduction were better than previous results. Dimension reduction data was trained on LR SGD and the accuracy improved by 1%.

2. Add Category and Read_time as input

For optimization of the model, on the basis of the data after dimensionality reduction, we added Category and Read_time as X to input, and compared the results obtained with those only use the data after dimensionality reduction as X to input. The summary table compares the results obtained by taking only the dimensionally reduced data as input, and the results obtained by adding the classification and reading time as inputs.

The accuracy of the results using different inputs were compared, and an average increase of 0.5% was observed.

This was due to the addition of comparative dimensions, especially the factors that cause people to share an article must have a lot to do with reading time and classification of articles, so the accuracy of classification was further improved.



3. Add Ridge method when processing data

Because the data is very sparse, the ridge Classifier method was used, which produced a high precision for classifications such as amazing. But the overall accuracy was still the highest for Logistic Regression. The summary table below summarizes the comparison the accuracy of the results between using the Ridge Classifier CV method and without the Ridge Classifier CV method.

4.3 Summary Table

Below table summarizes the model performance and the accuracy measures achieved.

Data	Model	Accuracy	Class	Precision	Recall	F1 Score
Content (Noun & Verb)	Logistic Regression	50.99%	Amazing	56%	33%	41%
			Good	49%	71%	58%
			Common	54%	38%	45%
	SGD Classifier	50.59%	Amazing	55%	32%	41%
			Good	49%	71%	58%
			Common	53%	38%	44%
	Ridge Classifier CV	50.19%	Amazing	59%	24%	34%
			Good	47%	80%	59%
			Common	57%	31%	40%
Content (Noun & Verb) -SVD	Logistic Regression	51.31%	Amazing	59%	35%	44%
			Good	50%	69%	58%
			Common	51%	39%	44%
	SGD Classifier	49.41%	Amazing	53%	40%	46%
			Good	52%	52%	52%
			Common	45%	53%	49%
	Ridge Classifier CV	51.28%	Amazing	64%	28%	39%
			Good	49%	77%	60%
			Common	52%	34%	41%
With addition of Read_time & Category Excluding Popular) & Content (Noun & Verb) -SVD	Logistic Regression	54.98%	Amazing	79%	42%	55%
			Good	51%	73%	60%
			Common	52%	40%	45%
	SGD Classifier	53.21%	Amazing	85%	37%	52%
			Good	49%	87%	63%
			Common	52%	20%	29%
	Ridge Classifier CV	54.68%	Amazing	88%	36%	51%
			Good	50%	80%	62%
			Common	54%	35%	42%

5. Classification

5.1 Pre-processing and Feature Extraction

The first step to handling text is to break the stream of characters into words, popularly known as tokens. After obtaining tokens, the next step was to remove common punctuations and turn all tokens to lower case text. Following this, stop words were removed using a standard dictionary of stop words in English since these words have very low predictive power. Lemmatization was preferred over stemming as stemming resulted in loss of information, while reducing the number of unique words.

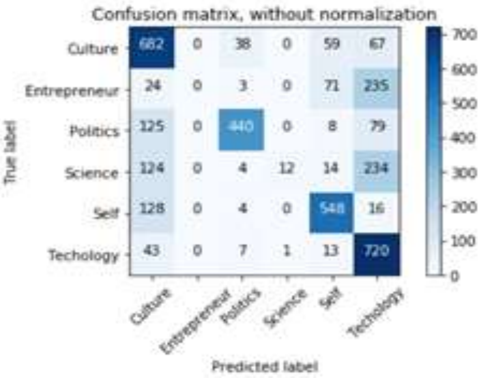
The first step in feature extraction is to transform documents to vector form. Since longer documents will have higher average count values than shorter documents, dividing the number of occurrences of each word in a document by the total number of words in a document, will result in a term frequency feature which is more appropriate than just counting occurrences. Further, to modify the frequency of a word in a document by the perceived importance of the word, a feature representing the inverse document frequency was extracted. This was done to not give weightage to common words that are used too frequently.

5.2 Modelling

The data was initially split in to training data and test data. The preprocessing steps were applied to the test data were applied to the test data and the following results were obtained from each of the classification methods.

5.2.1 Naïve Bayes

Naïve Bayes algorithm works by calculating the conditional probability of a word occurring in a document given that the document belongs to a category. Naïve bayes resulted into no prediction for the Entrepreneur category. Multinomial Naïve Bayes was run on the data to obtain classification results as:

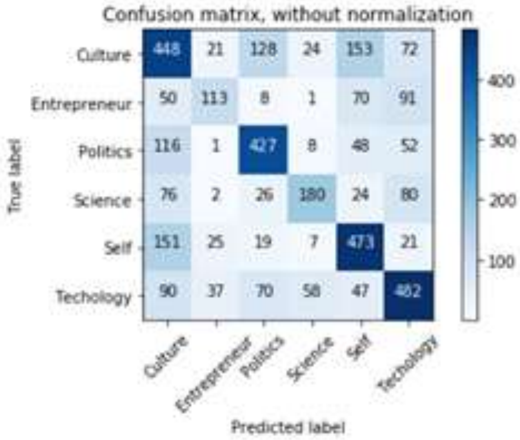


Confusion matrix, without normalization

True label \ Predicted label	Culture	Entrepreneur	Politics	Science	Self	Techology
Culture	682	0	38	0	59	67
Entrepreneur	24	0	3	0	71	235
Politics	125	0	440	0	8	79
Science	124	0	4	12	14	234
Self	128	0	4	0	548	16
Techology	43	0	7	1	13	720

5.2.2 Random Forest Classifier

A random forest classifier was used to classify an example by starting at the root of the tree and moving through it until a leaf node, which provides the classification of the instance. For text classification, separate trees are built for each category where word counts are used as features.

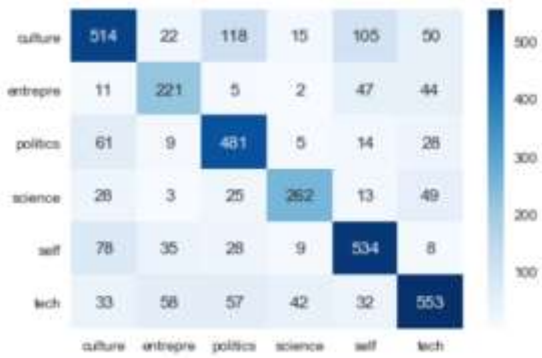


Confusion matrix, without normalization

True label \ Predicted label	Culture	Entrepreneur	Politics	Science	Self	Techology
Culture	448	21	128	24	153	72
Entrepreneur	50	113	8	1	70	91
Politics	116	1	427	8	48	52
Science	76	2	26	180	24	80
Self	151	25	19	7	473	21
Techology	90	37	70	58	47	462

5.2.3 Neural Network

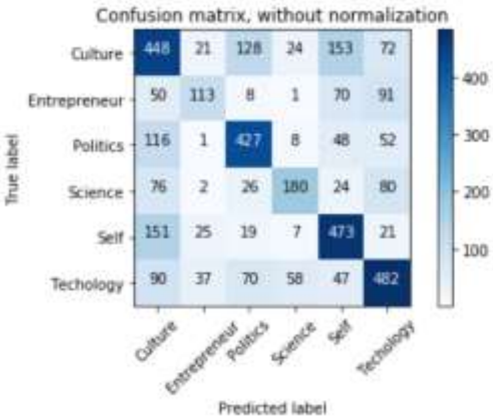
We built a multi-layer perceptron with dense hidden layers and relu activation functions. We set the interlayer dropout rate to 50% to avoid overfitting. The final output layer consists of 6 softmax nodes denoting the classes. 10% of the regular train set was split to create a validation set to drive ADAM's greedy search and compute the loss function. The loss function used is categorical cross-entropy.



	culture	entrepre	politics	science	self	tech
culture	514	22	118	15	105	50
entrepre	11	221	5	2	47	44
politics	61	9	481	5	14	28
science	28	3	25	262	13	49
self	78	35	28	9	534	8
tech	33	58	57	42	32	553

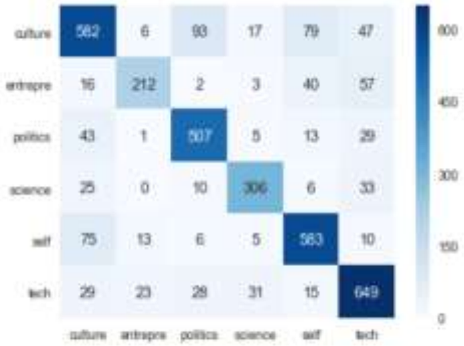
5.2.4 Support Vector Machines

Using SVM would provide an option to pick between many possible classifiers in a way that guarantees a higher chance of correctly labeling the test data. As seen in the confusion matric on the right below, this classifier classifies documents more accurately than the other methods used previously and gives accuracy of 79%. The confusion matrix on the left was obtained without using SVD.



Confusion matrix, without normalization

True label \ Predicted label	Culture	Entrepreneur	Politics	Science	Self	Techology
Culture	448	21	128	24	153	72
Entrepreneur	50	113	8	1	70	91
Politics	116	1	427	8	48	52
Science	76	2	26	180	24	80
Self	151	25	19	7	473	21
Techology	90	37	70	58	47	482



	culture	entrepre	politics	science	self	tech
culture	582	6	93	17	79	47
entrepre	16	212	2	3	40	57
politics	43	1	507	5	13	29
science	25	0	10	306	6	33
self	75	13	6	5	563	10
tech	29	23	28	31	15	649

5.3 Summary Table

The following table summarizes the accuracy results obtained from the 4 classifiers. As seen in the results below, SVM classifier with SVD gives the best output.

Classifier	Accuracy	Category	Precision	Recall	F1 score
Naïve Bayes	**64.90%**	Culture	61%	81%	69%
		Entrepreneur	0%	0%	0%
		Politics	89%	67%	77%
		Science	92%	3%	6%
		Self	77%	79%	78%
		Technology	53%	92%	67%
Random Forest	**57.30%**	Culture	48%	53%	50%
		Entrepreneur	57%	34%	42%
		Politics	63%	65%	64%
		Science	65%	46%	54%
		Self	58%	68%	63%
		Technology	60%	61%	61%

Support Vector Machine	**76.30%**	Culture	73%	68%	70%
		Entrepreneur	79%	62%	69%
		Politics	77%	83%	80%
		Science	78%	77%	78%
		Self	77%	84%	80%
		Technology	77%	79%	78%
Support Vector Machine with SVD	**79.00%**	Culture	76%	71%	73%
		Entrepreneur	83%	64%	72%
		Politics	78%	85%	82%
		Science	83%	81%	82%
		Self	79%	84%	82%
		Technology	79%	84%	81%
Neural Network with SVD	**71%**	Culture	71%	62%	66%
		Entrepreneur	64%	67%	65%
		Politics	67%	80%	73%
		Science	78%	69%	73%
		Self	72%	77%	74%
		Technology	76%	71%	73%

6. What makes news Articles Viral

Following are the key factor for an article to be viral:

- The important prerequisite for making an article viral is to have captivating content.
- The share-ability of an article depends on multiple factors ranging from the length of the article, read time required to the quality and uniqueness that the article comprises.
- For an article to go viral, it should firstly make a connection with the readers. It also depends on the sentiment being conveyed in it and how well it relates to the readers. Thus, an article should be relatable and convincing enough for the audience to further share it.
- The article share also depends on whether it sends out a positive, negative or neutral message which is identified as the polarity of an article.
- Along with the sentiment, articles that have a surprise element or interesting facts or intense discussions are highly likely to go viral.
- Another factor to be considered is the trigger that led the readers to read the article and the time at which the article was published.
- The time component is also an important factor in the article going viral. However, for this study, time component has not been taken into consideration.

Following table illustrates the features that we created based on the available dataset.

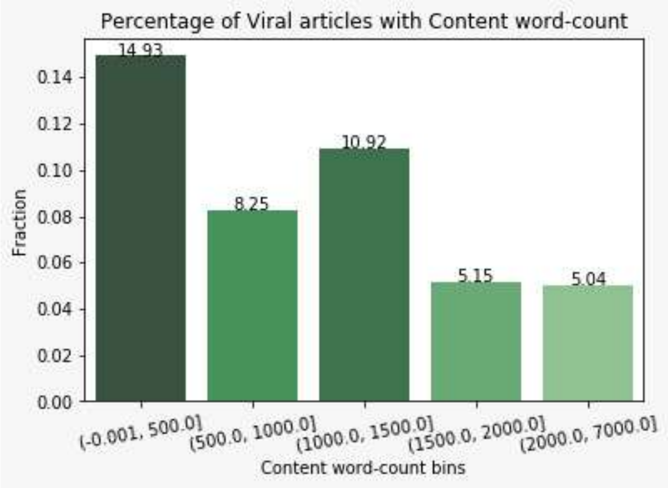
S.No.	Field	Description & Significance
1	Title Word Count	Total count of the words in article title
2	Content Word Count	Total count of words in the article content
3	Read Time Count	Total time needed to read the article
4	Image Count	Total number if images present in the article
5	Title Sentiment Polarity	Sentiment polarity in article title (ranges between -1 to 1)
6	Title Sentiment Subjectivity	Sentiment subjectivity in article title
7	Content Sentiment Polarity	Sentiment polarity in article content (ranges between -1 to 1)
8	Content Sentiment Subjectivity	Sentiment subjectivity in article content
9	Content concept count	Count of concepts representing complexity of an article (Sparse matrix contained 43K tokens which were reduced to 4K tokens using SVD)
10	Weekday	Count of articles published on days of a week

6.1 Analysis of Features

Below is the analysis of various features and their impact on the shareability or virality of an article:

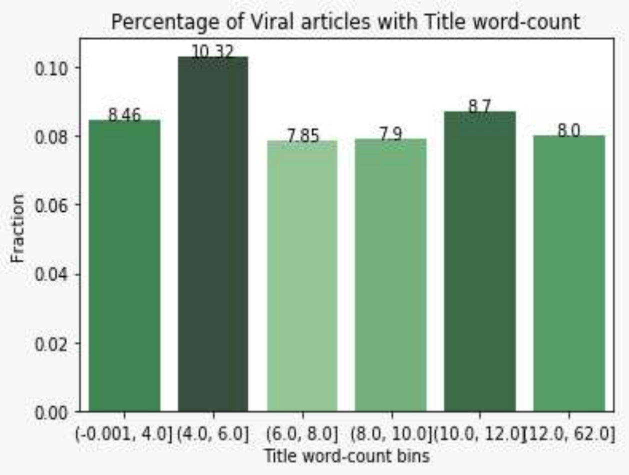
Content word-count v/s Viral

From the below graph, it can be seen that an article is likely to be viral if it is written in a short and crisp manner with number of words being limited to 500 highlighting the reader's short attention spans.



Title word-count v/s Viral

Article had higher chances of going Viral if the title word count lies within the range of four to six.

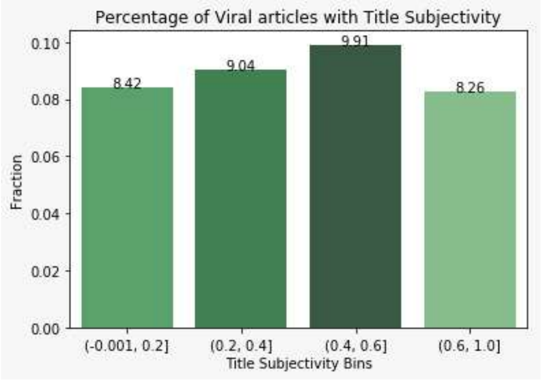


Article sentiments were identified using the TextBlob library in Python which is based on NLTK and Pattern libraries. Textblob.sentiments module consists of two sentiment analysis implementations namely PatternAnalyzer (based on the Pattern library) and NaiveBayesAnalyzer (based on the NLTK library).

The default implementation i.e. PatternAnalyzer has been used for text analysis and sentiment detection in this study.

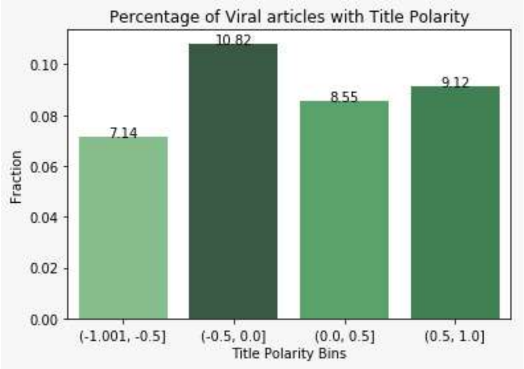
Title sentiment subjectivity

Looking at the graph below, content of the article title should be a balance of both subjective and factual information to become viral.



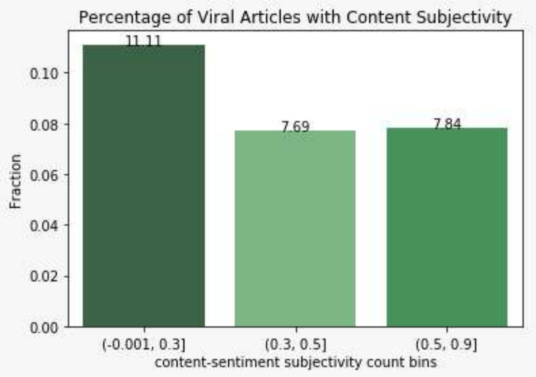
Title sentiment polarity

If title is slightly negative, chances of it going viral are higher.



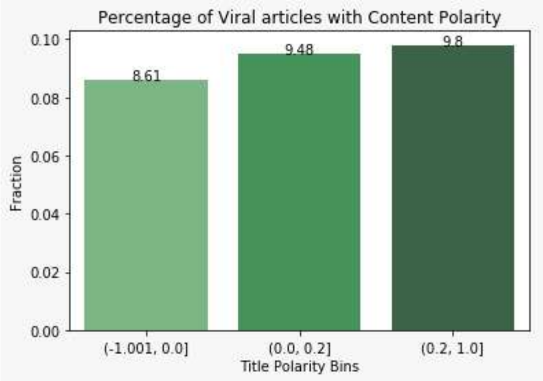
Content Sentiment Subjectivity v/s Viral

The article is likely to go viral if it contains more factual information than subjective information i.e. based on public opinions



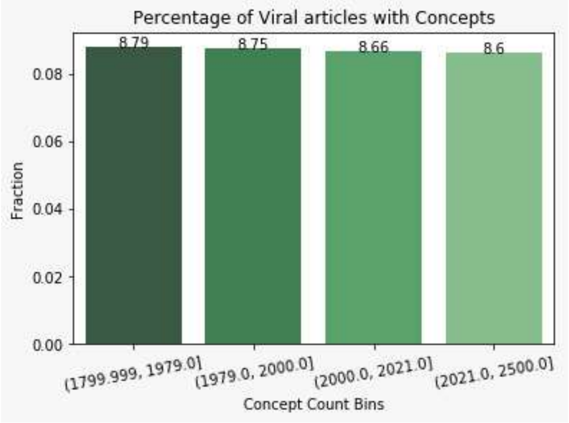
Content Sentiment polarity

The chances of an article going viral is higher if it has positive content.



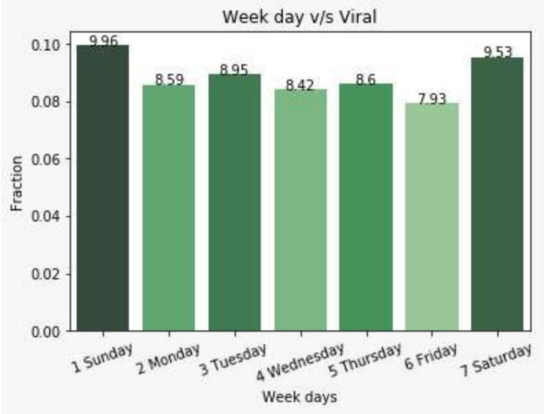
Content concept Count

The number of concepts in an article do not have a major impact on the virality of an article as seen



Day of week

The chances of an article going viral is the highest when it is published on the weekend as can be seen



Read Time Count

The time taken to read an article should be either below 6 minutes or lie between the range of 9 to 12 minutes to have higher chance of becoming viral. However, the chances of an article going viral start reducing beyond 12 minutes of read time.

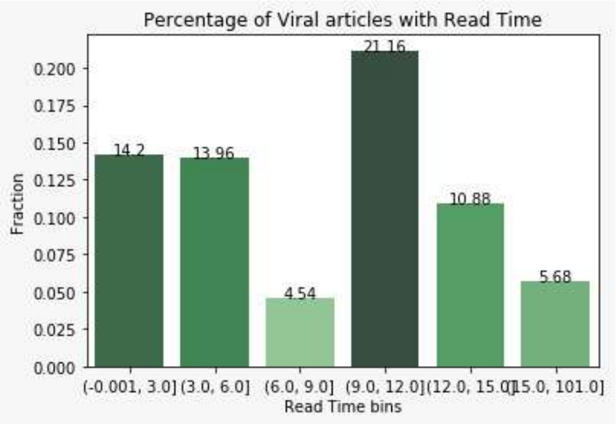
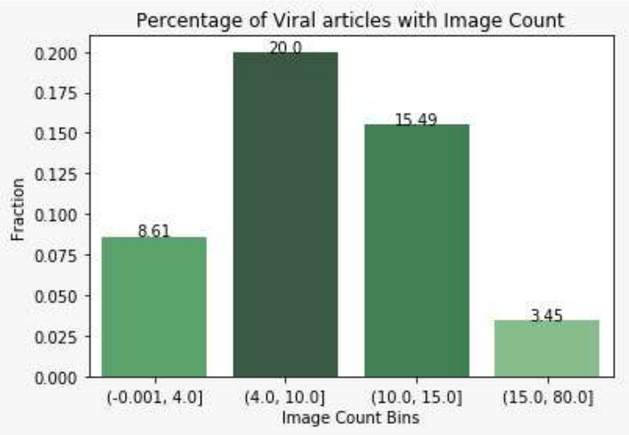


Image Count

The chances of an article going viral is the highest when the number of images it contains lies within the range of 4 to 10. Beyond a count of 15, the percentage of article going viral reduces extensively.



Based on the analysis of the features done above, it can be recommended that for an article to go viral it must satisfy most of the below mentioned properties:

- Article must be crisp i.e. must have words less than 500 words and should not take more than 12 minutes to read
- Title of the article should range between 4 to 6
- Title of the article should have balanced subjectivity, however, the content of the article must have more factual data than subjective data (i.e. contain less public discussions or opinions)
- Title of the article should highlight negative emotion, but the content of the article must convey positive emotions
- Article published on a weekend i.e. Saturday or Sunday have a greater chance of getting viral.
- Number of images in an article should lie within a range of 4 to 10

7 Conclusion

In this age of information overload where thousands of articles are being published online every day, knowing what topics and content interest readers is invaluable to both individual bloggers and publishing agencies.

The tasks performed in this project, which include extracting and transforming unstructured text data to a structured form and applying various text mining techniques and machine learning models generate insights on what constitutes popular categories and content.

These insights can help publishers generate relevant and meaningful content for users, content righters & publishers.