

Personalised News Recommendation Using Machine Learning

A

Project Report

submitted

in partial fulfillment

for the award of the Degree of

Bachelor of Technology

In Department of Computer Science & Engineering

(with specialization in Computer Engineering)



SESSION 2018-2019

Submitted To:

Mr. Basant Agarwal
Asso. Prof., Department of CS

Supervisor:

Mr. Vinay Kanungo
Asst. Prof, Department of CS

Submitted By:

Rahul Saini 15ESKCS138
Prachi Yadav 15ESKCS130
Pramod Kr 15ESKCS132
Riya Ritu 15ESKCS147

Department of Computer Science & Engineering

Swami Keshvanand Institute of Technology, Management & Gramothan, Jaipur

Rajasthan Technical University

CANDIDATE’S DECLARATION

I hereby declare that the work, which is being presented in the Project Report entitled “**News Recommendation Using Machine Learning**” in partial fulfilment for the award of Degree of “**Bachelor of Technology**” in Department of Computer Science & Engineering with specialization in Computer Engineering and submitted to the **Department of Computer Science & Engineering, Swami Keshvanand Institute of Technology, Management and Gramathan, Jaipur**, affiliated to **Rajasthan Technical University**, Kota is a record of my own investigations carried under the Guidance of **Mr. Vinay Kanungo** , Asst. Professor, Department of Computer Science & Engineering, Swami Keshvanand Institute of Technology, Management and Gramathan, Jaipur.

I have not submitted the matter presented in this Seminar report anywhere for the award of any other Degree.

Rahul Saini

Roll No: 15ESKCS138

Prachi Yadav

Roll No: 15ESKCS130

Pramod Kumar

Roll No: 15ESKCS132

Riya Ritu

Roll No: 15ESKCS147

B.Tech. (Computer Science & Engineering)
Swami Keshvanand Institute of Engineering & Technology, Jaipur

Counter Signed by

Mr. Vinay Kanungo

Asst. Professor

Department of Computer Science & Engineering

Swami Keshvanand Institute of Technology, Management and Gramathan, Jaipur

ACKNOWLEDGEMENT

I feel immense pleasure in expressing my regards to the Chairman **Mr. Surja Ram Meel**, Director **Mr. Jaipal Meel**, Registrar **Mrs. Rachana Meel**, Director (Academics) **Prof. (Dr.) S. L. Surana**, Principal & Director (D&W) **Prof. (Dr.) S. K. Calla** Swami Keshvanand Institute of Technology, Management & Gramothan, Jaipur for providing me necessary facilities during the various stages of this work.

I would like to thank **Dr. C. M. Choudhary**, Professor & Head, Department of Computer Science & Engineering and **Dr. Anil Chaudhary**, Professor & Head, Department of Information Technology for provide me opportunity to work in consistent direction and providing their valuable suggestions to improve thesis writing. name

I would like to thank my esteem guide **Mr. Vinay Kanungo**, Assistant Professor, Department of Computer Science & Engineering, Swami Keshvanand Institute of Technology, Management & Gramothan, Jaipur for his valuable guidance, keen interest, constant encouragement, incessant inspiration and continuous help throughout this work. Especially I acknowledge his support when I was stuck and he is suggesting me new ideas to solve the problems. His vast experience and realistic approach have been of great help to me. I am honored having **Mr. Basant Agarwal** as my project report supervisor. His excellent guidance has been instrumental in making this work a success.

I would also like to express my thanks to my parents for their support and blessings. In addition, a very special thanks to all my colleagues and friends for their support in the completion of this work.

Rahul Saini

Roll No: 15ESKCS138

Prachi Yadav

Roll No: 15ESKCS130

Pramod Kumar

Roll No: 15ESKCS132

Riya Ritu

Roll No: 15ESKCS147

B.Tech. (Computer Science & Engineering)
Swami Keshvanand Institute of Engineering & Technology, Jaipur

TABLE OF CONTENTS

Candidate Declaration.....	2
Acknowledgement.....	3
Contents.....	4
Figures and Tables.....	6
Abstract.....	7
1. Introduction.....	8
1.1 Definition	9
1.2 Introduction to Recommender System.....	9
1.3 Why Recommender System.....	9
1.4 Phases of Recommender System.....	11
1.5 Filtering Techniques	13
1.6 Websites using Recommendation System.....	21
1.7 Advantage of Recommendation System.....	21
1.8 Problem with Recommendation System.....	21
2. Literature Survey.....	22
2.1 Text Classification Algorithms.....	22
2.1.1 Naïve Bayes Classifier.....	22
2.1.2 SVM classifier	26
2.1.2 KNN Classifier	27
2.1.4 Logistic Regression Classifier.....	24
2.1.5 Decision Tree Algorithm.....	31
2.1.6 Random forest Algorithm.....	33
2.2 Text Similarities	35
2.2.1 Similarity Measures.....	35
3. Modelling and system implementation.....	36
3.1 Back end.....	36
3.1.1 Data Collection.....	36
3.1.2 Splitting the data.....	37
3.1.3 Tokenization, stopwords removal.....	37

3.1.4 Fitting the category variable as follow.....	37
3.1.5 Classifier Model Creation.....	37
3.1.6 Fitting the classifier on training data.....	38
3.1.7 Making the prediction and generating the classifier report.....	38
3.1.8 Prediction Score.....	38
3.1.9 Most Efficient Classifier.....	41
3.1.10 Comparison Chart of classifier.....	41
3.1.11 Flask.....	41
3.2 Front-end	42
3.2.1 Bootstrap.....	42
3.2.2 JavaScript.....	42
3.2.3 HTML5.....	42
3.2.4 CSS3.....	42
4. Output and Result.....	43
4.1 News Headline Website Interface.....	43
4.2 Recommendation of News.....	44
5. Conclusion.....	51
6. Future Work.....	52
5. Bibliography.....	53

LIST OF FIGURES AND TABLES

Figure 1.1: Exponential Growth of Data.....	9
Figure 1.2: Recommender System vs Searching	10
Figure 1.3: Recommendation Phases.....	11
Figure 1.4: Types of Recommender System.....	13
Figure 1.5: Collaborative Filtering.....	17
Figure 1.6: Long Tail problem.....	19
Figure 2.1: Non-linear classification using SVM.....	24
Figure 2.2: KNN classification with K=1	27
Figure 2.3 Select an initial data point	29
Figure 2.4: calculate the distance from each data point.....	28
Figure 2.5: vote for class.....	29
Figure 2.6 graph showing sigmoid function.....	30
Figure 2.7 Decision Tree.....	31
Figure 2.8 Process of Decision Tree.....	32
Figure 2.9 Random Forest Algorithm.....	34
Figure 3.1 Performance of various classifiers.....	41
Figure 3.2 Flask work flow using MVC	42
Table 1.1: TFIDF data table	15
Table 1.2: user-user similarity Collaborative filtering.....	18
Table 1.3 item-item similarity Collaborative filtering.....	19
Table 2.1 Training data for Naïve bayes classifier example.....	25

ABSTRACT

Online news reading has become very popular as the web provides access to news articles from millions of sources around the world. But a key challenge of news app is to find out the news that is relevant to a particular user. In this paper, we represent our report on developing personalized news recommendation application called “**Headlines**”. We use the combination of multiple ML classifiers for finding the category in which user may be interested and assume the category on which most of the classifier are agree. Then we fetch the news of that predicted category. Then we find the most similar news among them to users previously selected news using the cosine similarity. We then sort the news according to the similarity score after which we recommend the news to the user.

Keywords— News, Personalization, collaborative filtering, news trend

CHAPTER-1

INTRODUCTION

1.1 Definition

A **Recommender System** is a subclass of information filtering system that recommends something based on the users interest. Information personalization or recommendation based on user profile or interest is one of the solutions to these problems and it aims to overcome the aforementioned problems.

1.2 Introduction to Recommender System

While reading any online news we are habitual to say “hey show me Today’s headlines”, we don’t put the news relevant keywords or our preferences on the news websites. The recommendation system behind the website automatically figure out our preference of our previous browsing history, current location, click stream analysis and any other Informations. Recommender systems are a type of information filtering system that aim to provide recommendations to users of items in which they will likely be interested, as they are selected from a set of choices. Recommendation engines help users to discover the space of options and multiple domains in a specific context, such as in e-commerce, cinema, or music based on their past behavior in an application or website in relation to other users. Book suggestions on Amazon, movie recommendations on Netflix, and news article recommendations on Google News are great real-world, industry-level examples of recommender systems. Users on Netflix receive recommendations by providing a rating on movies on a scale of one (disliked) to five (liked) stars. Similarly, on Google News users receive recommendations based on their search preferences on Google Search Engine. Such systems may keep track of user-specific attributes, such as age range, gender, and location, and item-specific attributes, such as item descriptions, demographic data, categories, and relations between these users and items on a website or application, to provide quality recommendations. These data are used during the recommendation process to identify relations between users and/or items; then, based on these relations, recommendations are generated.

We are leaving in the age of information and entering the age of recommendation.

1.3 Why Recommender System?

We are overloaded with data e.g. Billions of web pages on Google, Millions of Facebook users, Millions of YouTube videos, Thousands of News Articles and blogs posts each day. We use the term '**big data**' to refer to this overloaded data. The biggest growing issue htat needs to be deal with now is finding the right information from exponentially growing data and providing a personalized experience to users. Big data refers to data sets whose size is beyond the ability of typical database software tools to capture, store, manage and analyze.

Exponential Growth in Big data:-

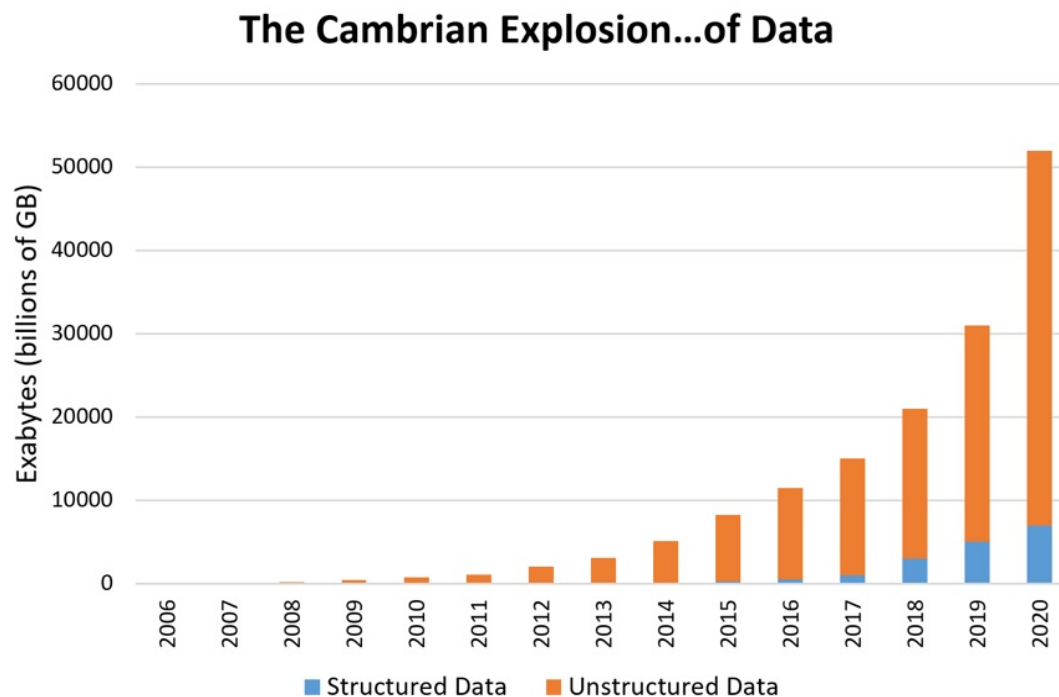


Fig1.1. Exponential Growth of Data

Big data is data that exceeds the processing capacity of conventional database systems. The data is too big, moves too fast, or does not fit the strictures of your database architectures. To gain value from this data, you must choose an alternative way to process it. Twitter processes 1 Petabyte of data daily while Google processes 100 Petabyte data.

At the user side, Big data causes the problem while searching the product of our interest for example let's say we want to search for a news on news website of our interest. While reading any

online news we are habitual to say “hey show me Today’s headlines”, we don’t put the news relevant keywords or our preferences on the news websites. The recommendation system behind the website automatically figure out our preference of our previous browsing history, current location, click stream analysis and any other Informations. We are leaving in the age of information and entering the age of recommendation.

Search:



Recommend:



While searching an item user have to pick / search the item manually based on his interest whereas in recommendation systems items are recommended to the user based on his past behavior. For example Google search is one of the example of Recommender System whereas Wikipedia Search is not. Best example to show the need of recommender system is following:

E.g.1. Google Search is a Recommender System whereas Wikipedia Search is not.

The image shows two side-by-side search results for the query "recommender system tutorial".

Left: Google Search Results

- Search bar: "recommender system tutorial"
- Results: About 4,75,00,000 results (0.48 seconds)
- Top results include:
 - "What Is Recommendation System - Find the Top Results Here" (downloadsearch.cnet.com)
 - "Recommender Systems in Python Tutorial (article) - DataCamp" (https://www.datacamp.com/community/tutorials/recommender-systems-python)
 - "Comprehensive Guide to build Recommendation Engine from scratch" (https://www.analyticsvidhya.com/...)
 - "Building a Recommendation Engine: An Algorithm Tutorial | Toptal" (https://www.toptal.com/...)
 - "What Is Recommendation System - Find the Top Results Here" (downloadsearch.cnet.com)
 - "Recommender Systems | Coursera" (https://www.coursera.org/specializations/recommender-systems)

Right: Wikipedia Search Results

- Search bar: "Recommender Sys"
- Results: "Recommender system" (containing... Recommender Sys)
- Article content:
 - Recommender system**: From Wikipedia, the free encyclopedia
 - A **recommender system** or a **recommendation system** (sometimes replacing "system" with a synonym such as platform or engine) is a subclass of **information filtering system** that seeks to predict the "rating" or "preference" a user would give to an item.^{[1][2]}
 - Recommender systems are utilized in a variety of areas including movies, music, news, books, research
 - Recommender systems** sidebar:
 - Concepts**: Collective intelligence · Relevance · Star ratings · Long tail
 - Methods and challenges**: Cold start · Collaborative filtering · Dimensionality reduction · Implicit data collection · Item-item collaborative filtering · Matrix factorization · Preference elicitation · Similarity search · Social loafing
 - Implementations**: Collaborative search engine · Content discovery platform · Decision support system · Music Genome Project · Product finder

Figure1.2. Recommender System vs searching

A study on 11 national news websites in the US conducted b/t 2007-10 shows that profile based personalized recommendations were only deployed on two websites during those years. This is because the building of a recommender system is a challenging process; since they require the real-time computation and filtering information relevant to the user which is very difficult to achieve with the big data.

1.4 Phases of recommendation process

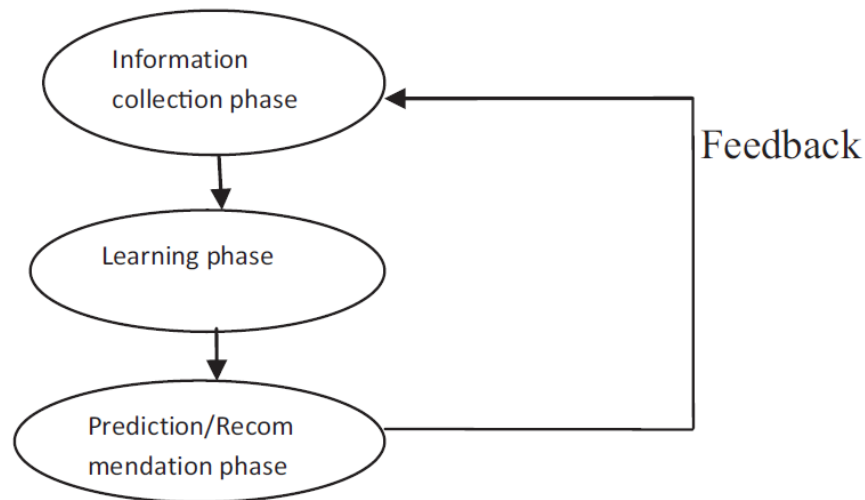


Figure 1.3 Recommendation Phases

1.4.1 Information collection phase

This collects relevant information or users to generate a user profile or model for the prediction tasks including user's attribute, behaviors or content of the resources the user accesses. A recommendation agent cannot function accurately until the user profile/model has been well constructed. The system needs to know as much as possible from the user in order to provide reasonable recommendation right from the onset. Recommender systems rely on different types of input such as the most convenient high quality explicit feedback, which includes explicit input by users regarding their interest in item or implicit feedback by inferring user preferences indirectly through observing user behavior.

Feedback type based on information acquisition:

a) Explicit feedback

Explicit feedback is acquired by the rating, voting or opinion type of information. Here system provides prompts the user through the system interface to provide rating for items in order to

construct and improve his personalized model. The accuracy of recommendation depends on the quantity of rating provided by the user. The only shortcoming of this method is, it requires effort from the users and also, users are not always ready to supply enough information. Despite the fact that explicit feedback requires more effort from user, it is still seen as providing more reliable data, since it does not involve extracting preferences from actions, and it also provides transparency into the recommendation process that results in a slightly higher perceived recommendation quality and more confidence in the recommendation.

b) Implicit feedback

Explicit feedback is acquired by the click, purchase or follow type of information. The system automatically infers the user's preferences by monitoring the different actions of users such as the history of purchases, navigation history, and time spent on some web pages, links followed by the user, content of e-mail and button clicks among others. Implicit feedback reduces the burden on users by inferring their user's preferences from their behavior with the system. The method though does not require effort from the user, but it is less accurate. Also, it has also been argued that implicit preference data might in actuality be more objective, as there is no bias arising from users responding in a socially desirable way and there are no self-image issues or any need for maintaining an image for others.

c) Hybrid feedback

Hybrid feedback can also be obtained through the combination of both explicit and implicit feedback. The strengths of both implicit and explicit feedback can be combined in a hybrid system in order to minimize their weaknesses and get a best performing system. This can be achieved by using an implicit data as a check on explicit rating or allowing user to give explicit feedback only when he chooses to express explicit interest.

Feedback type based on information scale:

a) Binary feedback

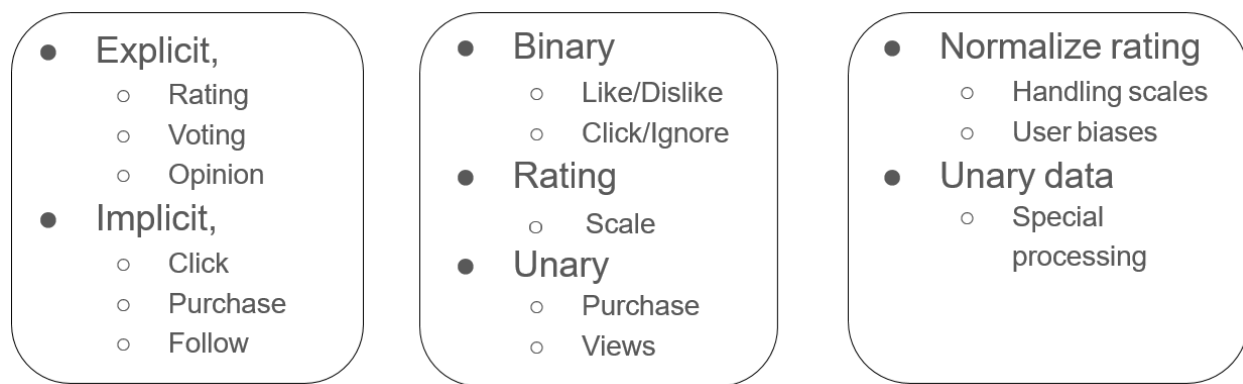
Binary feedback includes like/dislike or click/ignore types of options.

b) Rating feedback

Rating feedback includes different type of scale like 1-5 rating like amazon, and 1-10 rating scale like imdb.

c) Unary feedback

Unary feedback includes Purchase, Views, or like type of option like Facebook and YouTube.



1.4.2. Learning phase

It applies a learning algorithm to filter and exploit the user's features from the feedback gathered in information collection phase. In our 'headlines' news recommendation system we use cosine similarity method find the similar news and classifiers to find the news category.

1.4.3 Prediction/recommendation phase

It recommends or predicts what kind of items the user may prefer. This can be made either directly based on the dataset collected in information collection phase which could be memory based or model based or through the system's observed activities of the user.

1.5 Filtering techniques

There are following types of filtering techniques:

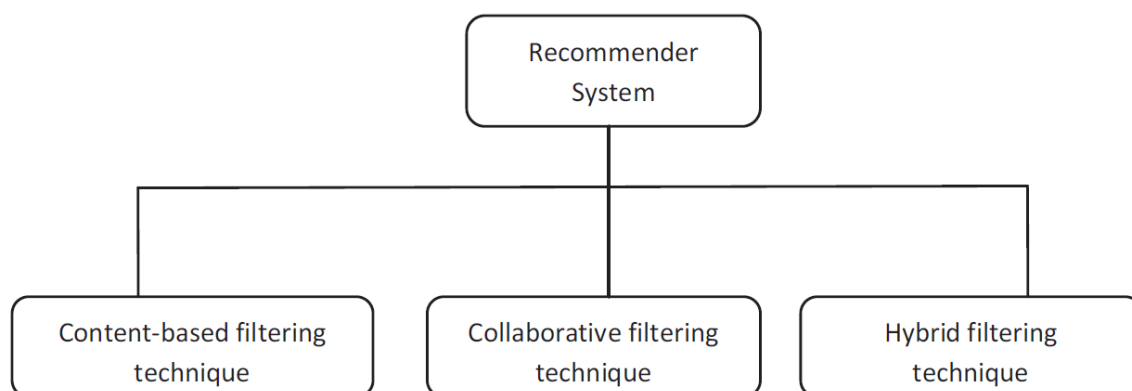


Figure: Types of Recommender System

1.5.1 Content-based filtering

Content-based filtering emphasizes more on the analysis of the attributes of items in order to generate predictions. When documents such as web pages, publications and news are to be recommended, content-based filtering technique is the most successful. In content-based filtering technique, recommendation is made based on the user profiles using features extracted from the content of the items the user has evaluated in the past. Items that are mostly related to the positively rated items are recommended to the user.

CBF uses different types of models to find similarity between documents in order to generate meaningful recommendations. It could use Vector Space Model such as Term Frequency Inverse Document Frequency (TF/IDF) or Probabilistic models such as Naive Bayes Classifier, Decision Trees or Neural Networks to model the relationship between different documents within a corpus.

1.5.1.1 TF-IDF:

TF-IDF is a vector space model that we use for modeling the relationship between different documents.

Term Frequency (Tf): gives us the frequency of the word in each document in the corpus. It is the ratio of number of times the word appears in a document compared to the total number of words in that document. It increases as the number of occurrences of that word within the document increases. Each document has its own tf.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}}$$

Inverse Data Frequency (idf): used to calculate the weight of rare words across all documents in the corpus. The words that occur rarely in the corpus have a high IDF score. It is given by the equation below.

$$idf(w) = \log\left(\frac{N}{df_t}\right)$$

Combining these two we come up with the TF-IDF score (w) for a word in a document in the corpus. It is the product of tf and idf:

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

tf_{ij} = number of occurrences of i in j
 df_i = number of documents containing i
 N = total number of documents

Example:

Let's take an example to get a clearer understanding.

Sentence 1: The car is driven on the road.

Sentence 2: The truck is driven on the highway.

Word	TF		IDF	TF*IDF	
	A	B		A	B
The	1/7	1/7	$\log(2/2) = 0$	0	0
Car	1/7	0	$\log(2/1) = 0.3$	0.043	0
Truck	0	1/7	$\log(2/1) = 0.3$	0	0.043
Is	1/7	1/7	$\log(2/2) = 0$	0	0
Driven	1/7	1/7	$\log(2/2) = 0$	0	0
On	1/7	1/7	$\log(2/2) = 0$	0	0
The	1/7	1/7	$\log(2/2) = 0$	0	0
Road	1/7	0	$\log(2/1) = 0.3$	0.043	0
Highway	0	1/7	$\log(2/1) = 0.3$	0	0.043

Table 1.1: a TFIDF data table

From the above table we can see that TF-IDF of common words was zero, which shows that they are not significant. On the other hand, the TF-IDF of 'car', 'truck', 'road', and 'highway' are non-zero. They words have more significances.5/8/2019

1.5.1.2 Similarity Measures

We find the similarity between different vector space models using the following similarity measures:

1. Cosine-similarity:

$$CosSim(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}} = \frac{\langle x, y \rangle}{||x|| ||y||}$$

2. Pearson-similarity:

$$Corr(x, y) = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2} \sqrt{\sum (y_i - \bar{y})^2}}$$

1.5.1.3 Pros and Cons of content-based filtering techniques

CB filtering techniques overcome the challenges of CF. They have the ability to recommend new items even if there are no ratings provided by users. So even if the database does not contain user preferences, recommendation accuracy is not affected. Also, if the user preferences change, it has the capacity to adjust its recommendations in a short span of time. They can manage situations where different users do not share the same items, but only identical items according to their intrinsic features. Users can get recommendations without sharing their profile, and this ensures privacy [3]. CBF technique can also provide explanations on how recommendations are generated to users. However, the techniques suffer from various problems as discussed in the literature [1]. Content based filtering techniques are dependent on items' metadata. That is, they require rich description of items and very well organized user profile before recommendation can be made to users. This is called limited content analysis. So, the effectiveness of CBF depends on the availability of descriptive data. Content overspecialization [4] is another serious problem of CBF technique. Users are restricted to getting recommendations similar to items already defined in their profiles.

Content based techniques does not need the profile of other users since they do not need the profile of other users since they do not influence recommendation.

1.5.2. Collaborative filtering

Collaborative filtering is a domain-independent prediction technique for content that cannot easily and adequately be described by metadata such as movies and music. Collaborative filtering technique works by building a database (user-item matrix) of preferences for items by users. It then matches users with relevant interest and preferences by calculating similarities between their profiles to make recommendations [4]. Such users build a group called neighborhood. A user gets recommendations to those items that he has not rated before but that were already positively rated by users in his neighborhood.

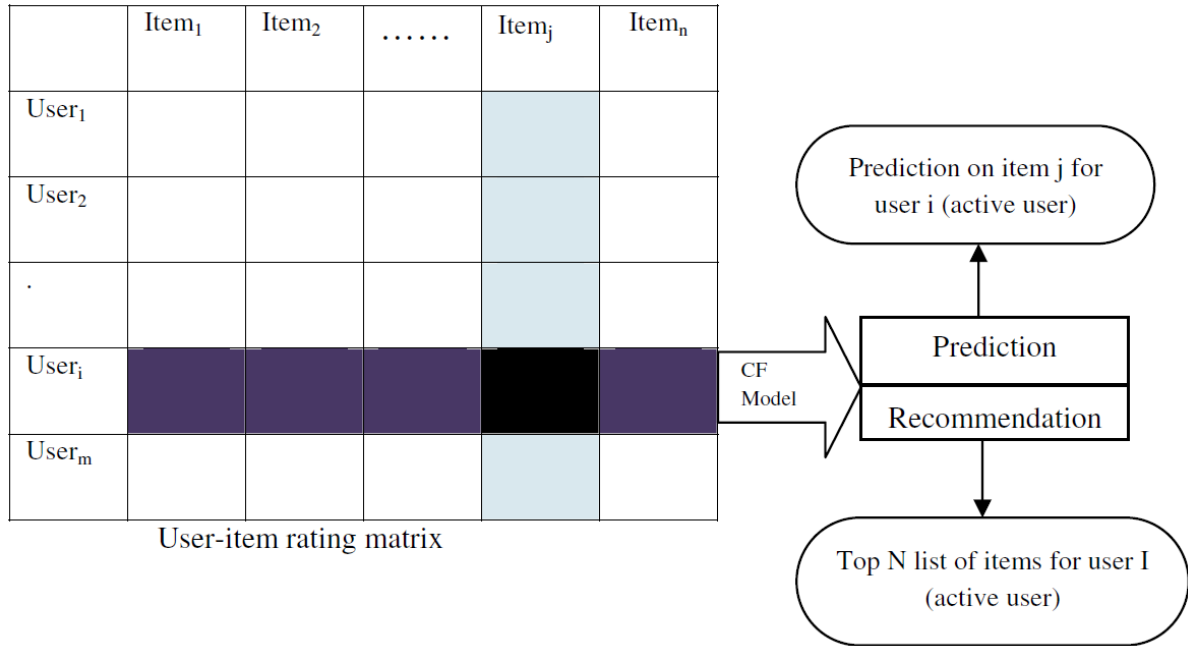


Figure1.5: Collaborative filtering process

In collaborative filtering we maintain the two types of information for recommendation.

1.5.2.1. User-user similarity

1. Build neighborhood for each user based on the User x Item matrix (by using the correlation or cosine similarity of each user with others)

$$Sim(a, u) = \frac{\bar{a} \cdot \bar{u}}{||\bar{a}|| ||\bar{u}||} = \frac{\sum_i \hat{r}_{ai} \hat{r}_{ui}}{\sqrt{\sum_i (\hat{r}_{ui})^2} \sqrt{\sum_i (\hat{r}_{ai})^2}}$$

2. Use the likes/interactions/ratings of the top k users to build a potential set of items to recommend for the user a.

3. Score each item I in the potential set based on the preference of the user u and their similarity with user a.

Another popular similarity measure in user-based collaborative filtering is **Pearson Correlation**:

$$s(a, u) = \frac{\sum_{i=1}^n (r_{a,i} - \bar{r}_a)(r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i=1}^n (r_{a,i} - \bar{r}_a)^2} \sqrt{\sum_{i=1}^n (r_{u,i} - \bar{r}_u)^2}}$$

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

sim = 0,85
sim = 0,70
sim = -0,79

Table 1.2 user-user similarity in collaborative filtering

1.5.2.2. Item-item similarity

1. Build an item neighborhood based on the preferences expressed by different users (i.e. based on the rating vectors of two different items).

$$s(\vec{u}, \vec{v}) = \frac{\vec{u} \cdot \vec{v}}{|\vec{u}| * |\vec{v}|} = \frac{\sum_i r_{u,i} r_{v,i}}{\sqrt{\sum_i r_{u,i}^2} \times \sqrt{\sum_i r_{v,i}^2}}$$

2. Use set of items that the user has expressed preferences on up front to generate potential items as recommendations for them
3. Score each item in the potential set based on their similarity with the item that the user has liked (expressed preference) earlier

$$S(u, i) = \frac{\sum_{j \in N} Sim(i, j) r_{uj}}{\sum_{j \in N} |Sim(i, j)|}$$

Basic idea is to use the similarity between items (and not users) to make predictions

Example:

- Look for items that are similar to Item5
- Take Alice's ratings for these items to predict the rating for Item5

	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

Table 1.3 Item-Item similarity in collaborative filtering

1.5.2.3 Pros and Cons of collaborative filtering techniques

Collaborative Filtering has some major advantages over CBF in that it can perform in domains where there is not much content associated with items and where content is difficult for a computer system to analyze (such as opinions and ideal). Also, CF technique has the ability to provide serendipitous recommendations, which means that it can recommend items that are relevant to the user even without the content being in the user's profile [6].

1. Cold-start problem. This refers to a situation where a recommender does not have adequate information about a user or an item in order to make relevant predictions [6]. This is one of the major problems that reduce the performance of recommendation system. The profile of such new user or item will be empty since he has not rated any item; hence, his taste is not known to the system.

2. Data sparsity problem. This is the problem that occurs as a result of lack of enough information, that is, when only a few of the total number of items available in a database are rated by users [7]. This always leads to a sparse user item matrix, inability to locate successful neighbors

and finally, the generation of weak recommendations. Also, data sparsity always leads to coverage problems, which is the percentage of items in the system that recommendations can be made for [6]

3. Scalability: This is another problem associated with recommendation algorithms because computation normally grows linearly with the number of users and items [7]. A recommendation technique that is efficient when the number of dataset is limited may be unable to generate satisfactory number of recommendations when the volume of dataset is increased. Thus, it is crucial to apply recommendation techniques which are capable of scaling up in a successful manner as the number of dataset in a database increases. Methods used for solving scalability problem and speeding up recommendation generation are based on Dimensionality reduction techniques, such as Singular Value Decomposition (SVD) method, which has the ability to produce reliable and efficient recommendations.

4 Long tail problem: below graph shows that the items which are less popular are not recommended in our algorithm this stops the sailing of new products.



Figure 1.4 Long tail problem

1.5.3 Hybrid filtering

It combines the both the approach (Content based and Collaborative filtering approach) and hence leverage the both of the approaches benefits. By using the both it avoid some limitations and problems of pure recommendation systems as the disadvantages of one algorithm can be overcome by another algorithm.

As the volume of data that these entity recommendation systems process is very large, the goal of an entity recommendation system over big data is to design a system that is scalable, efficient, and provides the best possible results for a large variety of queries.

1.6 Websites using Recommendation System:

1. It is estimated that close to 30% of Amazon's revenue comes from the way they integrated recommendations.
2. About 75% of Netflix Business is driven through recommendations.
3. Posts, Groups, People and jobs are recommended through LinkedIn which increases the network.
4. Posts, friends suggested by Facebook to increase their revenue through advertisements.
5. Stories recommended through Quora

1.7 Advantage of Recommendation System:

1. Increase customer engagement.
2. Increase customer satisfaction by delivering relevant contents.
3. Increase sales with cross sell options etc.

1.8 Problem with Recommendation System:

But most of the Recommender Systems are Data – Driven and we need a real-time computation at large scale for recommending a product.

We need a large scale, real-time computing engine. That where *Apache Spark* come to the Rescue.

CHAPTER-2

LITERATURE SURVEY

News recommendation system is the process in which we filter the information according to the interest of user and recommend him. While recommending we require the classifier which can classify the news articles read by the user and find the category of that article, then we fetch the article of that category and recommend to him based on the similarity score.

2.1. Text Classification Algorithms

We used the following classification method in our project:

1. Naïve bayes classifier
2. Support vector machine
3. K-nearest neighbor
4. Logistic regression
5. Decision tree
6. Random forest classifier

2.1.1. Naïve Bayes Classifier:

Input:

- A document d
- fixed set of classes $C = \{c_1, c_2, \dots, c_J\}$
- A training set of m hand-labeled document $(d_1, c_1), \dots, (d_m, c_m)$

Output:

- A learned classifier: $\gamma: d \rightarrow c$

How to represent the document text, so that we can perform numeric computation?

We use Bag of words or vector space model or tf-idf for that.

Bayes' Rule Applied to **Documents (d)** and **Classes (c)**

$$P(c | d) = \frac{P(d | c)P(c)}{P(d)}$$

Naïve Bayes Classifier:

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(c | d)$$

MAP is "maximum a posteriori" = most likely class

$$= \operatorname{argmax}_{c \in C} \frac{P(d | c)P(c)}{P(d)}$$

Bayes Rule

$$= \operatorname{argmax}_{c \in C} P(d | c)P(c)$$

Dropping the denominator

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(d | c)P(c)$$

$$= \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n | c)P(c)$$

Document d represented as features $x_1 \dots x_n$

It is the very popular supervised classification algorithm. This algorithm is called "Naive" because it makes a naive assumption that each feature is independent of other features which is not true in real life. As for the "Bayes" part, it refers to the statistician and philosopher, Thomas Bayes and the theorem named after him, Bayes' theorem, which is the base for Naive Bayes Algorithm.

Multinomial Naïve Bayes Independence Assumptions:

- Bag of Word assumption: Assume position doesn't matter
- Conditional independence: feature probabilities $P(x_i | c_j)$ are independent, given the class C.

$$P(x_1, \dots, x_n | c) = P(x_1 | c) \cdot P(x_2 | c) \cdot P(x_3 | c) \cdot \dots \cdot P(x_n | c)$$

Using the above property:

$$c_{MAP} = \operatorname{argmax}_{c \in C} P(x_1, x_2, \dots, x_n | c) P(c)$$

$$c_{NB} = \operatorname{argmax}_{c \in C} P(c_j) \prod_{x \in X} P(x | c)$$

Text Classification and Naïve Bayes:

1. Learning:

1. First option is to simply use the frequencies in the data

$$\hat{P}(w_i | c_j) = \frac{\text{count}(w_i, c_j)}{\sum_{w \in V} \text{count}(w, c_j)}$$

This is the fraction of the time W_i , appears among all words in documents of topic C_j .

Problem with Maximum Likelihood

What if we have seen no training documents with the word *fantastic* and classified in the topic positive?

$$\hat{P}(\text{"fantastic"} | \text{positive}) = \frac{\text{count}(\text{"fantastic"}, \text{positive})}{\sum_{w \in V} \text{count}(w, \text{positive})} = 0$$

Zero probabilities cannot be conditioned away, no matter the other evidence!

Option 2: (Laplace (add-1) smoothing for Naïve bayes)

$$\begin{aligned} \hat{P}(w_i | c) &= \frac{\text{count}(w_i, c) + 1}{\sum_{w \in V} (\text{count}(w, c) + 1)} \\ &= \frac{\text{count}(w_i, c) + 1}{\left(\sum_{w \in V} \text{count}(w, c) \right) + |V|} \end{aligned}$$

We choose this option.

$$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

$$P(w_k | c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |Vocabulary|}$$

Example:

Assume we have the following training and test set:

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

Table 2.1: Training data for Naïve bayes classifier example

Step1. Calculate the priors as follow:

$$\hat{P}(c) = \frac{N_c}{N}$$

$$P(c) = \frac{3}{4} \quad P(j) = \frac{1}{4}$$

Step2. Calculate the Conditional Probabilities:

$$\hat{P}(w | c) = \frac{\text{count}(w, c) + 1}{\text{count}(c) + |V|}$$

$$\begin{aligned}
P(\text{Chinese} | c) &= (5+1) / (8+6) = 6/14 = 3/7 \\
P(\text{Tokyo} | c) &= (0+1) / (8+6) = 1/14 \\
P(\text{Japan} | c) &= (0+1) / (8+6) = 1/14 \\
P(\text{Chinese} | j) &= (1+1) / (3+6) = 2/9 \\
P(\text{Tokyo} | j) &= (1+1) / (3+6) = 2/9 \\
P(\text{Japan} | j) &= (1+1) / (3+6) = 2/9
\end{aligned}$$

Step3. Choosing the class:

$$\begin{aligned}
P(c | d_5) &\propto 3/4 * (3/7)^3 * 1/14 * 1/14 \\
&\approx 0.0003
\end{aligned}$$

$$\begin{aligned}
P(j | d_5) &\propto 1/4 * (2/9)^3 * 2/9 * 2/9 \\
&\approx 0.0001
\end{aligned}$$

Above naïve bayes classifier predict the class of document d_5 as the C.

Advantages of Naïve bayes:

- Very fast, low storage requirements.
- Since irrelevant features cancels each other therefore robust to irrelevant features.
- Very good in domains with many equally important features.

2.1.2 Support Vector Machine (SVM) Classifier:

SVM is an algorithm that determines the best decision boundary between vectors that belong to a given group (category) and vectors that do not belong to it. This means that in order to leverage the power of SVM text classification, texts have to be transformed into vectors. In our ‘headlines’ web site we used **Tfidfvectorizer** to transform the text into numeric vectors.

Vectors are (sometimes huge) lists of numbers which represent a set of coordinates in some space. In SVM, SVM decides where to draw the best “line” (or the best hyperplane) that divides the space into two subspaces: one for the vectors which belong to the given category and one for the vectors which do not belong to it.

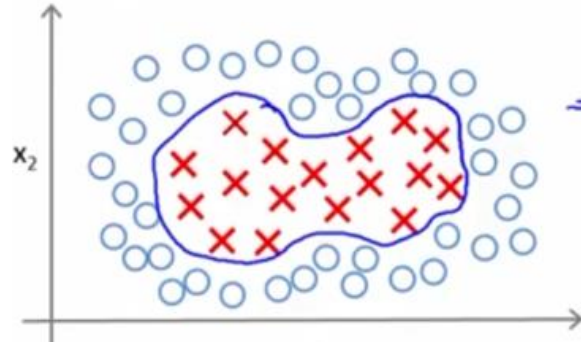


Figure 2.1: non-linear classification using svm

2.1.3. KNN classifier:

KNN is a non-parametric and lazy learning algorithm. Non-parametric means there is no assumption for underlying data distribution. I.e. the model structure is determined from the dataset. Lazy algorithm means it does not need any training data points for model generation. All training data used in the testing phase. This makes training faster and testing phase slower and costlier.

How does the KNN works?

Pseudo Code of KNN

We can implement a KNN model by following the below steps:

1. Load the data
2. Initialize the value of k
3. For getting the predicted class, iterate from 1 to total number of training data points
 1. Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it's the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.
 2. Sort the calculated distances in ascending order based on distance values
 3. Get top k rows from the sorted array
 4. Get the most frequent class of these rows
 5. Return the predicted class

The classification process determines the vectors distance between the documents by using the following equation:

$$d(x, y) = \sqrt{\sum_{r=1}^N (a_{rx} - a_{ry})^2}$$

Where N is the number of unique words in the document collection and a_{rx} is a weight of the term r in document x , a_{ry} is a weight of the term r in document y .

Example:

In KNN, K is the number of nearest neighbors. The number of neighbors is the core deciding factor. K is generally an odd number if the number of classes is 2. When $K=1$, then the algorithm is known as the nearest neighbor algorithm. This is the simplest case. Suppose $P1$ is the point, for which label needs to predict. First, you find the one closest point to $P1$ and then the label of the nearest point assigned to $P1$.

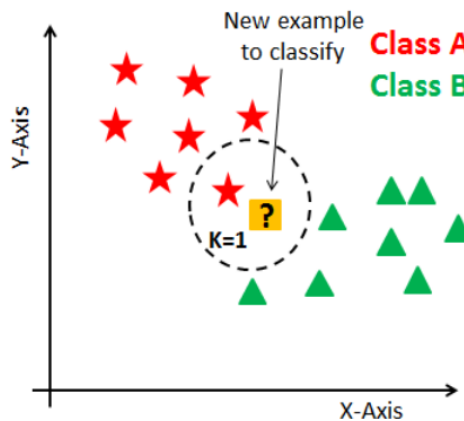


Figure 1.2 KNN classification $K=1$

Suppose $P1$ is the point, for which label needs to predict. First, you find the k closest point to $P1$ and then classify points by majority vote of its k neighbors. Each object votes for their class and the class with the most votes is taken as the prediction. For finding closest similar points, you find the distance between points using distance measures such as Euclidean distance, Hamming distance, Manhattan distance and Minkowski distance. KNN has the following basic steps:

Generally, Data scientists choose as an odd number if the number of classes is even. You can also check by generating the model on different values of k and check their performance. You can also try Elbow method here.

1. Calculate distance

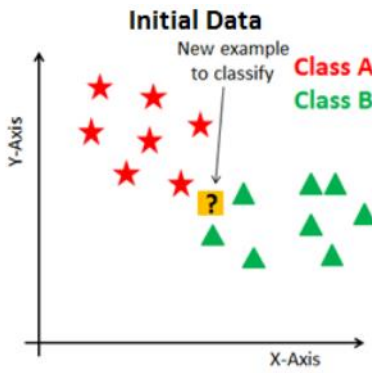


Figure1.3: select and initial data point

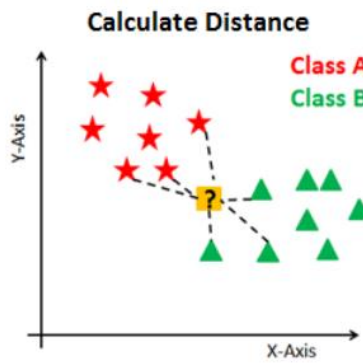


Figure1.4: calculate the distance from each data point to selected point

2. Find closest neighbors
3. Vote for labels/class

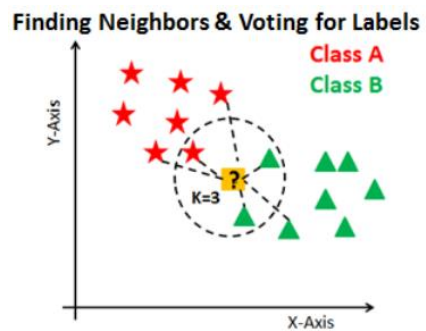


Figure 1.5 finding neighbors and vote for class/labels

Eager vs. Lazy Learners

Eager learners mean when given training points will construct a generalized model before performing prediction on given new points to classify. You can think of such learners as being ready, active and eager to classify unobserved data points.

2.1.4. Logistic Regression classifier:

Logistic regression is used when the dependent variable (target) is categorical. For example:

- To predict whether an email is spam (1) or (0)
- Whether the tumor is malignant (1) or not (0)

Sigmoid Function

Model

Output = 0 or 1

Hypothesis $\Rightarrow Z = WX + B$

$H(x) = \text{sigmoid}(Z)$

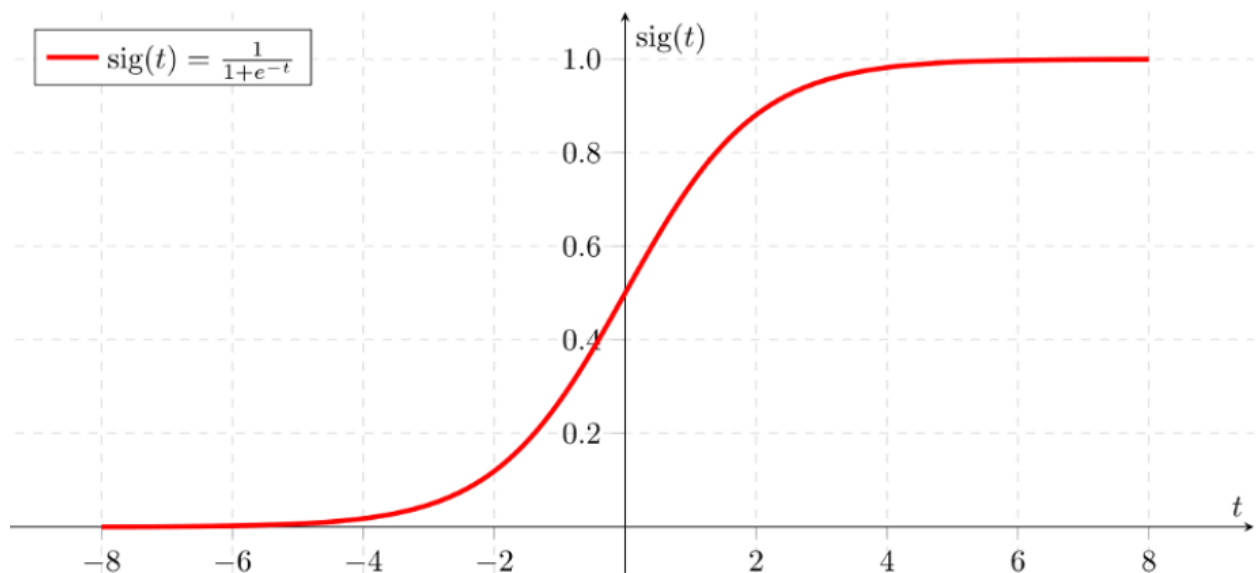


Figure 1.6 graph showing sigmoid function

If 'Z' goes to infinity, Y(predicted) will become 1 and if 'Z' goes to negative infinity, Y(predicted) will become 0.

Types of Logistic Regression

1. Binary Logistic Regression

The categorical response has only two possible outcomes. Example: Spam or Not

2. Multinomial Logistic Regression

Three or more categories without ordering. Example: Predicting which food is preferred more (Veg, Non-Veg, Vegan)

2.1.5. Decision Tree Algorithm

A decision tree is a flowchart-like tree structure where an internal node represents feature (or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in recursively manner call recursive partitioning. This flowchart-like structure helps you in decision making. It's visualization like a flowchart diagram which easily mimics the human level thinking. That is why decision trees are easy to understand and interpret.

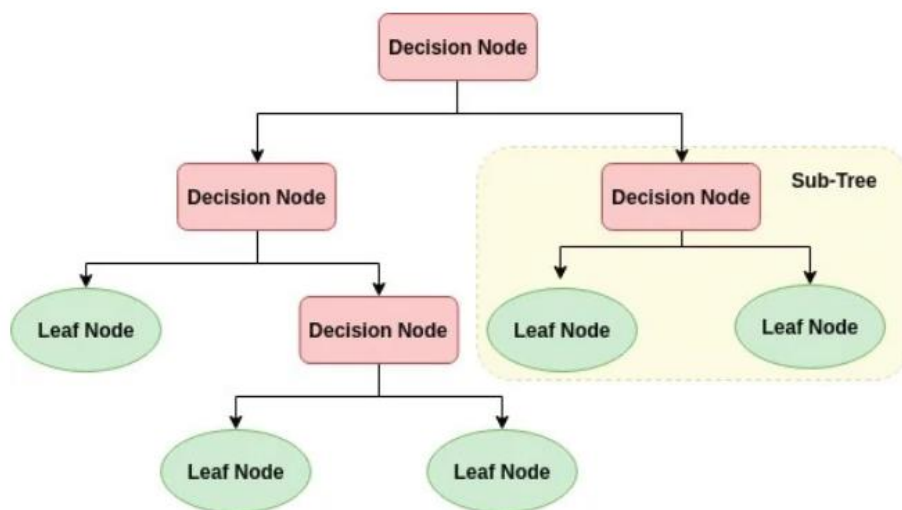


Figure1.7: Decision Tree Diagram

Decision Tree is a white box type of ML algorithm. It shares internal decision-making logic, which is not available in the black box type of algorithms such as Neural Network. Its training time is faster compared to the neural network algorithm. The time complexity of decision trees is a function of the number of records and number of attributes in the given data. Decision trees can handle high dimensional data with good accuracy.

How does the Decision Tree algorithm work?

The basic idea behind any decision tree algorithm is as follows:

1. Select the best attribute using Attribute Selection Measures (ASM) to split the records.
2. Make that attribute a decision node and breaks the dataset into smaller subsets.
3. Starts tree building by repeating this process recursively for each child until one of the condition will match.

All the tuples belong to the same attribute value.

There are no more remaining attributes.

There are no more instances.

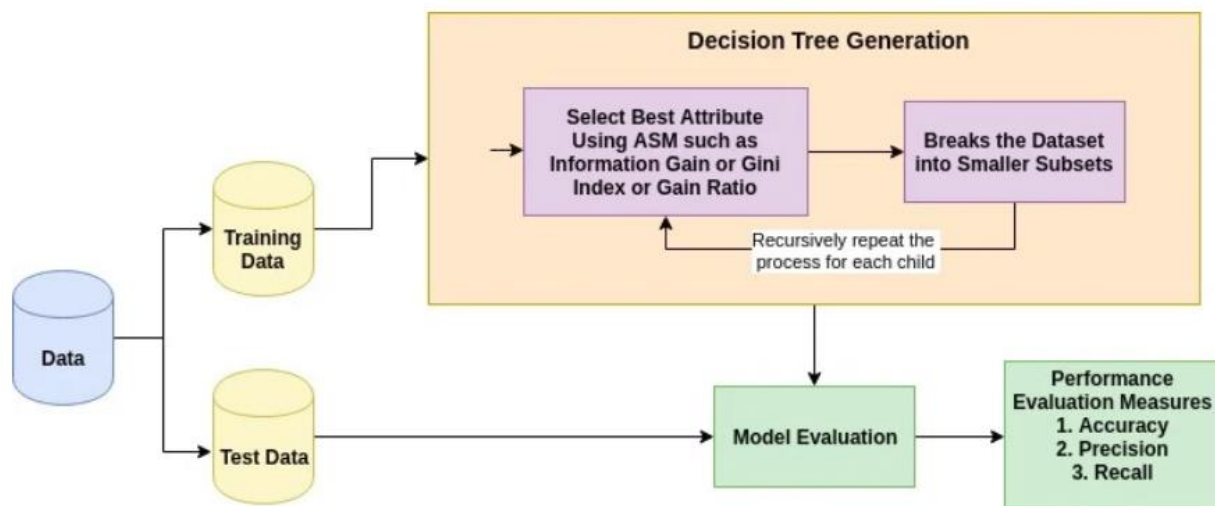


Figure 1.8 process of decision tree

2.1.6. Random Forest Algorithm

Let's understand the algorithm in layman's terms. Suppose you want to go on a trip and you would like to travel to a place which you will enjoy. So what do you do to find a place that you will like? You can search online, read reviews on travel blogs and portals, or you can also ask your friends.

Let's suppose you have decided to ask your friends, and talked with them about their past travel experience to various places. You will get some recommendations from every friend. Now you have to make a list of those recommended places. Then, you ask them to vote (or select one best place for the trip) from the list of recommended places you made. The place with the highest number of votes will be your final choice for the trip.

In the above decision process, there are two parts. First, asking your friends about their individual travel experience and getting one recommendation out of multiple places they have visited. This part is like using the **decision tree algorithm**. Here, each friend makes a selection of the places he or she has visited so far.

The second part, after collecting all the recommendations, is the voting procedure for selecting the best place in the list of recommendations. This whole process of getting recommendations from friends and voting on them to find the best place is known as the **random forests algorithm**.

This collection of decision tree is also known as forest. The individual decision trees are generated using an attribute selection indicator such as information gain, gain ratio, and Gini index for each attribute. Each tree depends on an independent random sample. In a classification problem, each tree votes and the most popular class is chosen as the final result. In the case of regression, the average of all the tree outputs is considered as the final result. It is simpler and more powerful compared to the other non-linear classification algorithms.

How does the algorithm work?

1. Select random samples from a given dataset.
2. Construct a decision tree for each sample and get a prediction result from each decision tree.

3. Perform a vote for each predicted result.
4. Select the prediction result with the most votes as the final prediction.

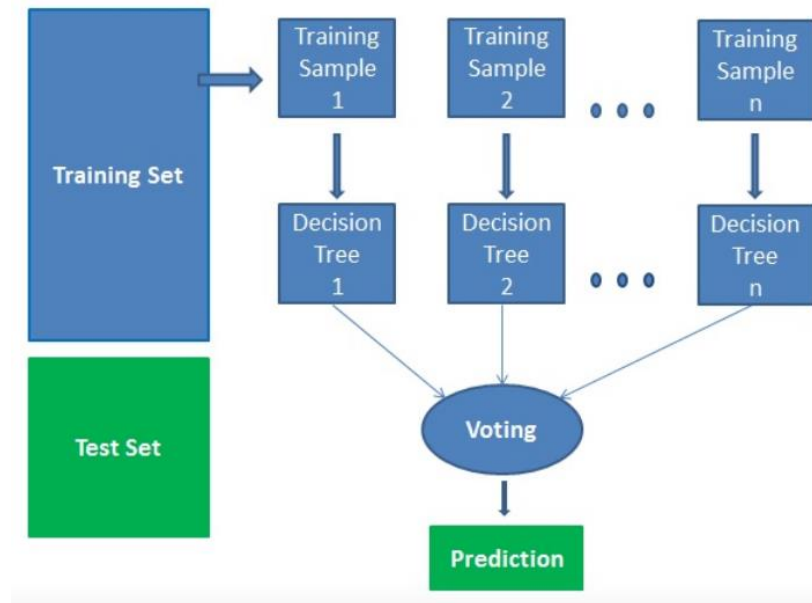


Figure 1.9 Random Forest Algorithm

Advantages:

- Random forests is considered as a highly accurate and robust method because of the number of decision trees participating in the process.
- It does not suffer from the overfitting problem. The main reason is that it takes the average of all the predictions, which cancels out the biases.
- The algorithm can be used in both classification and regression problems.

Disadvantages:

- Random forests is slow in generating predictions because it has multiple decision trees. Whenever it makes a prediction, all the trees in the forest have to make a prediction for the same given input and then perform voting on it. This whole process is time-consuming.

- The model is difficult to interpret compared to a decision tree, where you can easily make a decision by following the path in the tree.

2.2 Text Similarities: Estimate the degree of similarity between two texts

Text similarity has to determine how ‘close’ two pieces of text are both in surface closeness [**lexical similarity**] and meaning [**semantic similarity**].

For instance, how similar are the phrases “*the cat ate the mouse*” with “*the mouse ate the cat food*” by just looking at the words?

- **On the surface**, if you consider only **word level similarity**, these two phrases appear very similar as 3 of the 4 unique words are an exact overlap. It typically does not take into account the actual meaning behind words or the entire phrase in context
- Instead of doing a **word for word comparison**, we also need to pay attention to **context** in order to capture more of the **semantics**. To consider **semantic similarity** we need to focus on **phrase/paragraph levels** (or lexical chain level) where a piece of text is broken into a relevant group of related words prior to computing similarity. We know that while the words significantly overlap, these two phrases actually have **different meaning**.

2.2.1 Similarity Measures

We find the similarity between different vector space models using the following similarity measures:

1. Cosine-similarity:

$$CosSim(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}} = \frac{\langle x, y \rangle}{||x|| ||y||}$$

2. Pearson-similarity:

$$Corr(x, y) = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2} \sqrt{\sum_i (y_i - \bar{y})^2}}$$

CHAPTER-3

MODELLING AND SYSTEM IMPLEMENTATION

3.1 Back-End

Machine Learning:

We used Scikit-Learn (python Machine Learning library) for implementing the above algorithms. Scikit-Learn provides a range of supervised and unsupervised learning algorithm via a consistent interface in python.

The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

NumPy: Base n-dimensional array package

SciPy: Fundamental library for scientific computing

Matplotlib: Comprehensive 2D/3D plotting

Ipython: Enhanced interactive console

Sympy: Symbolic mathematics

Pandas: Data structures and analysis

3.1.1. Data Collection

```
t_path = "../dataset/bbc/"

all_docs = defaultdict(lambda: list())

topic_list = list()
text_list = list()
```

```

print("Reading all the documents...\n")

for topic in os.listdir(t_path):
    d_path = t_path + topic + '/'

    for f in os.listdir(d_path):
        f_path = d_path + f
        file = open(f_path, 'r')
        text_list.append(file.read())
        file.close()
        topic_list.append(topic)

```

3.1.2. Splitting the data into training data and test data.

```

from sklearn.model_selection import train_test_split
title_train, title_test, category_train, category_test = train_test_split(text_list, topic_list)
title_train, title_dev, category_train, category_dev = train_test_split(title_train, category_train)

```

3.1.3. Tokenizing words, removing stopwords and converting the raw data into numeric data vector space model using tfidfvectorizer and Tokenizer as follow:

```

tokenizer = nltk.tokenize.RegexpTokenizer(r"\w+")
stop_words = nltk.corpus.stopwords.words("english")
vectorizer = TfidfVectorizer(tokenizer=tokenizer.tokenize,
                             stop_words=stop_words)

vectorizer.fit(iter(title_train))
Xtr = vectorizer.transform(iter(title_train))
Xde = vectorizer.transform(iter(title_dev))
Xte = vectorizer.transform(iter(title_test))

```

3.1.4. Fitting the category variable as follow:

```

from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
encoder.fit(category_train)
Ytr = encoder.transform(category_train)

Yde = encoder.transform(category_dev)
Yte = encoder.transform(category_test)

```

3.1.5. Classifier Model creation:

```
from sklearn.naive_bayes import MultinomialNB
classifier = MultinomialNB()
```

3.1.6. Fitting the classifier on training data:

```
print('\nTraining...', classifier_names[i])
classifier.fit(Xtr, Ytr)
```

3.1.7. Making the prediction and generating the classification report:

```
print('\nerror(confusion) matrix of', classifier_names[i])
pred = classifier.predict(Xde)
prediction_list.append(pred[0]);
print('predicted index for category ', pred[0])
print(classification_report(Yde, pred,
target_names=encoder.classes_))
```

Output:

```
error(confusion) matrix of naive bayes
predicted index for category 1
              precision    recall  f1-score   support

   business      0.91      0.98      0.95        88
entertainment    1.00      0.89      0.94        73
   politics      0.86      0.97      0.91        69
      sport      0.98      1.00      0.99        98
       tech      1.00      0.90      0.95        89

 micro avg      0.95      0.95      0.95       417
 macro avg      0.95      0.95      0.95       417
weighted avg      0.95      0.95      0.95       417
```

3.1.8. Find the prediction score:

```
score = classifier.score(Xte, Yte)
print(classifier_names[i], 'score is =>', score)
```

Output:

```
naive bayes score is => 0.9371633752244165
```

Similarly from the above process from **step5** – **step8** we repeat the process for each type of classifier and find the result as follow:

Classifier name : support vector machine

Training... support vector machine

error(confusion) matrix of support vector machine

predicted index for category 1

	precision	recall	f1-score	support
business	0.97	0.97	0.97	88
entertainment	0.97	1.00	0.99	73
politics	0.97	0.97	0.97	69
sport	0.99	1.00	0.99	98
tech	1.00	0.97	0.98	89
micro avg	0.98	0.98	0.98	417
macro avg	0.98	0.98	0.98	417
weighted avg	0.98	0.98	0.98	417

support vector machine score is => 0.9748653500897666

Classifier name : k-nearest neighbour

Training... k-nearest neighbour

error(confusion) matrix of k-nearest neighbour

predicted index for category 1

	precision	recall	f1-score	support
business	0.93	0.91	0.92	88
entertainment	0.96	0.88	0.91	73
politics	0.88	0.94	0.91	69
sport	0.96	0.99	0.97	98
tech	0.98	0.98	0.98	89
micro avg	0.94	0.94	0.94	417
macro avg	0.94	0.94	0.94	417
weighted avg	0.94	0.94	0.94	417

k-nearest neighbour score is => 0.9371633752244165

Classifier name : logistic regression

Training... logistic regression

error(confusion) matrix of logistic regression

predicted index for category 1

	precision	recall	f1-score	support
business	0.93	0.97	0.95	88
entertainment	0.99	0.96	0.97	73
politics	0.93	0.97	0.95	69
sport	0.99	1.00	0.99	98
tech	0.99	0.93	0.96	89
micro avg	0.97	0.97	0.97	417
macro avg	0.97	0.97	0.97	417
weighted avg	0.97	0.97	0.97	417

logistic regression score is => 0.9551166965888689

Classifier name : decision tree

Training... decision tree

error(confusion) matrix of decision tree

predicted index for category 1

	precision	recall	f1-score	support
business	0.75	0.78	0.77	88
entertainment	0.83	0.82	0.83	73
politics	0.76	0.83	0.79	69
sport	0.91	0.90	0.90	98
tech	0.85	0.78	0.81	89
micro avg	0.82	0.82	0.82	417
macro avg	0.82	0.82	0.82	417
weighted avg	0.82	0.82	0.82	417

decision tree score is => 0.7809694793536804

Classifier name : randomforestclassifier

Training... randomforestclassifier

error(confusion) matrix of randomforestclassifier

predicted index for category 1

	precision	recall	f1-score	support
business	0.77	0.93	0.85	88
entertainment	0.85	0.88	0.86	73
politics	0.85	0.83	0.84	69
sport	0.94	0.96	0.95	98
tech	0.97	0.75	0.85	89
micro avg	0.87	0.87	0.87	417
macro avg	0.88	0.87	0.87	417
weighted avg	0.88	0.87	0.87	417

randomforestclassifier score is => 0.8707360861759426

3.1.9 Most Efficient classifier:

From the above results we conclude that SVM produces the highest accuracy with as follow:

```
max score classifier name : support vector machine  and score: 0.9748653500897666
```

3.1.10 Comparison Chart of Classifiers:

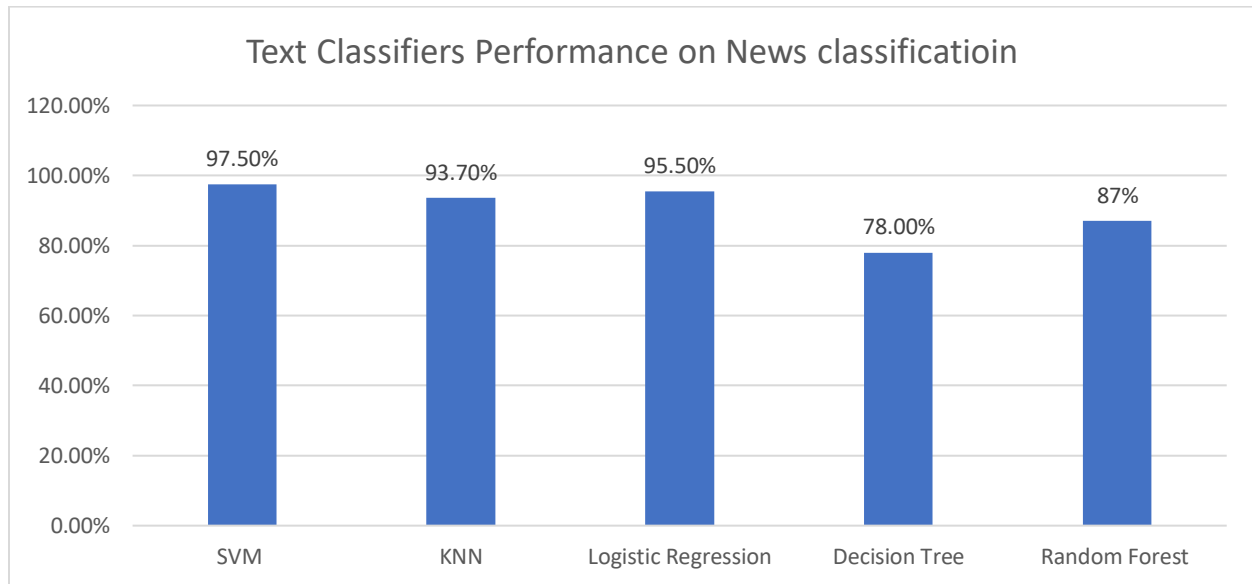


Figure 3.1: performance of various classifiers on our news dataset

3.1.11 Flask:

We used the MVC architecture using python Flask for connecting the front end to back end and hence implementing the business logic.

Flask is a web framework, which provides you with tools, libraries and technologies that allow you to build a web application. Flask is a micro framework for Python based on Werkzeug, Jinja 2.

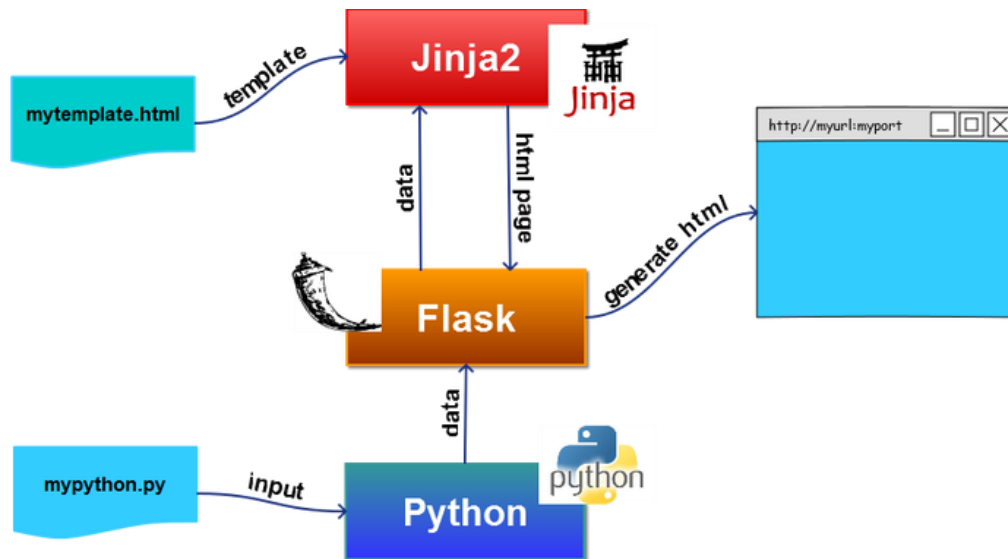


Figure 3.2: Flask work flow using MVC architecture

3.2 Front-End:

3.2.1 Bootstrap

Bootstrap is a free and open-source CSS framework directed at responsive, mobile first front-end framework for web development. It contains CSS and JavaScript – based design templates for typography, forms, buttons, navigations and other interface components.

3.2.2 JavaScript

JavaScript is the Programming language for the web which can update and change both HTML and CSS. JavaScript can calculate, manipulate and validate data.

3.2.3 HTML5

HTML is a standard markup language used for creating web pages.

3.2.4 CSS

CSS (cascade style sheet) is used for styling the web pages.

CHAPTER 4

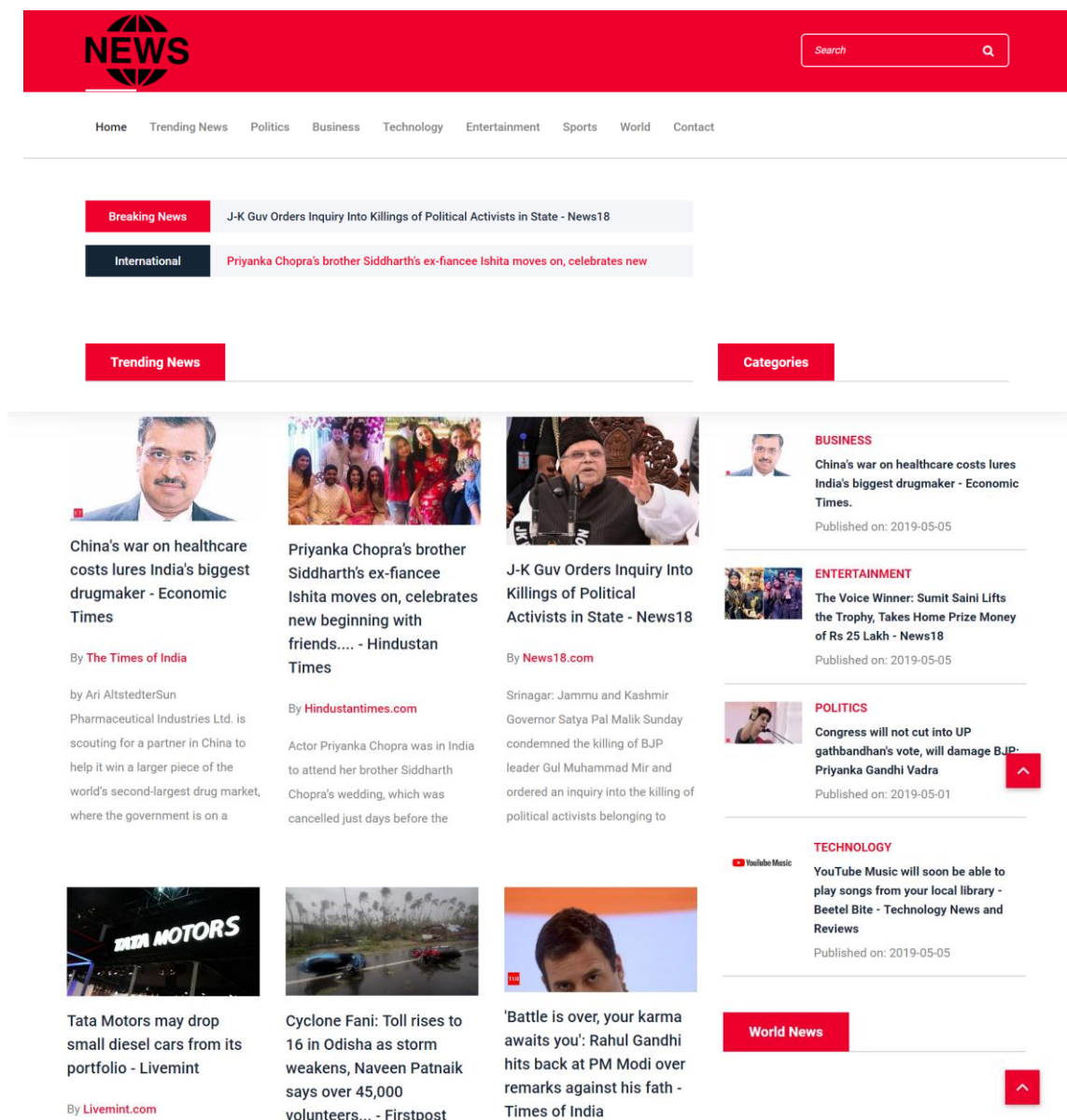
OUTPUT AND RESULTS

RESULT

```
(venvp) D:\flask\newsp>flask run
```

```
-----
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

4.1 News Headlines Website Interface:



New Delhi: Auto major Tata Motors may phase out small diesel cars from its portfolio as demand is expected to slow down due to upcoming BS-VI emission norms that would make such vehicles expensive, according to a senior company official. Already, market lead... [+2445 chars]

Published on: 2019-05-05

By **Firstpost.com**


Cyclone Fani, the worst storm to hit the Indian subcontinent in 20 years and which left at least 16 people dead in India, weakened into a 'deep depression' and lay centered over Bangladesh on Saturday morning, after it moved further north-east from West Bengea... [+7120 chars]

By **The Times of India**

Modi Ji, The battle is over. Your Karma awaits you. Projecting your inner beliefs about yourself onto my father wo...

<https://t.co/Q8cxPQtToy>
— Rahul Gandhi (@RahulGandhi) 1557035131000

Published on: 2019-05-05



New York Rain: Vintage Sights and Sounds of a Soaked City

Published on: 2019-05-05

Contact:

rahul77349@gmail.com
+8005510834


Copyright ©2019 All rights reserved

<mailto:rahul77349@gmail.com>

4.2 Recommendation of News:

Suppose we click / read the following news (see the picture of 'Virat kohli') highlighted in red text since we hover on it as follow:

[Home](#)
[Trending News](#)
[Politics](#)
[Business](#)
[Technology](#)
[Entertainment](#)
[Sports](#)
[World](#)
[Contact](#)




Arsenic-breathing microbes discovered in Pacific Ocean - The Tribune

By **Tribuneindia.com**

WASHINGTON Scientists have discovered microorganisms breathing arsenic—a deadly poison for most living things—in the low-oxygen parts of the Pacific Ocean. The researchers from the University of Washington (UW) in the US analysed seawater samples from a re... [+2438 chars]

Published on: 2019-05-05




Klopp Unsure About Salah after Head Injury, Firmino to Miss Barcelona Clash - News18

By **News18.com**

Newcastle: Liverpool will be without their Brazilian forward Roberto Firmino for Tuesday's Champions League semi-final second leg against Barcelona at Anfield, manager Juergen Klopp said on Saturday.Klopp could also be without another key forward in Mohamed S... [+1371 chars]

Published on: 2019-05-05




Fans Come Up With ROFL 'Game Of Thrones' Memes And Others To Troll SRH After Loss To RCB - NDTVSports.com

By **Ndtv.com**


SunRisers Hyderabad's hopes of qualifying for the IPL 2019 playoffs hang by a thread after a defeat to Royal Challengers Bangalore (RCB) on Saturday night. SRH lost their final league game of the season which means they would need Mumbai Indians to beat Kolka... [+2922 chars]

Published on: 2019-05-05




The Sunday school children: The little-known tragedy of the Sri Lankan Easter attacks

Published on: 2019-05-05



11 of Our Best Weekend Reads

Published on: 2019-05-05



127.0.0.1:5000/recommendation.html/11

We assign the indexes to the corresponding categories as follow:

```
prediction = {0: 'Business', 1: 'Entertainment', 2: 'Politics', 3: 'Sports', 4: 'Technology'}
```

In back-end our recommendation system classifiers results the categories as follow:

```

Classifier name: naive bayes
predicted category:  [3]

Classifier name: support vector machine
predicted category:  [3]

Classifier name: k-nearest neighbour
predicted category:  [3]

Classifier name: logistic regression
predicted category:  [3]

Classifier name: decision tree
predicted category:  [1]

Classifier name: randomforestclassifier
predicted category:  [3]

```

Here we can see that most of the classifier classify the selected news as 'Sports' news which is indexed '3'. Only Decision tree wrongly classify it since its accuracy is 77% only.

We select the most common class predicted among the all of the above classifier as follow:

```
prediction list: Counter({3: 5, 1: 1})
```

Here 5 classifier classify the news as 'sports' and one classify the news as 'Entertainment'.

Therefore we choose the final category as 'sports'.

```

final predicted category:  3
category of selected article:  Sports

```

Now we fetch the top sports news as follow:

```

url = ("https://newsapi.org/v2/top-headlines?country=in&category="
+ "sports"+"&apiKey=6b4e60b44b5746c3a5f75b18fccba442")
response = requests.get(url)
js = response.json()
Sports_articles = defaultdict(dict)
for article in js['articles']:
    Sports_articles[article['title']]={'content':article['content'],
    'urlToImage':article['urlToImage'], 'publishedAt':article['publishedAt'],
    'source':article['source']['name'], 'url':article['url']}

```

```

Sports_contents = []
Sports_image_urls = []
Sports_published_dates = []
Sports_sources = []
Sports_urls = []
Sports_titles = list(Sports_articles.keys())

for title in Sports_articles.keys():
    Sports_contents.append(str(Sports_articles[title]['content']))
    Sports_image_urls.append(str(Sports_articles[title]['urlToImage']))
    Sports_published_dates.append(str(Sports_articles[title]['publishedAt']))
    Sports_sources.append(str(Sports_articles[title]['source']))
    Sports_urls.append(str(Sports_articles[title]['url']))
for i in range(0, len(Sports_published_dates)):
    Sports_published_dates[i] = Sports_published_dates[i][:10]

Sports_news = [Sports_titles, Sports_contents, Sports_image_urls,
Sports_published_dates, Sports_sources, Sports_urls]

```

Now we transform the target news content and selected news content into tfidf vectoriser as follow:

```

stopWords = stopwords.words('english')
Recom_content_set = list()
for i in range(len(Recom_news[0])):
    Recom_content_set.append(str(Recom_news[1][i])[:-20])

tfidf_vectorizer = TfidfVectorizer(stop_words = stopWords)
tfidf_matrix_train = tfidf_vectorizer.fit_transform(iter(Recom_content_set))
tfidf_matrix_test = tfidf_vectorizer.transform([content])

```

Now we find the similarity b/t documents and sort and prints the document similarity as follow:

```

similarity = cosine_similarity(tfidf_matrix_test,tfidf_matrix_train);
print ("\nSimilarity Score [*] ",similarity)
cos_sim = pd.Series(similarity[0])
s_values = cos_sim.sort_values(ascending=False)
s_indexes = cos_sim.sort_index(ascending=False)
indexes = list(s_values.index)
indexes.remove(index)
values = list(s_values.values)
print('indexes:',indexes)
print('values: ',values)

```

```

Similarity Score [*] [[0.01461196 0.          0.03810037 0.          0.88503075 0.206
14703
0.02901825 0.          0.07927886 0.01863661 0.04857983 0.01613178
0.          0.          0.          0.          0.04021479 0.
0.          0.          ]]

```

```

indexes: [4, 5, 8, 10, 16, 2, 6, 11, 0, 18, 12, 7, 13, 14, 15, 3, 17, 1, 19]

```

```

values: [0.8850307452322776, 0.2061470348665893, 0.07927885733567683, 0.04857982847
896469, 0.040214794826134215, 0.03810037202466038, 0.02901825197236782, 0.0186366103
0892842, 0.016131775539641225, 0.01461196267516417, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0]

```

Based on the above sorted similarities score we show the news to user as follow:

Recommended Articles



Fans Come Up With ROFL 'Game Of Thrones' Memes And Others To Troll SRH After Loss To RCB - NDTVSports.com

By NDTV.com

SunRisers Hyderabad's hopes of qualifying for the IPL 2019 playoffs hang by a thread after a defeat to Royal Challengers Bangalore (RCB) on Saturday night. SRH lost their final league game of the season which means they would need Mumbai Indians to beat Kolkata... [+2922 chars]

Published on:2019-05-05



Watch: Virat Kohli Mocks Khaleel Ahmed By Impersonating His Wicket Celebration - NDTVSports.com

By [Ndtv.com](#)

Virat Kohli yet again failed to make a contribution with the bat as Khaleel Ahmed sent him packing in the second over but, for a change, despite the batting failures of Kohli and AB de Villiers, RCB managed to register a rare win in IPL 2019. Victory seemed L... [+1907 chars]

Published on:2019-05-05



IPL 2019 qualification scenarios updated: What SRH, K..ffs with MI, DC and CSK fighting to finish in top two - Firstpost

By [Firstpost.com](#)

Mumbai Indians' win over Sunrisers Hyderabad made them the third team to qualify for the Play-offs. Which leaves four teams fighting for a place in the top four with Royal Challengers having being knocked out. Here's what the other teams need to do to qualify... [+3027 chars]




Anushka Sharma and Virat Kohli return to Mumbai from Bengaluru; see pics - PINKVILLA


By [Pinkvilla.com](#)


Anushka Sharma and Virat Kohli were snapped at the Mumbai airport today. The couple returned from Bengaluru where they had gone for Royal Challengers Bangalore's match against Sun Risers Hyderabad which was held yesterday. Virat's team RCB won the match by 4 ... [+1044 chars]


Now we select the news based on categories also from the given side bar as follow:


Categories

**BUSINESS**
China's war on healthcare costs lures India's biggest drugmaker - Economic Times.
Published on: 2019-05-05

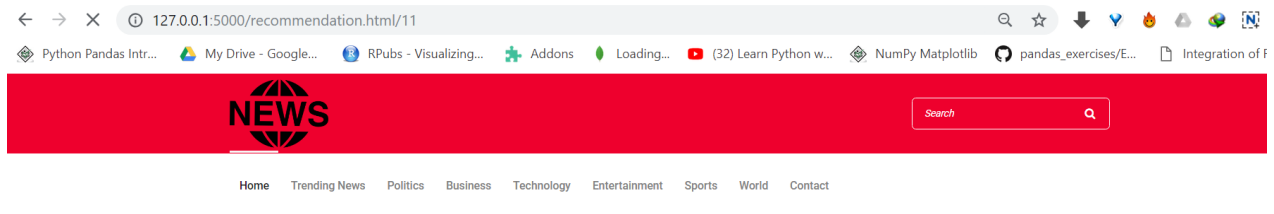
**ENTERTAINMENT**
The Voice Winner: Sumit Saini Lifts the Trophy, Takes Home Prize Money of Rs 25 Lakh - News18
Published on: 2019-05-05

**POLITICS**
Congress will not cut into UP gathbandhan's vote, will damage BJP: Priyanka Gandhi Vadra
Published on: 2019-05-01

**SPORT**
Riqui Puig: Barcelona boss Ernesto Valverde is 'a crack as a manager' - Barca Blaugranes
Published on: 2019-05-05

**TECHNOLOGY**
YouTube Music will soon be able to play songs from your local library - Beetel Bite - Technology News and Reviews
Published on: 2019-05-05

We can also select category news from the header bar as follow:



We also have the option to select world news in world news section in side bar corner as follow:



New York Rain: Vintage Sights and Sounds of a Soaked City

Published on: 2019-05-05



CNN goes inside the devastated St. Anthony's Church after Sri Lanka's Easter Sunday bombings - CNN Video

Published on: 2019-05-05

On each type of news weather it is categorical news or world news our recommendation system provides recommendation.

CHAPTER-5

CONCLUSION

Conclusion -

Recommendation systems are complementary to search and becoming mainstream for driving the business via advertisement or recommending the intended information to the users. Recommended systems is an important technology in combating information overload. Now they don't faces problems associated with real-time computations and data overload. Collaborative filtering and content-based filtering methods have their own problem but we can overcome on them by using the hybrid approach.

In our 'headlines' project we see that 'svm' classifier classify the most accurately with 97-98% accuracy. We find the similarity measures b/w documents using the cosine-similarity method which performs average i.e. cosine-similarity method can't differentiate the document with the same words with different meaning we could use any other algorithms also.

CHAPTER-6

FUTURE WORK

Since data is increasing exponentially day by day and it is predicted that in 2020 there would be 40 PB data generated by us. In this massive amount of data it is very difficult to find the information relevant to us at scale and at real-time. In future we can develop our project to use the distributed computing systems like Apache spark which computes on massive amount of data hence solves the problem of large matrix computation problem in collaborative filtering hence providing the real-time recommendation.. Therefore in future data would be a major economy that is utilized by the spark like computing engines and recommendation system to grow the business.

Our cosine similarity algorithm doesn't perform well when where we word level similarity but have the semantic level difference in text. E.g. doc1 is 'the dog bites the man' and doc2 is 'the man bites the dog', here both docs have the same words but different meaning. In such case the, we could use other algorithms to perform well and find the semantic level difference.

BIBLIOGRAPHY

- [1] Jones, Tim M. “Introduction to Approaches and Algorithms, Recommender Systems” *IBM Developer works*. IBM Corporation, 10 January 2015
 - [2] Lew, Daniel. Sowell, Ben. Steinberg, Leah. Tuladhar, Amrit. “Recommender Systems” *A computer Science Comprehensive Exercise*. Carleton College, Northfield, MN
 - [3]. Statistics How To. Internet. statisticshowto.com/ what is the pearson correlation coefficient, 20 December 2013
 - [4]. Wikipedia article. Internet. “Kmeans Clustering” en.wikipedia.org/wiki/Kmeans_clustering
 - [5]. Sun, Jiankai. Wang, Shuaiqiang. Gao, Byron J. Ma, Jun. “Learning to Rank for Hybrid Recommendation”. Natural Science Foundation of China
 - [6]. Li, Hang. “A Short Introduction to Learning to Rank” *Information Based Induction Sciences and Machine Learning*. Microsoft Research, 10 October 2011
 - [7]. Cao, Zhe. Tao, Qin. Liu, TieYan. Tsai, MingFeng. Li, Hang. “Learning to Rank: From Pairwise Approach to Listwise Approach” *A probabilistic Framework for Learning to Rank*. Proceedings of the 24th International Conference on Machine Learning, Corvallis, OR, 2007
 - [8]. Bappalige, Sachin. “An Introduction to Apache Hadoop for BigData”. Internet. opensource.com/life/14/8/introapachehadoopbigdata
- 59
- [9]. Zaharia, M., Chowdhury M., Franklin M., Shenker S., Stoica I. Spark: Cluster Computing with Working Sets, ACM, 2010
 - [10] Adrien Sieg ‘Text Similarities: Estimate the degree of similarity between two texts’. <https://medium.com/@adriensieg/text-similarities-da019229c894>