**INTRODUCTION**

With the rapid growth of online information, text categorization has become one of the key

techniques for handling and organizing text data. Automatic text categorization is a process of

automatically assigning text documents to different categories [1]. With the increased use of

internet, the generation of huge online documents in different languages has also increased. This

increased use of computers has led to great research and advancement in Machine learning.

Till now there are number of researches have been done on English text documents [3] as it is

a widely spoken language. Proposed work presents the automatic text categorization of Marathi

documents, mainly news articles available in Marathi using Machine Learning Algorithms. The

system will categorize the news articles based on different categories of news like sports, politics,

entertainment, editorial, local etc

From literature survey it is found that, digitization is at its peak nowadays. Digitization means

converting text, photos, or voice sound into a digital form that can be processed by a computer.

Hence, huge online data is being generated every day. Different News publishers have also moved

towards digitization and hosted their own news websites where people can read news on daily basis

online with the help of internet connectivity. According to 2017 report of IRS released [27] on 18th

Jan 2018, 39% of Indians (above 12 years) read newspapers and 20% of all newspaper readers in

50 lakhs plus population in towns read newspapers online and this number is increasing day by

day. According to the report, one of the Marathi daily occupies the sixth place with a total

readership of 1,80,66,000 among all the top readership of regional language dailies [27]. That

means in coming years more and more people will prefer to read online news in their native

languages and handling such huge amount of generated regional languages data will become the

challenge.

In proposed work introduces a system called Marathi news categorization as one of the solution

for automatically categorizing Marathi news articles. The system mainly includes gathering of

various online Marathi news from news websites, forming the training corpus, pre-processing of

the data, TFIDF calculations on training-testing dataset and finally using K-NN on test data with

best chosen distance measure formula for classifying test news article correctly. As a result of this

automatic text categorization, the user can fetch the news of his own “news category” choice in

more easy and faster manner.

**1) Pre-requisites :-**

1. python

2. PyCharm(Professional free trial version available for 30 days)

3. Internet connection in case you want to crawl newspaper website.

**Note:** Refer following YouTube videos to know how to work with PyCharm.

<https://www.youtube.com/watch?v=HBxCHonP6Ro&list=PL6gx4Cwl9DGAcbMi1sH6oAMk4JHw91mC_>

**2) First thing first :-**

Following are the list of useless files and folders. You might want to delete them to reduce the confusion.

**Folders** : eco\_dupli, TextTest\_eco\_new, TextTest\_ecod, TextTest\_en\_newt, TextTest\_entd, TextTest\_sprtd

**Files**: Deciding\_Kvalue.py, dict.txt, output\_old.txt, readword.py, testnewsarticles.txt, tfidf.py

**3) Significance of each remaining files and folders :-**

**3.1**

|  |
| --- |
| economy\_train\_data.py |
| entertain\_train\_data.py |
| sport\_tain\_data.py |
| test\_news\_data.py |

Above mentioned all files are used for collecting training and testing data samples (i.e. news articles of 3 different categories: Sports, Entertainment & Economy ) by crawling through site <https://www.esakal.com/> (**logic of crawler is used in these files**)

That explains the usage of following folders –

|  |  |
| --- | --- |
| Folder Name |  |
| Text | Has 3500 training data samples of sports category |
| TextEconoy | Has 3500 training data samples of economy category |
| TextEnter | Has 3500 training data samples of entertain category |
| TextTest | Has 1750 testing data samples containing mixture of all 3 categories. |

Note –

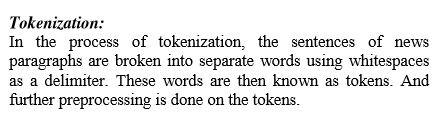
* I have used news articles as a data samples of this research.
* You can reduce the amount of training and testing data samples to get better understanding of code(i.e. Create a small subset of each folder and work on small dataset for initial stage)

**3.2**

|  |  |
| --- | --- |
| Filename | Significance |
| noun.txt | Dictionary of nouns in Marathi |
| adjective.txt | Dictionary of adjectives in Marathi |
| adverb.txt | Dictionary of adverbs in Marathi |
| verb.txt | Dictionary of verbs in Marathi |

- Above all .txt files are used in .py files that are mentioned in step 3.1

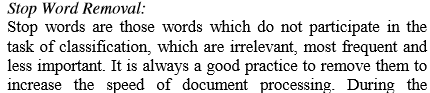
- These files are used for tokenization step of pre-processing.



**3.3**

|  |  |
| --- | --- |
| Filename | Significance |
| stopword.txt | Dictionary of stop words in Marathi |

- This file is used for removing stop words from the data samples.



**Note –**

* Above all files are used for collecting and preprocessing all the data samples of research and it is a onetime activity.
* You can refer these files if need to collect more or recent data samples(news articles) form newspaper sites.
* But you need all folders mentioned in 3.1 **(Text, TextEconoy, TextEnter, TextTest)** for further conduction of research.

**4) Q. Which all final python files are required to carry out the experiment and how to run them?**

**Ans - Actual file used for final research -**

**4.1 Start with running any of the files from below –**

|  |  |
| --- | --- |
| FileName | Distance Measure of K-NN algorithm |
| finalprediChebyshev.py | Includes logic of Chebyshev distance Measure |
| finalprediCosine.py | Includes logic of Cosine distance Measure |
| finalprediction.py | Includes logic of Euclidean distance Measure |
| finalprediJaccard.py | Includes logic of Jaccard similarity Measure |
| finalprediManhatt.py | Includes logic of Manhattan distance Measure |

**Note -**

Check out following sites to know more about Distance measures of K-NN algorithm –

<https://dataaspirant.com/2015/04/11/five-most-popular-similarity-measures-implementation-in-python/>

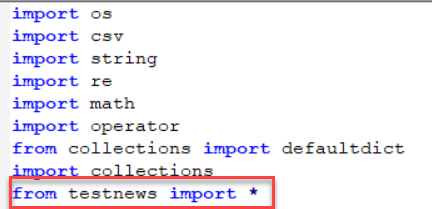
<https://www.saedsayad.com/k_nearest_neighbors.htm>

**Example**

If you want to categorize the test sample data to its correct news category using K-NN algorithm with Euclidean distance measure then follow the following order to run the code.

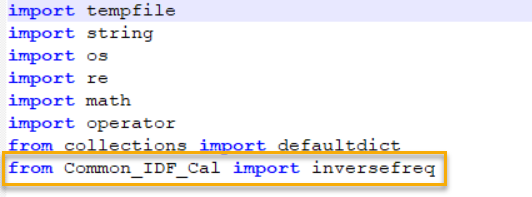
1. Run file “**finalprediction.py**” in PyCharm -

**Screenshot of import section in finalprediction.py -**



**finalprediction.py** file imports ‘**testnews.py’** file which in turn imports variables from file ‘**Common\_IDF\_Cal.py’**

**Screenshot of import section in testnews.py -**



So, when we run **finalpredication.py** file it gives call to **testnews.py** and **Common\_IDF\_Cal.py.**

**Notes:-**

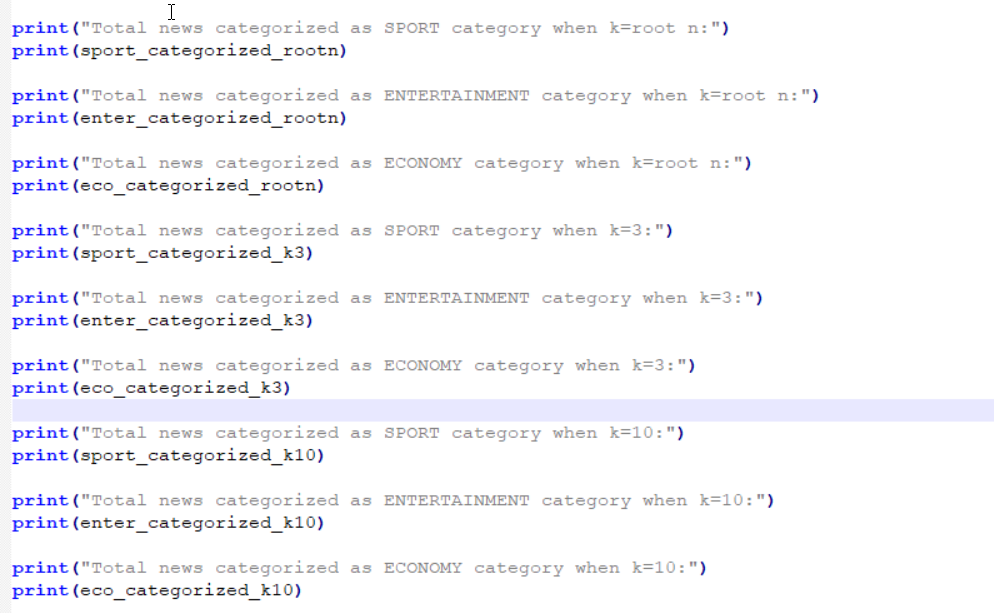
1. **Common\_IDF\_Cal.py** python file is used for creating final **IDF** vector for each test sample(i.e. test news article which needs to be categorized)
2. **testnews.py python** file is used for calculating **TF** & finally **TF-IDF** vector for each test sample and at the end the final vector space for test news article is created.
3. **finalpredication.py** python file includes calculation of TF, IDF & TF-IDF of all training samples and creates final vector space for each training sample**.**

**Also,** it includes logic for K-NN classification using Euclidean distance measure to correctly categorize each test sample available in “**TextTest**” folder.

Experiment is carried out using 3 different values of K in K-NN (i.e. when K = 3, K = 10 and K = root of total number of training data) to decide the best value of K at the end.

1. All other files finalprediChebyshev.py, finalprediCosine.py, finalprediJaccard.py, finalprediManhatt.py follow the same workflow but has different distance measure logic.
2. Note that some of the “print” statement in these python files are useless and used just for keeping the track of code.

That’s why the last lines of code appear as :



To know more about TF-IDF classification follow –

1. <https://www.geeksforgeeks.org/tf-idf-model-for-page-ranking/>
2. <https://towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089>
3. <https://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html>
4. <https://www.coursera.org/lecture/ml-foundations/calculating-tf-idf-vectors-1rg5n>