A PROJECT REPORT ON

NEWS RECOMMENDATION SYSTEM

Spring 2020-21/BM69006
Data Science Laboratory
PGDBA Semester 2, IIT Kharagpur

COURSE INSTRUCTOR

Prof. Pabitra Mitra

Submitted By:
Sundeep Samant Singh (20BM6JP51)

PGDBA 2020-2022

Date of Submission: 11th June 2021

INDEX

1. INTRODUCTION:

News is information on various events. It is presentation of new information. News can be delivered through various platforms i.e. word of mouth, printing, electronic communication etc. For generations, News has been typically delivered through paper medium. With the advent of television live news became more prevalent and with the introduction of smartphones and tablets the news started to be delivered directly to the population.

With the advent of digital media and the sheer amount of news in the world its difficult to keep track of everything. As such, people have habituated themselves on reading particular type of topic on a daily basis on several categories be it life style, entertainment, sports etc.

Many news companies have established platforms which recommends news of a particular category based on their browsing and reading pattern. And, with the introduction of machine learning, the task has been simplified as one can read the behaviour pattern of a particular person and recommend the categories of news relative to their interests. However, the data is huge and it becomes a behemoth task for the companies to design more relevant models which can much accurately deliver the categorical news which a person desire.

2. OBJECTIVE:

There has been rise of data being collected from various platform which makes it difficult to recommend news to a different people.

Each person has different reading behaviour and will search for words and topics which are relevant to their interest. Presently, the news short recommends all type of news to the user which creates an anomaly of data resulting in the user skipping the news.

Therefore, there is a need to develop a recommendation system for recommending news to people based on their search and reading pattern. This will result in people actively gaining insights and information.

Hence, we use machine learning to look into this behaviour and browsing pattern of people and recommend them the news based on their preferred category.

3. DATA & DATA DESCRIPTION:

The MIND dataset for news recommendation was collected from anonymized behaviour logs of Microsoft News website. The data randomly sampled 1 million users who had at least 5 news clicks during 6 weeks from October 12 to November 22, 2019. This dataset consists of randomly sampled 50,000 users and their behaviour logs.

The news.tsv file contains the detailed information of news articles. It has 7 columns, which are as follows -

- News ID:
  ID contains the unique ID of news content
- Category:
  Category contains the category of news accessed
- Sub Category:
  Sub category contains the sub category of news accessed
- Title:
  Title of the new contains the heading of news
- Abstract:
  Abstract of the news contains the description of headings i.e. the actual news content
- URL:
  URL contains the website from where the news was accessed
- Title Entities (entities contained in the title of this news):
  Title Entities contains labels of titles searched
- Abstract Entities (entities contained in the abstract of this news):
  Abstract Entities contains labels of titles matched with abstract

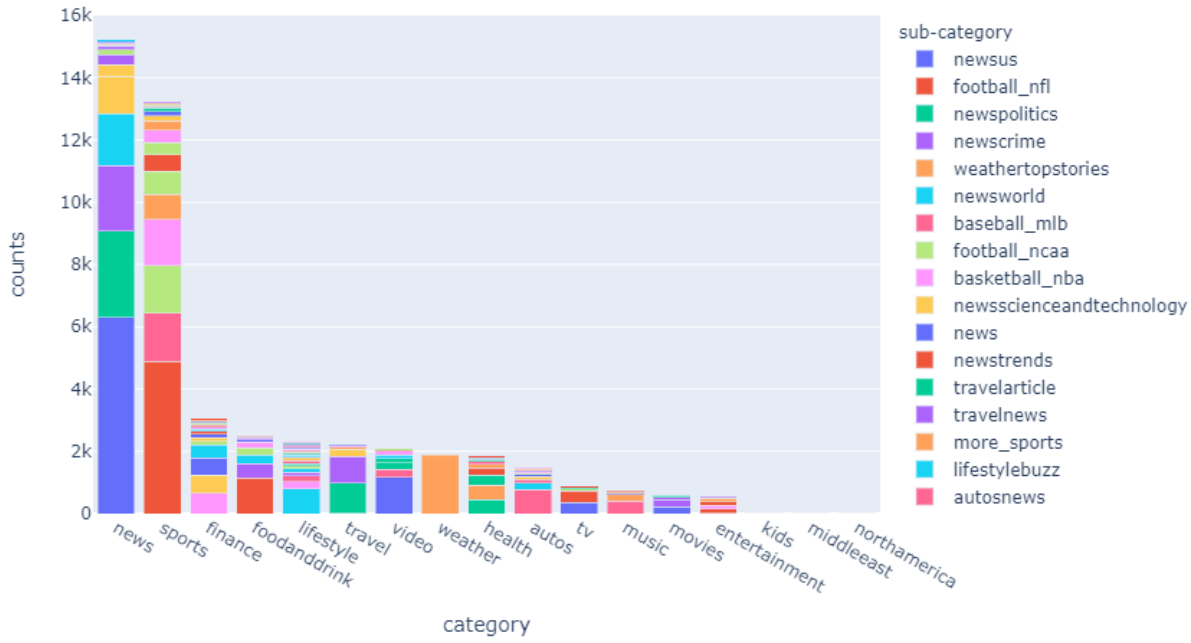
4. DATA EXPLORATION:

The dataset consists of 50,000 rows and 7 columns. The missing values in the columns are as –

Columns	Missing Values
News ID	0
Category	0
Sub Category	0
Title	0
Abstract	2666
URL	0
Title Entities	3
Abstract Entities	4

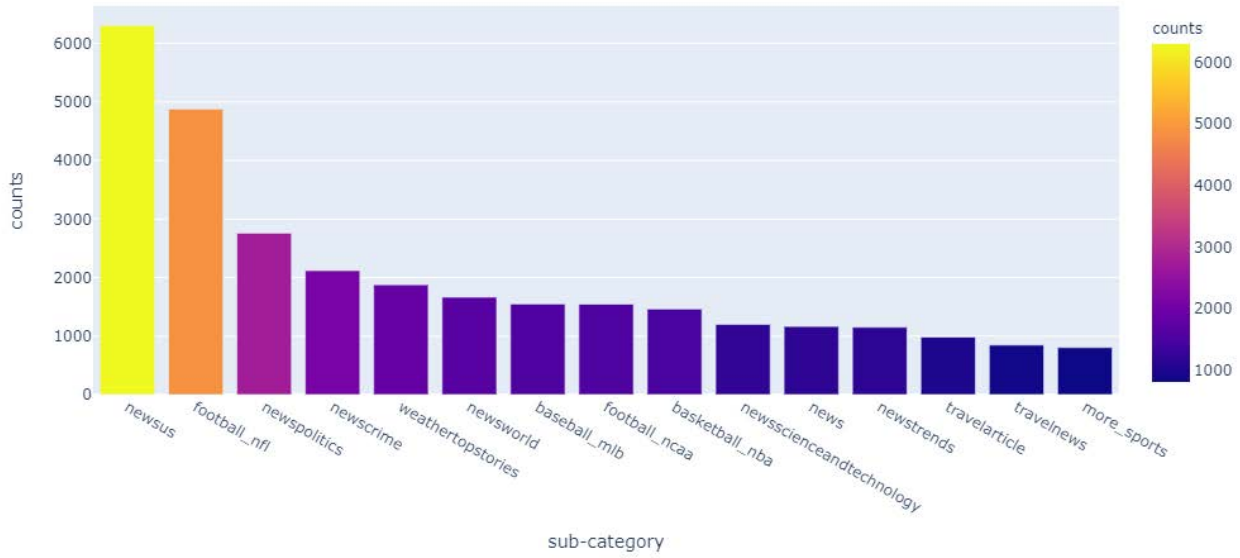
The missing values were dropped as it consist of text data.

Relation between category, sub category, and title was explored. The frequency of category searched by user is as -



It can be observed that news, sports, finance, food and drink and lifestyle are top 5 most searched categories.

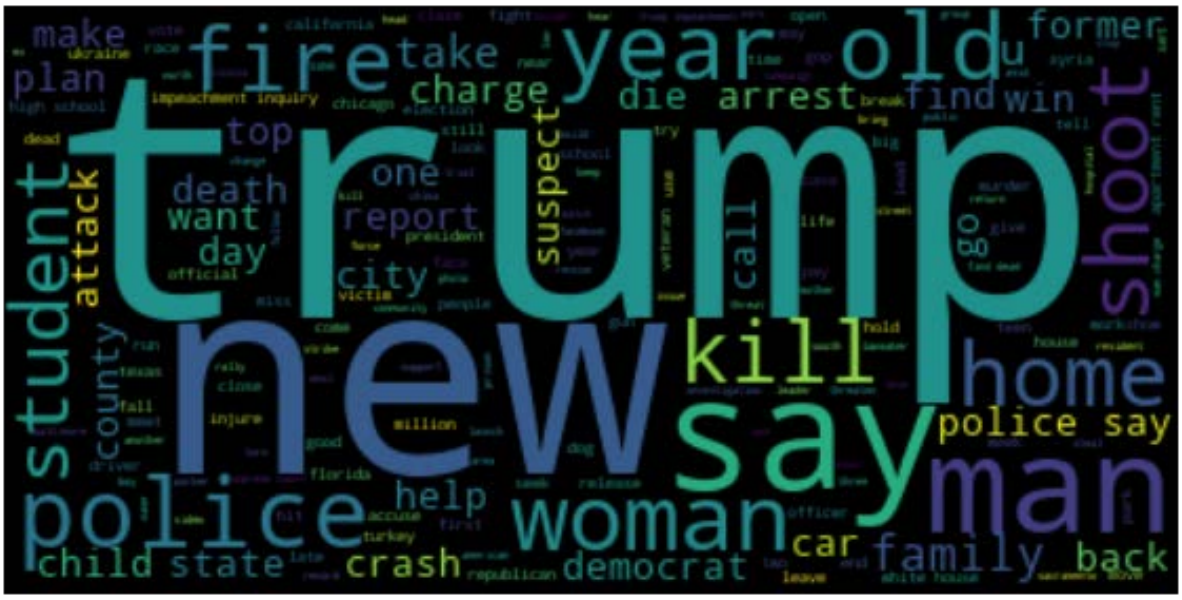
The frequency of sub category searched by user is as –



It can be observed that news us, football_nfl, news politics, news crime and weather top stories are the top 5 searched news sub categories.

Now, the frequency of words in title searched for top 5 category is as –

a) News

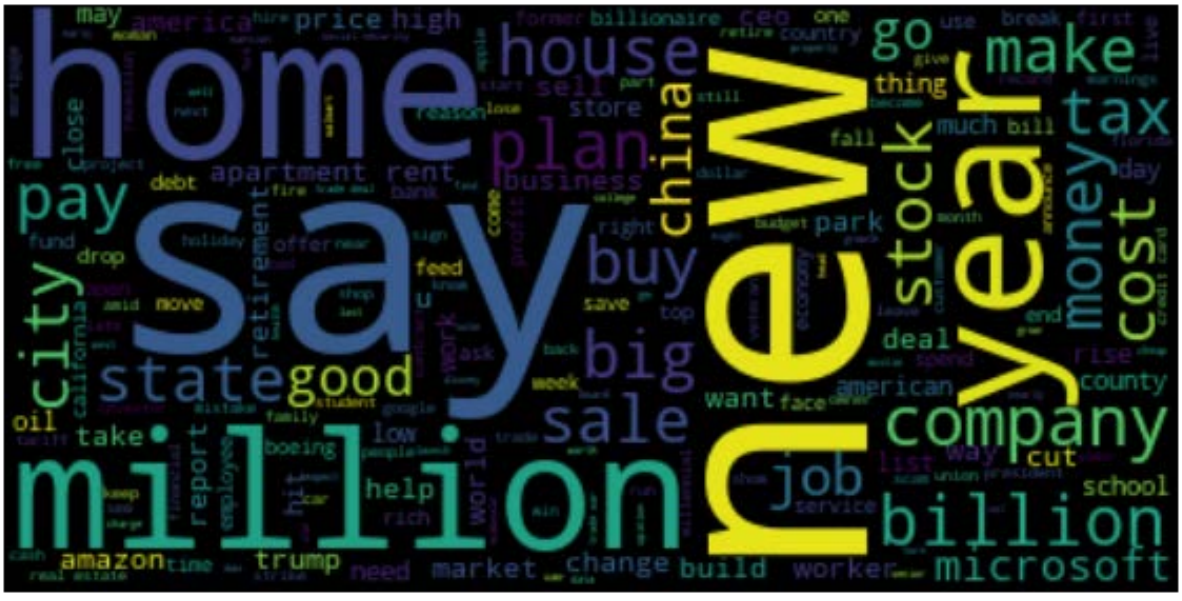


It can be observed that words like trump, new, say etc. are the top searched words in the news category.

b) Sports -

It can be observed that win, play, season, watch, week etc, are the top most searched words in the sports category.

c) Finance -

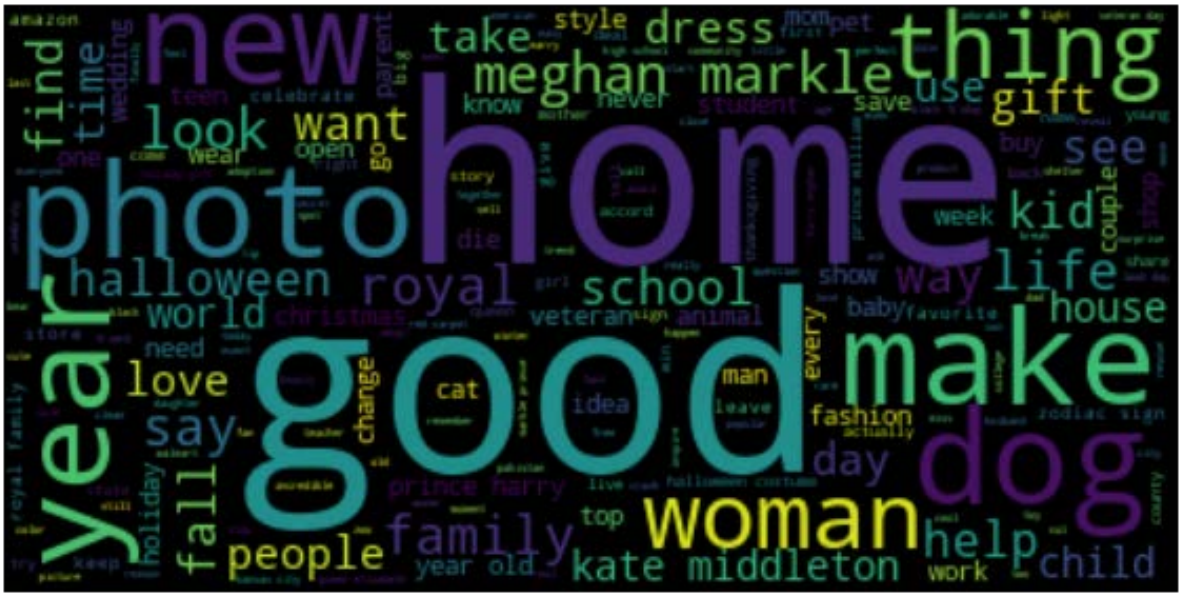


It can be observed that new, million, home, year, stock, billion, China etc. were among the top searched words for the finance category.

d) Food and Drink –

It can be observed that restaurant, good, make, recipe, food etc. were among the top searched words in food and drink category.

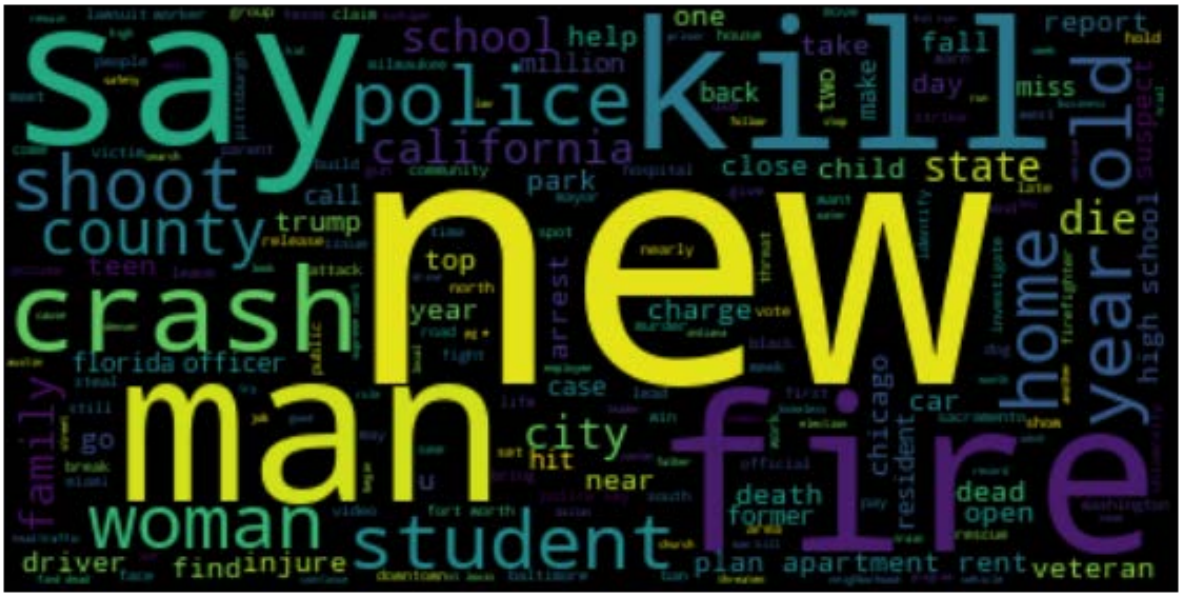
e) Lifestyle -



It can be observed that home, good, year, photo, woman etc. were among the top most searched words in the lifestyle category.

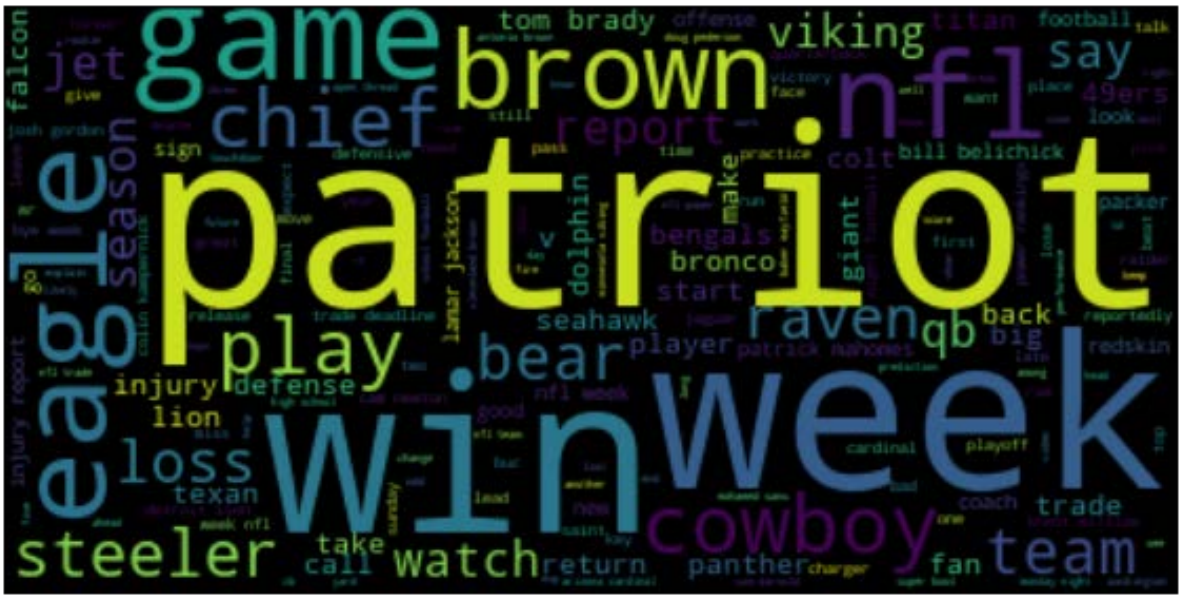
Similarly, the frequency of words in title for sub-category is as –

a) News US –



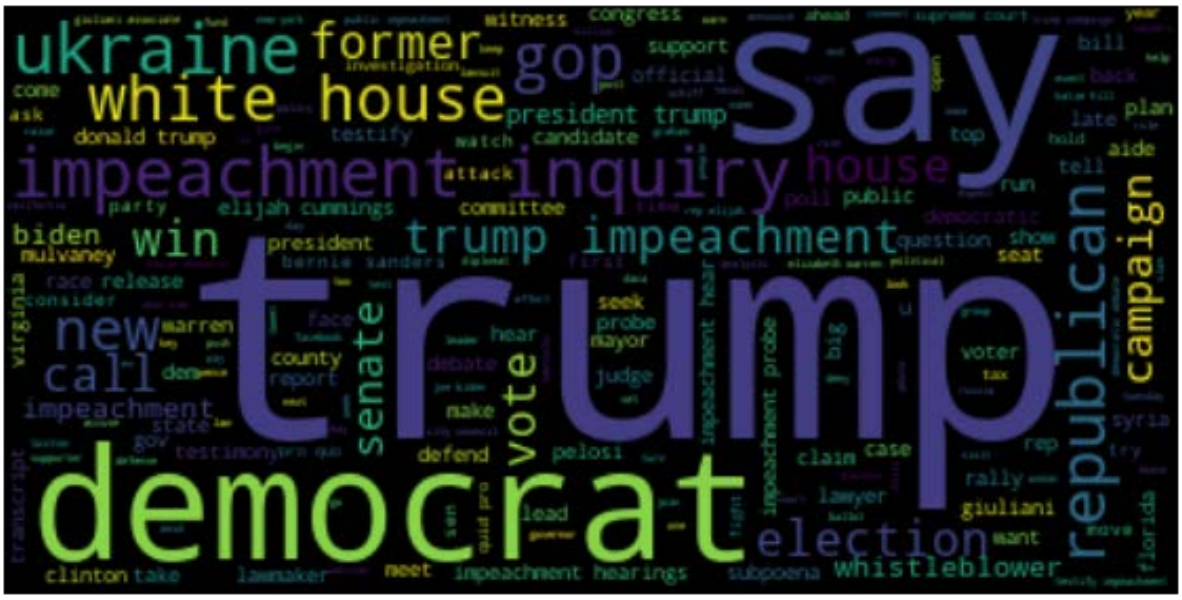
It can be observed that new, man, kill, fire, say, police etc. were the top searched words for News US sub-category.

b) Football NFL -



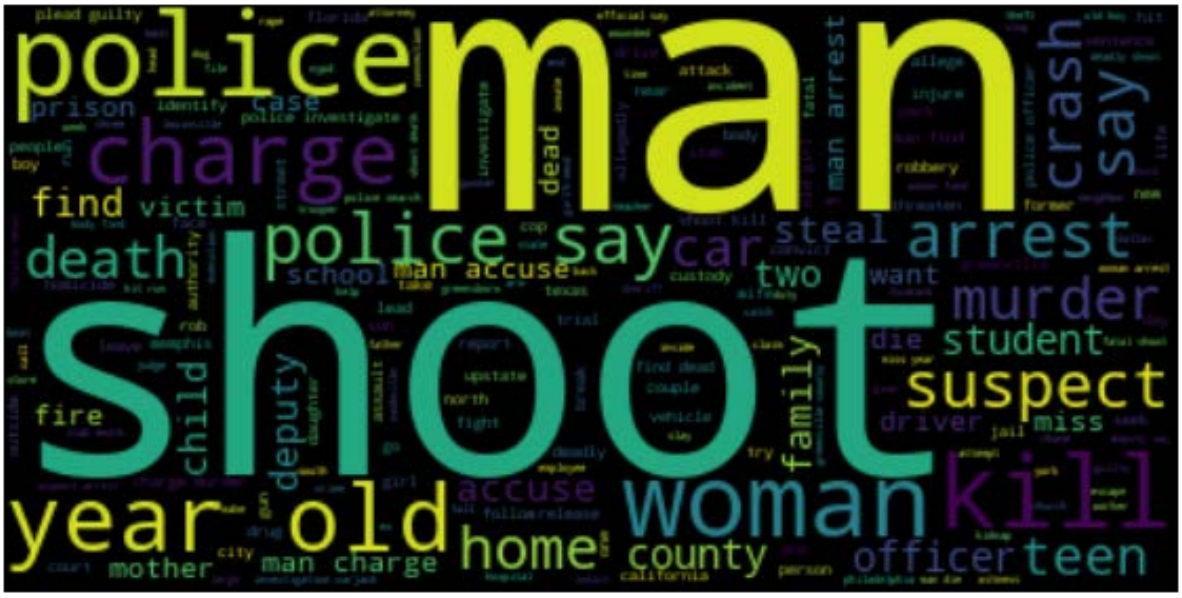
It can be observed that patriot, win, week etc. were some of the top words to be searched in Football NFL sub-category.

c) News Politics -



It can be observed that democrat, trump, Ukraine, say, republican etc. were the top words searched in News Politics sub category.

d) News Crimes -

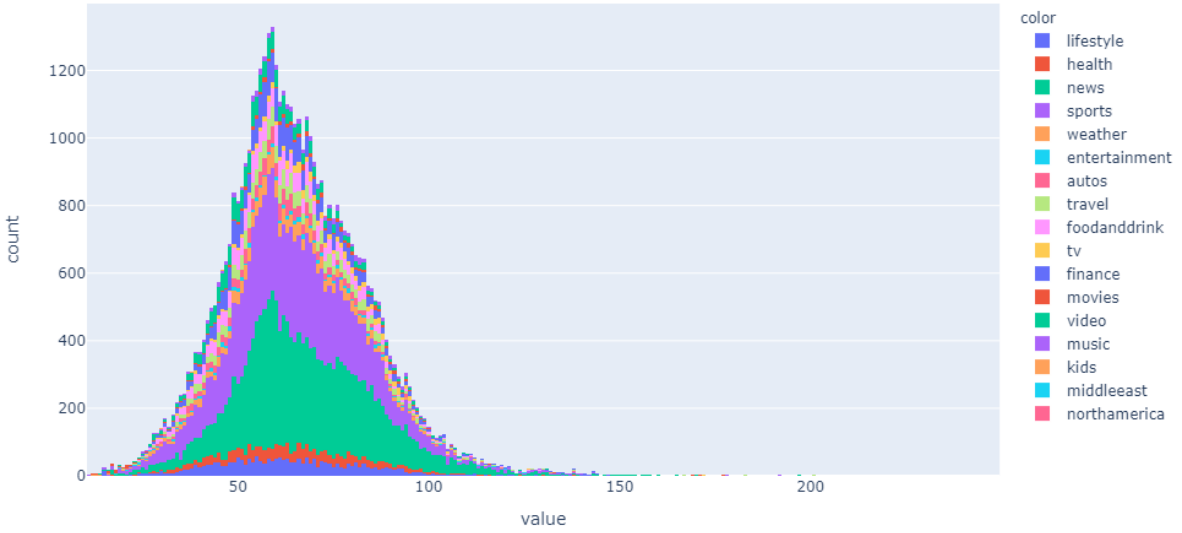


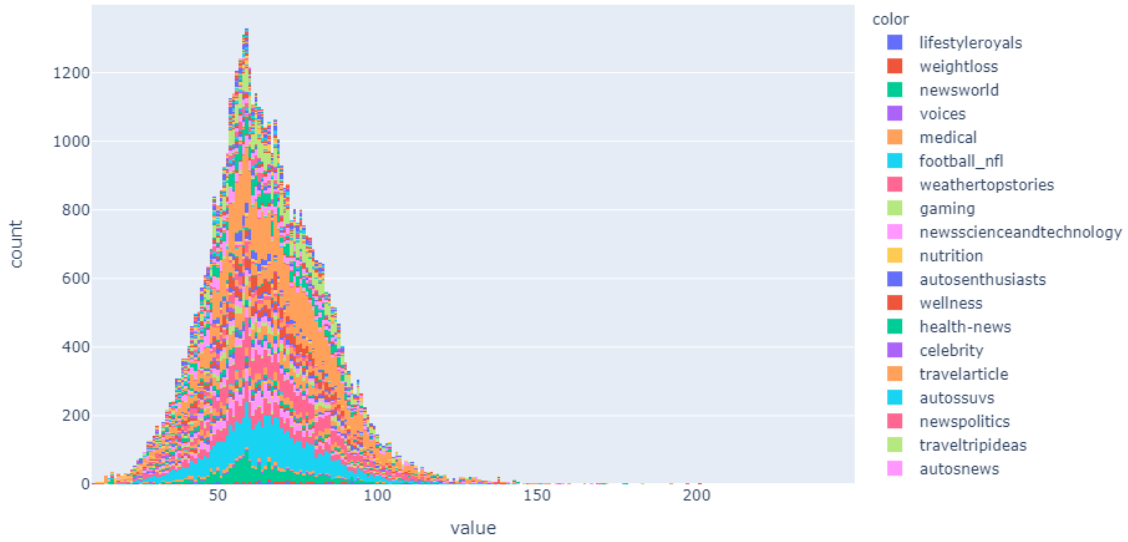
It can be observed that shoot, man, police, suspect etc. were some of the common words searched in News Crimes sub-category.

e) Weather Top Stories -

It can be observed that words like snow, fire, cold, storm, etc. were some of the common words searched in weather top stories sub category.

Now, looking into the word count in each title -



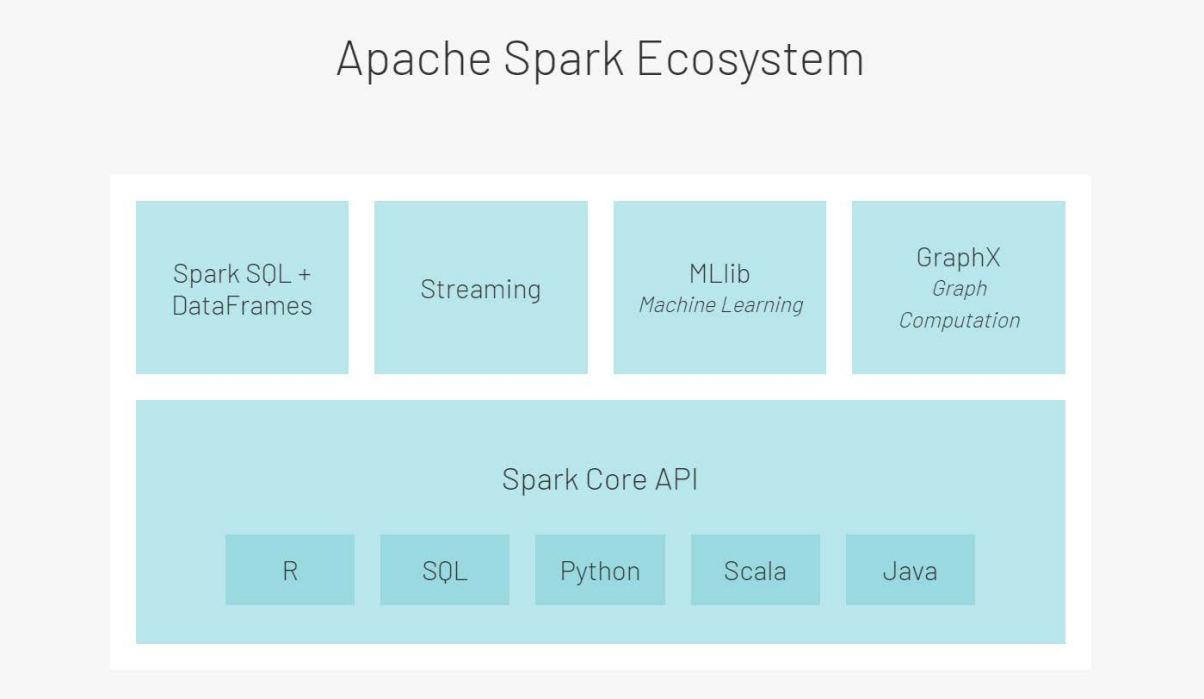


It can be observed that the title is usually in the length of 10 – 130 words.

5. METHODOLOGY:

The data available with us is text data. Hence, we have to tokenize, remove stop words, lemmatize and vectorize in order to train and test the data. The above task was accomplished using Spark.

Spark is a lightning-fast unified analytics engine for big data and machine learning. It was originally developed at UC Berkeley in 2009.



The benefits of using Spark are –

- Speed:
  Spark is 100x faster than other modules for large scale data processing by exploiting in memory computing and other optimizations. Spark is also fast when data is stored on disk
- Ease of Use:
  Spark has easy-to-use APIs for operating on large datasets and includes a collection of over 100 operators for transforming data and familiar data frame APIs for manipulating semi-structured data.
- Unified Engine:
  Spark has higher-level libraries, including support for SQL queries, streaming data, machine learning and graph processing.

Tokenization –

Tokenization is the process of taking text and breaking it into individual terms. The example below shows how a sentence is split into sequences of words.

Input - The quick brown fox jumps over the lazy dog

Output – 'The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog'

Stop Words Removal –

Stop words are words which should be excluded from the input, typically because the words appear frequently and don't carry as much meaning.

Input – I, saw, the, red, balloon

Output – saw, red, balloon

Lemmatization –

Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma.

Input – the boy's cars are different colours

Output - the boy car be differ color

TF-IDF Vectorizer –

Term frequency-inverse document frequency (TF-IDF) is a feature vectorization method widely used in text mining to reflect the importance of a term to a document in the corpus.

Denote a term by t, a document by d, and the corpus by D. Term frequency $TF(t,d)$ is the number of times that term t appears in document d, while document frequency $DF(t,D)$ is the number of documents that contains term t.

If we only use term frequency to measure the importance, it is very easy to over-emphasize terms that appear very often but carry little information about the document, e.g. "a", "the", and "of". If a term appears very often across the corpus, it means it doesn't carry special information about a particular document. Inverse document frequency is a numerical measure of how much information a term provides:

$$IDF(t, D) = \log \frac{|D| + 1}{DF(t, D) + 1},$$

where |D| is the total number of documents in the corpus. Since logarithm is used, if a term appears in all documents, its IDF value becomes 0. A smoothing term is applied to avoid dividing by zero for terms outside the corpus. The TF-IDF measure is simply the product of TF and IDF:

$$TFIDF(t, d, D) = TF(t, d) \cdot IDF(t, D).$$

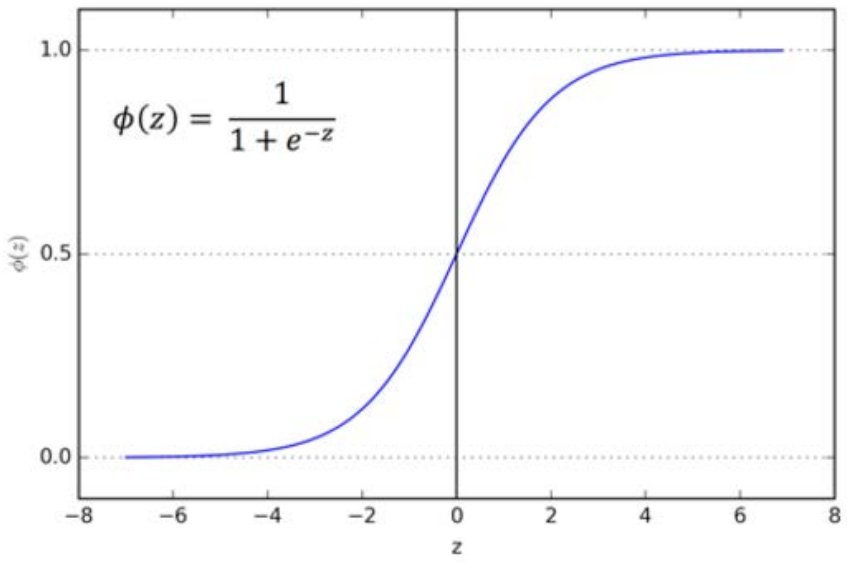
Logistic Regression –

Logistic regression is named for the function used at the core of the method, the logistic function.

The logistic function, also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

1 / (1 + e^-value)

where e is the base of the natural logarithms (Euler's number or the EXP () function in your spreadsheet) and value is the actual numerical value that you want to transform. Below is a plot of the numbers between -5 and 5 transformed into the range 0 and 1 using the logistic function.



Extra Trees –

Extremely Randomized Trees, or Extra Trees for short, is an ensemble machine learning algorithm.

It is an ensemble of decision trees and is related to other ensembles of decision trees algorithms such as bootstrap aggregation (bagging) and random forest.

The Extra Trees algorithm works by creating a large number of unpruned decision trees from the training dataset. Predictions are made by averaging the prediction of the decision trees in the case of regression or using majority voting in the case of classification.

- **Regression**: Predictions made by averaging predictions from decision trees.
- **Classification**: Predictions made by majority voting from decision trees.

Unlike bagging and random forest that develop each decision tree from a bootstrap sample of the training dataset, the Extra Trees algorithm fits each decision tree on the whole training dataset.

Like random forest, the Extra Trees algorithm will randomly sample the features at each split point of a decision tree. Unlike random forest, which uses a greedy algorithm to select an optimal split point, the Extra Trees algorithm selects a split point at random.

As such, there are three main hyperparameters to tune in the algorithm; they are the number of decision trees in the ensemble, the number of input features to randomly select and consider for each split point, and the minimum number of samples required in a node to create a new split point.

The random selection of split points makes the decision trees in the ensemble less correlated, although this increases the variance of the algorithm. This increase in variance can be countered by increasing the number of trees used in the ensemble.

XG Boost –

Gradient boosting refers to a class of ensemble machine learning algorithms that can be used for classification or regression predictive modelling problems.

Ensembles are constructed from decision tree models. Trees are added one at a time to the ensemble and fit to correct the prediction errors made by prior models. This is a type of ensemble machine learning model referred to as boosting.

Models are fit using any arbitrary differentiable loss function and gradient descent optimization algorithm. This gives the technique its name, "*gradient boosting*," as the loss gradient is minimized as the model is fit, much like a neural network.

Extreme Gradient Boosting, or XGBoost for short is an efficient open-source implementation of the gradient boosting algorithm. As such, XGBoost is an algorithm, an open-source project, and a Python library.

It was initially developed by Tianqi Chen and was described by Chen and Carlos Guestrin in their 2016 paper titled “XGBoost: A Scalable Tree Boosting System.”

It is designed to be both computationally efficient and highly effective. The two main reasons to use XGBoost are execution speed and model performance. Generally, XGBoost is fast when compared to other implementations of gradient boosting.

Light GBM –

Gradient boosting refers to a class of ensemble machine learning algorithms that can be used for classification or regression predictive modelling problems.

Ensembles are constructed from decision tree models. Trees are added one at a time to the ensemble and fit to correct the prediction errors made by prior models. This is a type of ensemble machine learning model referred to as boosting.

Models are fit using any arbitrary differentiable loss function and gradient descent optimization algorithm. This gives the technique its name, “*gradient boosting*,” as the loss gradient is minimized as the model is fit, much like a neural network.

Light Gradient Boosted Machine, or LightGBM for short, is an open-source implementation of gradient boosting designed to be efficient and perhaps more effective than other implementations.

LightGBM was described by Guolin Ke, et al. in the 2017 paper titled “LightGBM: A Highly Efficient Gradient Boosting Decision Tree.” The implementation introduces two key ideas: GOSS and EFB.

Gradient-based One-Side Sampling, or GOSS for short, is a modification to the gradient boosting method that focuses attention on those training examples that result in a larger gradient, in turn speeding up learning and reducing the computational complexity of the method.

Exclusive Feature Bundling, or EFB for short, is an approach for bundling sparse (mostly zero) mutually exclusive features, such as categorical variable inputs that have been one-hot encoded. As such, it is a type of automatic feature selection.

Together, these two changes can accelerate the training time of the algorithm by up to 20x. As such, LightGBM may be considered gradient boosting decision trees (GBDT) with the addition of GOSS and EFB.

Stacking –

Stacked Generalization, or stacking for short, is an ensemble machine learning algorithm.

Stacking involves using a machine learning model to learn how to best combine the predictions from contributing ensemble members.

In voting, ensemble members are typically a diverse collection of model types. Predictions are made by averaging the predictions, such as selecting the class with the most votes (the statistical mode) or the largest summed probability.

An extension to voting is to weigh the contribution of each ensemble member in the prediction, providing a weighted sum prediction. This allows more weight to be placed on models that perform better on average and less on those that don't perform as well but still have some predictive skill.

The weight assigned to each contributing member must be learned, such as the performance of each model on the training dataset or a holdout dataset.

The model that combines the predictions is referred to as the meta-model, whereas the ensemble members are referred to as base-models.

In the language taken from the paper that introduced the technique, base models are referred to as level-0 learners, and the meta-model is referred to as a level-1 model. Naturally, the stacking of models can continue to any desired level.

Importantly, the way that the meta-model is trained is different to the way the base-models are trained.

The input to the meta-model are the predictions made by the base-models, not the raw inputs from the dataset. The target is the same expected target value. The predictions made by the base-models used to train the meta-model are for examples not used to train the base-models, meaning that they are out of sample.

For example, the dataset can be split into train, validation, and test datasets. Each base-model can then be fit on the training set and make predictions on the validation dataset. The predictions from the validation set are then used to train the meta-model. This means that the meta-model is trained to best combine the capabilities of the base-models when they are making out-of-sample predictions, e.g. examples not seen during training.

Once the meta-model is trained, the base models can be re-trained on the combined training and validation datasets. The whole system can then be evaluated on the test set by passing examples first through the base models to collect base-level predictions, then passing those predictions through the meta-model to get final predictions. The system can be used in the same way when making predictions on new data.

This approach to training, evaluating, and using a stacking model can be further generalized to work with k-fold cross-validation.

Typically, base models are prepared using different algorithms, meaning that the ensembles are a heterogeneous collection of model types providing a desired level of diversity to the predictions made. However, this does not have to be the case, and different configurations of the same models can be used or the same model trained on different datasets.

On classification problems, the stacking ensemble often performs better when base-models are configured to predict probabilities instead of crisp class labels, as the added uncertainty in the predictions provides more context for the meta-model when learning how to best combine the predictions.

The meta-model is typically a simple linear model, such as a linear regression for regression problems or a logistic regression model for classification. Any machine learning model can be used as the meta learner.

6. RESULTS:

The dataset was modelled as multi class and multi output multi class. The multi class consists of Category class output for recommending news of those categories. The multi output multi class consists of both category and sub-category for a much better recommendation of sub-category along with category.

Multi class –
The dataset was split into train and test set with 80% as train set and 20% as test set. Train model was vectorized and the vocabulary were used to vectorize the test set. Then, different models were used with different parameters. The results were –

Model	Accuracy for Train Set	Accuracy for Test Set
Logistic Regression	0.8086	0.6983
XG Boost	0.7616	0.6157
Extra Trees	0.9983	0.6933
Light GBM	0.7089	0.6024
Stacking	0.9961	0.7208

Multi Output Multi Class –
Similar to multi class, the dataset was again split into train and test set with 80% as train set and 20% as test set. The vocabulary of train models was used to vectorize the test set. Then, different models were used with different parameters. The results were –

Model	Accuracy for Train Set	Accuracy for Test Set
Logistic Regression	0.5975	0.4853
XG Boost	0.6208	0.4091
Extra Trees	0.9965	0.4793

Due to limited computational power of machine, Light GBM and Stacking were unable to run.

7. DISCUSSIONS:

With the resources and models currently available at our hand, it can be concluded from the results that recommending news articles based on words searched is a difficult task.

The multi class model has train set accuracy of 99% to 70% and test set accuracy of 72% to 60%. The logistic regression performed best with this model. The train set accuracy for logistic regression was 81% and test set accuracy was 70%. Stacking led to improvement in result but is overfitting a lot. Hence, for multi class recommendation we can utilize the logistic regression model.

The multi output multi class model has train set accuracy of 99% to 59% and test set accuracy of 48% to 40%. Due to limited computational power of our system, we were unable to run Light GBM and Stacking in the model. In multi output multi class also logistic regression has the best accuracy.

Hence, from the results we can conclude that from the available resources at hand we can only recommend news to our customer base on category classification with accuracy of 70% which can be considered satisfactory. However, recommending news based on sub-category along with category is a difficult task.

Numerous developments and research are on-going to set a base model to more accurately recommend news based on the customer search pattern. More relevant news will result in customers attentively gaining information and insights. The Microsoft MIND Dataset plans to achieve the same with various organisation and institutions currently working in this dataset to generate a model to better predict the output.

Therefore, with the help of machine learning we can study the behaviour and reading pattern of customers and recommend them the news that they are much more interested in.

8. REFERENCES:

- NEWS - https://en.wikipedia.org/wiki/News
- MIND DATASET - https://www.kaggle.com/arashnic/mind-news-dataset
- Introduction to Spark - https://databricks.com/spark/about
- Features of Spark - https://spark.apache.org/docs/latest/ml-features
- Logistic Regression Algorithm - https://machinelearningmastery.com/logistic-regression-for-machine-learning/
- Extra Trees Algorithm - https://machinelearningmastery.com/extra-trees-ensemble-with-python/
- XGBoost Algorithm - https://machinelearningmastery.com/extreme-gradient-boosting-ensemble-in-python/
- Light GBM Algorithm - https://machinelearningmastery.com/light-gradient-boosted-machine-lightgbm-ensemble/
- Stacking Algorithm - https://machinelearningmastery.com/essence-of-stacking-ensembles-for-machine-learning/