**Project Requirement and Specification**

**on**

**Sentiment Analysis Using Machine Learning**

**(CSE(AIML) III Semester Mini project PCS-307)**

**2020-2021**

****

**Submitted to: Submitted by:**

Mr. B. P. Dubey Devendra Johari

(CC-CSE(AIML)-D-III-Sem) Roll. No.: 2015240

**Guided by:** CSE(AIML)-D-III-Sem

Dr. Rahul Nijhawam Session: 2020-2021

(Resource Person)

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**GRAPHIC ERA DEEMED TO BE UNVERSITY, DEHRADUN**

---------------------------CONTENTS----------------------------

* + - About Project
    - System Requirements
    - Techniques and Algorithms used
    - Machine Learning Model Evaluation
    - Steps of Machine Learning
    - Experimental Results
    - Conclusion
    - Source Code
    - Refrence

--------------------------------------------------------------------------------------------

* 1. **About Project**

[Sentiment analysis](https://monkeylearn.com/sentiment-analysis/) is a [machine learning](https://monkeylearn.com/machine-learning/) tool that analyzes texts for polarity, from positive to negative. By training machine learning tools with examples of emotions in text, machines automatically learn how to detect sentiment without human input.

Machine learning allows computers to learn new tasks without being expressly programmed to perform them. Sentiment analysis models can be trained to read beyond mere definitions, to understand things like, context, sarcasm, and misapplied words.For example:

“Super user-friendly interface. Yeah right. An engineering degree would be helpful.”

Out of context, the words ‘super user-friendly’ and ‘helpful’ could be read as positive, but this is clearly a negative comment. Using sentiment analysis, computers can automatically process text data and understand it just as a human would, saving hundreds of employee hours.

Similarly, we use Sentiment Analysis here to predict Reviews of peoples on Movies as a positive or negative comment.  We also use Sentiment Analysis to process customer service tickets, categorize them in order of urgency, and automatically route them to the correct department or employee. Or, to analyze thousands of product reviews and social media posts to Gauge Brand Sentiment.

* 1. **Requirement of Project**
     1. **Database Requirement**

The "Large Movie Review Dataset" is used in this project. It has a collection of 50,000 reviews from IMDB, with 25,000 positive reviews and 25,000 negative.

The following dataset is available on following link. You can download and decompress the training and testing from: <https://ai.stanford.edu/~amaas/data/sentiment/>. There will be one folder name "aclImdb", in which contains both "train" and "test" folders.

#### Publications using the dataset:

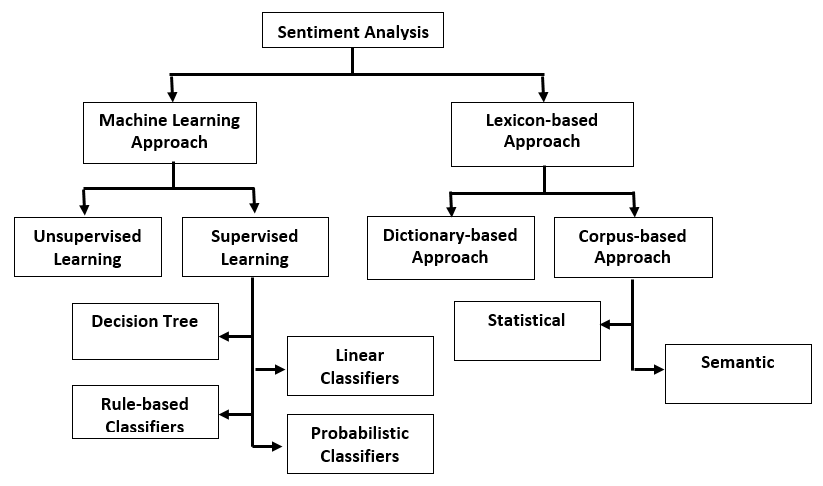
Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. (2011). Learning Word Vectors for Sentiment Analysis. The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011).

* Benchmark Accuracy for the dataset in 2011 is **88.90%**.
  + 1. **Software Requirement**
    - **MATLAB**
    - **Python (Language)**

**Libraries of Python used :-**

* + - nltk(Natural Language Toolkit)
      * NLTK is a leading platform for building Python programs to work with human language data.
      * It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.
    - collections
      * This module implements specialized container datatypes providing alternatives to Python’s general purpose built-in containers, [dict](https://docs.python.org/3/library/stdtypes.html" \l "dict" \o "dict), list, set, and [tuple](https://docs.python.org/3/library/stdtypes.html" \l "tuple" \o "tuple).
    - sklearn (scikit-learn)
      * Most Important module in Machine Learning .
      * Thislibrary contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.
    - itertools
      * The module standardizes a core set of fast, memory efficient tools that are useful by themselves or in combination. Together, they form an “iterator algebra” making it possible to construct specialized tools succinctly and efficiently in pure Python.
    - string
      * + It contains a single utility function – capwords(s, sep=None).
        + It split the specified string into words using str.split().
        + It capitalizes each word using str.capitalize()
        + It joins the capitalized words using str.join().
    - pandas
      * **pandas** is a Python package that provides fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive
      * It aims to be the fundamental high-level building block for doing practical, **real world** data analysis in Python.
      * pandas is well suited for many different kinds of data:
        + Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
        + Ordered and unordered (not necessarily fixed-frequency) time series data
        + Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels.
        + Any other form of observational / statistical data sets. The data actually need not be labeled at all to be placed into a pandas data structure
    - spaCy
      * **spaCy** is a free and open-source library for **Natural Language Processing** (NLP) in Python with a lot of in-built capabilities
      * It’s becoming increasingly popular for processing and analyzing data in NLP. Unstructured textual data is produced at a large scale, and it’s important to process and derive insights from unstructured data.
      * To do that, you need to represent the data in a format that can be understood by computers. NLP can help you do that.
    - re
      * A **Re**gular **Ex**pression (RegEx) is a sequence of characters that defines a search pattern.
    - os
      * This module provides a portable way of using operating system dependent functionality.
      * If you just want to read or write a file see [open()](https://docs.python.org/3/library/functions.html#open),
      * if you want to manipulate paths, see the [os.path](https://docs.python.org/3/library/os.path.html" \l "module-os.path" \o "os.path: Operations on pathnames.) module,
      * if you want to read all the lines in all the files on the command line see the [fileinput](https://docs.python.org/3/library/fileinput.html" \l "module-fileinput" \o "fileinput: Loop over standard input or a list of files.) module.
  1. **Techniques and Algorithms used -:**

There are a number of techniques and complex algorithms used to command and train machines to perform sentiment analysis. There are pros and cons to each. But, used together, they can provide exceptional results. Below are some of the most used algorithms.



* + - **Multinomial Naive Bayes**

Naive Bayes is a fairly simple group of probabilistic algorithms that, for sentiment analysis classification, assigns a probability that a given word or phrase should be considered positive or negative.

Naive Bayes is a fairly simple group of probabilistic algorithms that, for sentiment analysis classification, assigns a probability that a given word or phrase should be considered positive or negative.

Essentially, this is how Bayes’ theorem works. The probability of A, if B is true, is equal to the probability of B, if A is true, times the probability of A being true, divided by the probability of B being true:

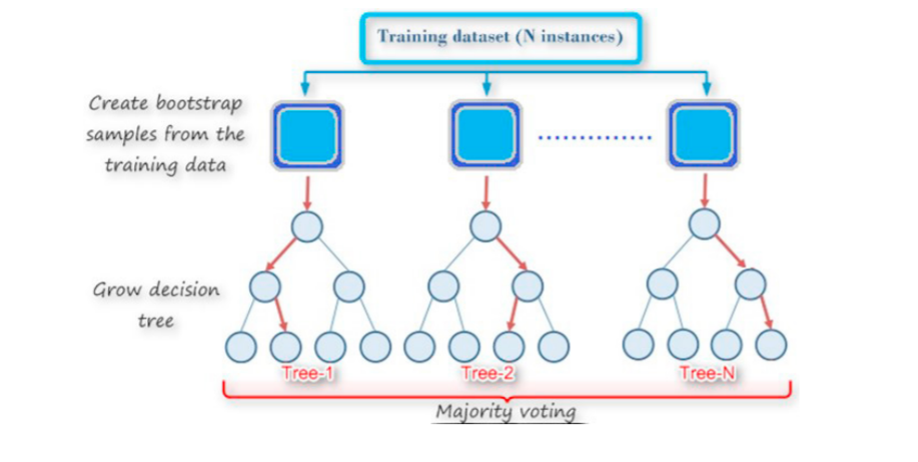
P(A|B) = (P(B|A) x P(A) )/ P(B)

But that’s a lot of math! Basically, Naive Bayes calculates words against each other. So, with machine learning models trained for word polarity, we can calculate the likelihood that a word, phrase, or text is positive or negative.

* + - **Random Forest Classifier**

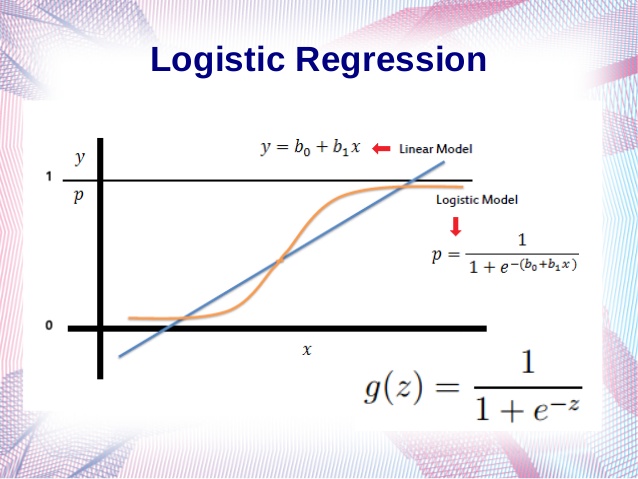
In random forest classification method, many classifiers are generated from smaller subsets of the input data and later their individual results are aggregated based on a voting mechanism to generate the desired output of the input data set. This ensemble learning strategy has recently become very popular. Before RF, Boosting and Bagging were the only two ensemble learning methods used. RF has been extensively applied in various areas including modern drug discovery, network intrusion detection, land cover analysis, credit rating analysis, remote sensing and gene microarrays data analysis etc...

The random forest algorithm is one of the best among classification algorithms - able to classify large amounts of data with accuracy. It is an ensemble learning method for classification and regression that constructs a number of decision trees at training time and delivers the class that is the mode of the classes output by individual trees.

****

* + - **Logistic Regression**

Sentiment Analysis is a method for judging somebody's sentiment or feeling with respect to a specific thing. It is utilized to recognize and arrange the sentiments communicated in writings. The web-based social networking sites like twitter draws in a huge number of clients that are online for imparting their insights in the form of tweets or comments. The tweets can be then classified into positive, negative, or neutral. In the proposed work, logistic regression classification is used as a classifier and unigram as a feature vector. For accuracy, k fold cross validation data mining technique is used. For choosing precise training sample, tweet subjectivity is utilized. The idea of Effective Word Score heuristic is likewise presented to find the polarity score of words that are frequently used. This additional heuristic can speed up the classification process of sentiments with standard machine learning approaches.



* + - **Linear Support Vector Machine (SVM)**

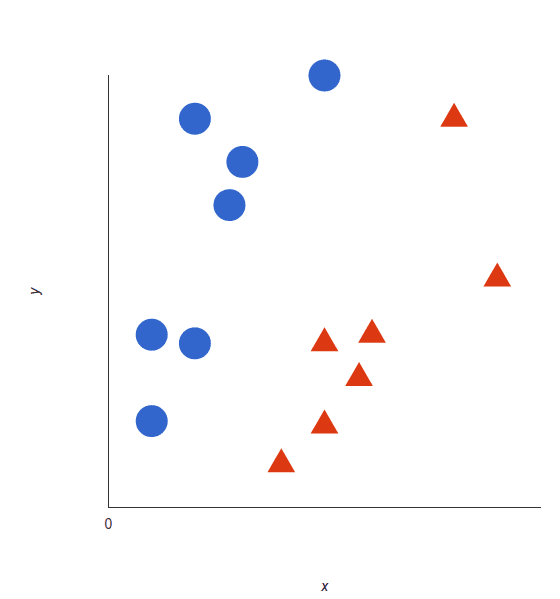
A support vector machine is another supervised machine learning model, similar to linear

regression but more advanced. SVM uses algorithms to train and classify text within our sentiment

polarity model, taking it a step beyond X/Y prediction.

For a simple visual explanation, we’ll use two tags: red and blue, with two data features: X and Y.

We’ll train our classifier to output an X/Y coordinate as either red or blue.

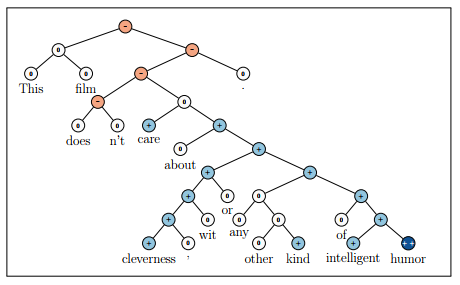


The SVM then assigns a hyperplane that best separates the tags. In two dimensions this is simply a line (like in linear regression). Anything on one side of the line is red and anything on the other side is blue. For sentiment analysis this would be positive and negative.

* + - **Decision Tree Algorithm**

Decision trees are supervised methods, so they need to be trained on some annotated data. Thus the general idea is the same as for any text classification: given a set of documents (for instance represented as TFIDF vectors) together with their labels, the algorithm will calculate which how much each word correlates with a particular label.

For instance it might find that the word "excellent" often appears in documents labeled as positive, whereas the word "terrible" mostly appears in negative documents. By combining all such observations it builds a model able to assign a label to any document.

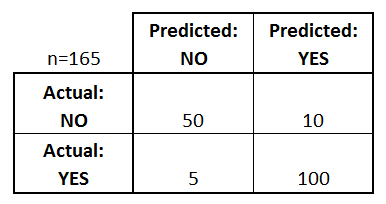


* 1. **Machine Learning Model Evaluation -:**

After you develop a machine learning model for your predictive modeling problem, how do you know if the performance of the model is any good?

There are various parameters through which we can predict our model is good or not.

* 1. Accuracy
     + Classification Accuracy is what we usually mean, when we use the term accuracy. It is the ratio of number of correct predictions to the total number of input samples.
     + Accuracy = Number of Correct Predictions / Total Number of Input Samples
  2. Confusion Matrix
     + Confusion Matrix as the name suggests gives us a matrix as output and describes the complete performance of the model



* + - There are four important terms - :
      * **True Positives** : The cases in which we predicted YES and the actual output was also YES.
      * **True Negatives** : The cases in which we predicted NO and the actual output was NO.
      * **False Positives** : The cases in which we predicted YES and the actual output was NO.
      * **False Negatives** : The cases in which we predicted NO and the actual output was YES.
    - Accuracy for the matrix can be calculated by taking average of the values lying across the**“main diagonal”.**
    - Accuracy = True Positive + True Negative / Total Sample
  1. F1 Score
     + *F1 Score is used to measure a test’s accuracy*
     + F1 Score is the Harmonic Mean between precision and recall. The range for F1 Score is [0, 1]. It tells you how precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances).
     + High precision but lower recall, gives you an extremely accurate, but it then misses a large number of instances that are difficult to classify. The greater the F1 Score, the better is the performance of our model. Mathematically, it can be expressed as :

C:\Users\devjo\Downloads\1__pYttqYh8w-EpLxMi84H8A.gif

* 1. Precision
     + It is the number of correct positive results divided by the number of positive results predicted by the classifier.

C:\Users\devjo\Downloads\1_wMIDfGwT9bA6HezvYhdbpQ.gif

* 1. Recall
     + It is the number of correct positive results divided by the number of **all**relevant samples (all samples that should have been identified as positive).

C:\Users\devjo\Downloads\1_gIlQMZBPjtUHWkwOhRG3KA.gif

* 1. **Steps of Machine Learning**
     + **Import Libraries**
     + **Load Dataset**
     + **Data Preprocessing**
     + **Train Model** 
       - * **Multinomial Naive Bayes**
         * **Random Forest Classifier**
         * **Logistic Regression**
         * **Linear Support Vector Classifier**
         * **Decision Tree Algorithm**
     + **Test Model using Test Dataset**
     + **Model Evaluation**
     + **Test Model using Human Generated Preview**
  2. **Experimental results**

Performance of Different Models are as follows -:

* + - Multinomial Naive Bayes Classifier
      * + Accuracy : 83.31%
        + Precision : 86.86%
    - Random Forest Classifier
      * + Accuracy : 85.16%
        + Precision : 86.03%
    - Logistic Regression
      * + Accuracy : 88.34%
        + Precision : 88.13%
    - Linear Support Vector Machines
      * + Accuracy : 87.26%
        + Precision : 88.12%
    - Decision Tree Algorithm
      * + Accuracy : 71.04%
        + Precision : 71.30%
  1. **Conclusion**

As you see in above results , We can suggest Best Machine Learning Model for our Movie Review dataset according to their Accuracy and Prediction Value. These values suggest how accurate and precisely our model gives output. As in about section We mentioned Benchmark Accuracy for the dataset in 2011 is **88.90%**. So as we tested different model we find Logic Regression Model is most accurate model of Accuracy **88.34**% which is pretty close to Benchmark Accuracy.

One of the major improvements that can be incorporated as we move ahead in this project is to merge words with similar meanings before training the classifiers[3]. Another point of improvement can be to model this problem as a multi-class classification problem where we classify the sentiments of reviewer in more than binary fashion like “Happy”, “Bored”, “Afraid”, etc[14]. This problem can be further remodeled as a regression problem where we can predict the degree of affinity for the movie instead of complete like/dislike.

* 1. **Source Code**

The whole source code is available on Github . The Github Link is as follows-: <https://github.com/DevendraJohari24/MachineLearning/tree/main/Sentiment%20Analysis>

**REFERENCE**

1. “Machine Learning For Dummies” by John Paul Mueller and Luca Massaron
2. “Hands-On Machine Learning with Scikit-Learn , Keras and TensorFlow”, 2nd Edition by Aurelien Geron.
3. “Python Machine Learning: Machine Learning and Deep Learning with Python , scikit-learn, and TensorFlow 2”, 3rd Edition by Sebastian Raschka.
4. “Python Programming: An Introduction to Computer Science”, 3rd Edition by John Zelle
5. Websites -: <https://www.geeksforgeeks.org> , <https://www.w3schools.com/python/python_ml_getting_started.asp> , <https://github.com>