Automatic Anecdote Detection

A Project Report

Submitted by

Piyush Yadav	**111408069**
Shirishkumar Togarla	**111408061**
Shradhit Subudhi	**111403074**

in partial fulfillment for the award of the degree

of

B.Tech Information Technology

Under the guidance of

Prof. Aparna Biswas

College of Engineering, Pune

DEPARTMENT OF COMPUTER ENGINEERING

AND

INFORMATION TECHNOLOGY,

COLLEGE OF ENGINEERING, PUNE-5

May, 2011

DEPARTMENT OF COMPUTER ENGINEERING

AND

INFORMATION TECHNOLOGY,

COLLEGE OF ENGINEERING, PUNE

CERTIFICATE

Certified that this project, titled “Automatic Anecdote Detector” has been successfully completed by

Piyush Yadav	**111408069**
Shirishkumar Togarla	**111408061**
Shradhit Subudhi	**111403074**

and is approved for the partial fulfillment of the requirements for the degree of “B.Tech. Computer Engineering/ Information Technology”.

SIGNATURE
NAME OF GUIDE
Project Guide
Department of Computer Engineering
and Information Technology,
College of Engineering Pune,
Shivajinagar, Pune - 5.

SIGNATURE
NAME OF HOD
Head
Department of Computer Engineering
and Information Technology,
College of Engineering Pune,
Shivajinagar, Pune - 5.

Abstract

Stories and anecdotes are often used by teachers, speakers, authors, and leaders to communicate abstract ideas and illustrate concepts. Stories by customers in business conversations (such as customer support) in social media channels are also important sources of information. Discovering these anecdotes from texts such as books, articles and blogs posts are currently done by human researchers. In this project, we aim to automatically identify these stories from such sources, so as to:

- Improve the productivity of such researchers
- Provide a story-detection component that can be used by other text analytics systems

To do this, we plan to:

- Define what a story is and its analytical components.
- Use a Machine Learning based approach to identify potential regions of stories in unstructured text sources
- Build an application to do this.

The application that we aim to build will attempt to:

- Automatically identify regions of anecdotes (think of it as putting a $< anecdote >< /anecdote >$ tag around such regions of text)
- Classify them by certain labels (such as sports, films, business etc.) - these labels will be provided by us.

Contents

List of Figures **ii**

1 Introduction **1**

1.1 Problem Statement 2

1.2 Aim and Objective 2

1.3 Methodology 3

1.4 Overview Of Report 3

2 Literature Survey **4**

2.1 Introduction of Literature Survey 4

2.2 Summarizing Algorithm 5

2.3 Text Extraction Algorithm 6

3 Referred Research Papers **7**

3.1 Text Segmentation into Paragraphs Based on Local Text Cohesion 7

3.1.1 Abstract 7

3.1.2 Approach 8

3.2 Measuring similarity between sentences 9

3.2.1 Abstract 9

3.2.2 Approach 9

4 Software Requirement Specification **10**

4.1 Introduction 10

4.1.1 Purpose 10

4.1.2 Document Conventions 10

4.1.3 Product Scope 10

4.2 Product Perspective 11

4.2.1 Product Functions 11

4.2.2 Design and Implementation Constraints 11

4.3 Functional Requirements 11

4.4 Other Nonfunctional Requirements 12

4.4.1 Performance Requirements 12

4.4.2 Safety and Security Requirements 12

4.4.3 Software Quality Attributes 12

5 Methodology 13

5.1 What is automatic anecdote detection? 13

5.2 How did we approach the problem of Automatic Anedote Detection? 13

6 Models Used 15

6.1 ADABoost 15

6.2 Random Forest 15

6.3 Naive Bayes 16

6.4 eXtreme Gradient Boost- XGBoost 17

6.5 Ensemble : Voting Classifier 18

6.6 Logistic Regression 20

6.7 Support Vector Machine 21

7 Implementation 23

7.1 Scraping the Dataset 23
7.2 Identifying the start of anecdote 23
7.3 Machine Learning Model 24
7.4 Identifying the start 25
7.5 Identifying the body and end of the anecdote 25
7.6 User Interface 26

8 Result & Analysis 28

9 Conclusion 30

9.1 Product Scope 30
9.2 Future Scope 31

List of Figures

6.1 Random Forest 16

6.2 Naive Bayes 17

6.3 XGBoost 18

6.4 Voting Classifier 19

6.5 Logistic Regression 21

6.6 Kernel RBF Function 21

6.7 SVM classification 22

7.1 Attributes of Emsemble Model 24

7.2 Login Screen 26

7.3 Registration Screen 27

7.4 Home Screen 27

8.1 Accuracy Comparison 29

8.2 Overall Comparison 29

Chapter 1

Introduction

The reason why humans find anecdotes interesting is because they find it very relatable to their personal self. Anecdotes are often about what we have personally experienced or heard about. Because anecdotes are exceptional and not at predictable we bother exchanging conversations around them. Anecdotes ate used as reinforcements to shore up our arguments. They also come to notice because of the already cherished ideas, we have!

Man prefers to believe what he prefers to be true. –Francis Bacon

Anecdote Detector is a research project, hence this document would help the researchers to understand the approach taken by the project to solve the problem of anecdote detections. This document would also be helpful for the non-technical/ lay person to understand the world of NLP, it's application and capabilities. The currently manual process of analyzing text and detecting these anecdotes can now be automated and save a lot of time while having an equivalent, if not better, efficiency and accuracy.

Anecdote: a short amusing or interesting story about a real incident or person.

1.1 Problem Statement

An anecdote is a short amusing or interesting story about a real person or incident. Our software aims to recognize such anecdotes or independent plots of stories, from a random text or a larger plot. Stories and anecdotes are often used by teachers, speakers, authors, and leaders to communicate abstract ideas and illustrate concepts. Stories by customers in business conversations (such as customer support) in social media channels are also important sources of information. Such anecdotes of important/influential personalities are also used as motivational triggers in employment circles, colleges, schools and almost everywhere. Currently, the scenario is majorly manual when it comes to creating these anecdotes, discovering these from texts is currently done by human researchers. However, we aim to automate this process by providing a chunk of data(a book, article, blog, etc) and get the output as anecdotes or short independent stories from it.

- Get an article or a document as input
- Analyze the text
- Extract relevant features using the feature extractor
- Input the text to the model along with the features
- Get textual output from the model with the whole or parts of the text tagged to indicate stories/anecdote.

1.2 Aim and Objective

Using the problem of Anecdote Detection using Machine Learning Techniques. We aim to get an accuracy of 70% in finding the anecdote.

1.3 Methodology

- Finding the probability of a sentence being an anecdote using machine learning models i.e. SVM, Random Forest, Artificial Neural Network - Recurrent Neural Network
- Finding the start of the anecdote in the given text using procedural algorithm and predition from the Machine Learning Model.
- Identifying the body of the anecdote using custom heuristic.
- Improving the accuracy of the model.

1.4 Overview Of Report

This document is contains ten chapters. The first chapter, that you have already read contains the introduction, problem statement, aims, objectives and methodology of the our research project. The second chapter contains the literature survey that we carried out last semester i.e. 7th semester. Chapter three contains the list of referred papers. In the eight chapter we have mentioned the Software Requirement. Chapter five contains the methodology for the project. In the sixth chapter, we try to explain about the all the models that we applied to the Automatic Anecdote Generator. We wrote down the elaborate methodology in the seventh chapter i.e. under Implementation. In the eighth chapter we wrote about the result that we gathered in the process. In the ninth chapter we have mentioned the Conclusion of the project. Conclusion is mentioned in the ninth chapter.

Chapter 2

Literature Survey

2.1 Introduction of Literature Survey

While looking for similar projects it seems that there has not been much research done in the sequential classification arena which can be useful in our use case scenario of Automatic Anecdote Detection. Hence after looking at the research projects/ papers that are in a close proximity to this project of anecdote detection, these were the two models that were close to our project:

'Classification of Text' using Machine learning, pos tagging using NLTK and model training using ensemble learning will be the crux on which our project: Automatic Anecdote Detector will be based.

However, the present technologies that help us achieve our goal includes 'Summarizing Algorithm' and 'Text Extraction algorithms'. We finally compare these with our approach.

2.2 Summarizing Algorithm

This is a naive summarisation algorithm which tries to summarize the given text based on the importances and intersection of the sentences. The intersection of two sentences is the normalized similarity score of the two sentences. The intersection is found by splitting each sentence into words/tokens and then counting the number of common tokens and then normalizing by taking the average of the lengths of the two sentences. The main part of the algorithm is the sentences dictionary. It calculates the score for each sentence of the given text. The score is calculated and composed by the following steps: Step 1) In this step, the text is split into sentences and the intersection of each sentence with the other is stored in a 2D array. For example matrix[1][2] will store the value of the intersection between second and third sentence (assuming 0-based indexing).

Step 2) The dictionary stores the sentences and their corresponding individual scores. This score is the total score. Then the score for each vertex in the graph is calculated by adding all the edges that connect the vertices.

Step 3) This is the last step for generating the summary of the text. The text is splitted into paragraphs and then the best sentence is chosen based on the dictionary values. Each paragraph represents subset of the graph, thus the best node is selected from each subset.

.

2.3 Text Extraction Algorithm

The goal of text extraction is to automatically extract useful structured from textual data or semistructured data. The data from outside sources can be automatically separated into required categories using machine learning.

Natural Language Processing Toolkit(NLTK) and Stanford Corenlp, these libraries include pretrained models which categorize the text and extract relevant information from text such as the word belongs to an organization, a person, place, etc. This type of extraction is popularly known as Named Entity Recognition where the named entities, i.e. the proper nouns or common nouns are recognized to belonging to a particular category which helps a lot in automatic question answering systems based on the text. Recently the research on deep learning has found astonishing results using the Recurrent Neural Networks especially LSTM which stands for Long Short Term Memory Networks. The LSTMs are by default built to remember long term relationships between the words, even when the words are separated through a large distance where the usual RNN networks suffer.

Other uses of text extraction include indexing large document collections, topic wise document classification, text management, text data analysis, relationship extraction, clustering social networking website data, fact extraction, etc

Chapter 3

Referred Research Papers

3.1 Text Segmentation into Paragraphs Based on Local Text Cohesion

Author : Igor A. Bolshakov and Alexander Gelbukh

3.1.1 Abstract

Automatic text segmentation problem[2] can be broadly divided into :
1) ***Thematic segmentation :***
Similar to topic wise classification of documents where each document contains some textual data. The document is classified into a category like News, Comic, Scientific discoveries, etc. The documents in this type are generally categorized based on the typical vocabulary,i.e. occurrence of words which are specific to the categories into which they are classified.
2) ***Paragraph Splitting :***
The problem of paragraph splitting is based on the context resolution. That is a group of sentences in the text usually belong to a particular context which separates them from other sentences. Thus grouping these similar sentences into paragraphs enhances the readability as well as the increases the performance, speed of the read and the reader can quickly locate and refer to the

context based on the splitting of the text into paragraphs. The procedure includes finding the strong cohesive links based on the syntactic, semantic and anaphoric links and then comparing these with a collocation database, smoothing, normalization and comparing with a specially constructed threshold.

3.1.2 Approach

The problem of splitting a text into paragraphs is addressed in this paper. The splitting of text into paragraphs is determined by the strong cohesive links within a paragraph and weak cohesive links between the paragraphs.There is no formal definition of the exact splitting of the text into paragraphs. The text is often spitted based on the context, i.e. if the sentences speak about a common entity, idea, place, etc then the sentences are combined into one paragraph. Approximation of text cohesion is done using the three links :

Syntactic Link These links are similar to the ones in the dependency grammars. The words that form a dependency sub-tree often show a kind of syntactic link. Collocations are nothing but the syntactic links between two words, often juxtaposed and have a higher frequency of occurring together as compared to a coincidence or chance.

Pseudo Syntactic Link These are the links that are similar to syntactic links but hold between words of different sentences.For example She asked the officer for help. He denied.

Semantic Link When there is a similarity between sentences in terms of the semantics, then there is a strong semantic link between them. Semantic links are more common than anaphoric links.

Combining probabilistic approximations based on the distance between the words and the above links, the splitting of text into paragraphs can be done using a large collocation database like CrossLexica which is a Russian collocation database. Unfortunately there is no large collocation database in English language and hence we cant use this approach.

3.2 Measuring similarity between sentences

Author : Thanh Ngoc Dao and Troy Simpson

3.2.1 Abstract

This paper finds an approach to compute similarity between two sentences using meanings of two words. It finds the semantic score of the two sentences which gives an estimate of the similarity between two sentences which in turn can be used to find the context of the sentences.

3.2.2 Approach

In this paper the approach taken to compute the semantic similarity between two sentences is stated as follows:

Tokenizing each word in the sentence and then using a Parts of Speech Tagger (such as nltk.pos$_{t}ags()inNLTKlibrary), totagtheindividualwordswiththeirrespec$

Chapter 4

Software Requirement Specification

4.1 Introduction

4.1.1 Purpose

This document gives details of the Software requirement for Automatic Anecdote Detector. It defines what the problem is and what problem the complete solution has to solve. The purpose of this SRS document is to verify that all specifications are correct and verified.

4.1.2 Document Conventions

The purpose of the project has been mentioned above. Process model : Spiral model has been used for our research product as the spiral model lets the developers use multiple models and work on their project cumulatively like iterative, waterfall, prototyping, etc.

4.1.3 Product Scope

1) Scope of this project is very broad in terms of research on the topic of classifying anecdotes from a given text of multiple stories which may or may not be related to each other. 2) It can be used by researchers who need such

functionalities either for AI objectives or for topics related to Psychology and motivation since currently the process is highly manual.

4.2 Product Perspective

Anecdote Detector enables us to detect and classify independent plots of short stories involving renowned examples and people from a given dataset of a book or a story.

4.2.1 Product Functions

Initially the software is trained with already retrieved data to set the classification standards. After that a new data set is provided and it is analysed and classified so as to fetch anecdotes from the data.

4.2.2 Design and Implementation Constraints

1)Time for Delivery of the software. 2)Implementation based on research of the topic. 3)Overviewing the output accuracy.

4.3 Functional Requirements

i)Software should be able to detect the anecdotes and prepend/append those with the ¡anecdote¿ ¡/anecdote¿ tags. ii)Dataset should be uniform and in lines with the structure of data assumed. iii)System should throw error for any file handling exception.

4.4 Other Nonfunctional Requirements

4.4.1 Performance Requirements

The product will be based on web and has to be run from a web server. The loading initial time will depend on the internet connection strength and it's bandwidth which further depends on the computing environment from which program is being run. The hardware components of the project also influence it's performance however there arent any in thsi particular project.

4.4.2 Safety and Security Requirements

Our software being a research product doesnt consider security as a major concern as yet and hence there is just a generic single log in at the start and no other considerations. However, Future scope might include security provisions for all round robustness.

4.4.3 Software Quality Attributes

The website made has following features : adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness and testability.

Chapter 5

Methodology

5.1 What is automatic anecdote detection?

An anecdote is any independent plot or story inside another story based on real person or an incident. Detection of such anecdotes using Machine Learning and rule based approach from any given random text is the essence of automatic anecdote detection.

5.2 How did we approach the problem of Automatic Anedote Detection?

The overall algorithm of Anecdote Detection can be boiled down to the following points.

1. Scraping data set for testing and training the model from Excel spreadsheet and online books to text files using python scripts.

2. Ensemble Learning - Voting Classifier has been used as the model for anecdotal sentences prediction.

3. Then, using NER(Named Entity Recognition) with POS tagging plus the Voting Classifier to find the starting sentence of any anecdote.

4. Features for model training are tweaked to improve the accuracy i.e. important features like the following POS tags: NNP, VBG, VBD, POS, are given more weightage than other POS tags.
5. After finding the start, model is then used to predict and identify the body of the anecdote, thereby finding the end.
6. Heuristics are used in addition to the ML prediction to get the final outputs to smoothen the edgy outputs or exceptions due to corner cases.
7. Finally, we show the automatic anecdote detection through our web-app(UI) to show the anecdotes identified from a random text file.

Chapter 6

Models Used

6.1 ADABoost

AdaBoost, stands for Adaptive Boosting. This is a machine learning algorithm that was developed by Yoav Freund and Robert Schapire.The other leaning algorithm i.e. the weak learners are combined into a weighted sum that represent the final output of the ADAboost. AdaBoost is contextually adadaptive in the sense that the weak learners are rectified in the way that the instances misclassified by the previous classifier.

Noisy data and outliers do affect the ADAboost algorithm. It's still better than the other algorithms in the context of the overfitting problem.

The individual models that used for prediction can be weak, but till the time performance of each weak algorithm is slightly better than random guessing, the final model will turn out to become a strong learner.

6.2 Random Forest

Random Forest is a flexible, easy to use machine learning algorithm that produces, even without much parametric tuning, a great result most of the time. It is also one of the most used algorithms, because its simplicity and the fact that it can be used for both classification and regression tasks. Random

Forest is a supervised learning algorithm.

The ‘forest it builds, is an ensemble of Decision Trees, most of the time trained with the bagging method. The general idea of the bagging method is that a combination of learning models increases the overall result.

The random-forest algorithm brings extra randomness into the model, when it is growing the trees. Instead of searching for the best feature while splitting a node, it searches for the best feature among a random subset of features. This process creates a wide diversity, which generally results in a better model.

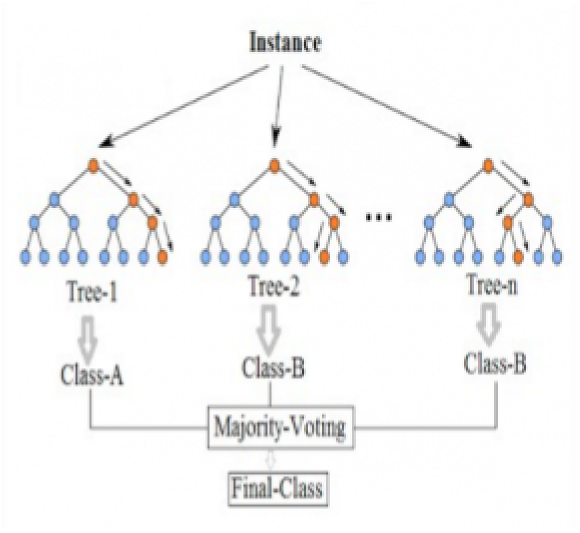


Figure 6.1: Random Forest

6.3 Naive Bayes

Naive Bayes is a machine learning algorithm that predicts the outcome of the input based on the likelihood, class prior probability and predictor prior probability of the dataset. This algorithm is based on the Bayes Theorem. It depends on the conditional probability of classes and the feature vector. The diagram below shows the evaluation of the probability of a feature vector belonging to a class.

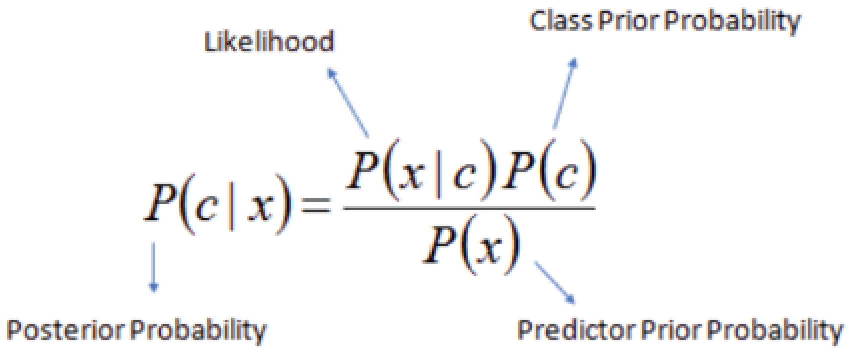


Figure 6.2: Naive Bayes

The details of the above diagram as given below:

- C is the class. It can be binary or multiclass.
- X is the input or feature vector.
- P(C—X) is the posterior probability. The probability of feature vector X belonging to a Class
- P(X—C) is the likelihood, is computed at the time of training.
- P(C) is the prior probability, is computed at the time of training.
- P(X) is the posterior prior probability, is computed at the time of training.

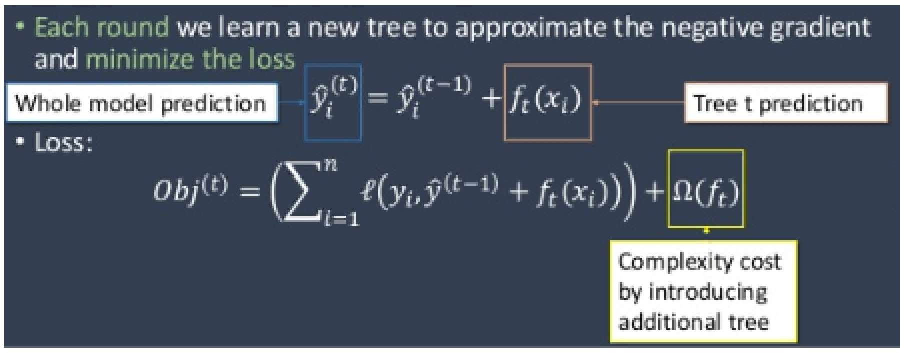
6.4 eXtreme Gradient Boost- XGBoost

XGBoost or Extreme Gradient Boosting is a type of gradient boosted decision trees which excels in the speed and performance when compared with other Gradient Boosting Algorithms. XGBoost excels in classification or regression modeling problems of structured or tabular datasets. It is a supervised machine learning algorithm.

XGBoost improves the prediction of the model by steadily improving ensemble of the weak/base learners. It builds the model in a step-by-step manner like other boosting methods do, and tries to generalize its model by optimizing an arbitrary loss function provided that it is differentiable.

Boosting is an ensemble technique where new models are added to lessen the error or loss due to the existing models. Models are added sequentially until the learner becomes good enough to maximally predict the output correctly.

Gradient Boosting is an approach where the new model trains such that they try to reduce the errors made by the previous models(prior to the new one) and then add together in such a way that maximizes the final prediction accuracy. Gradient descent algorithm is used to reduce and minimize the loss. This is why the algorithm is termed as the gradient boosting algorithm. Gradient Boosting can solve both classification and regression problems by the predictions.



- Each round we learn a new tree to approximate the negative gradient and minimize the loss

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i)$$

- Loss:

$$Obj^{(t)} = \left(\sum_{i=1}^{n} \ell\left(y_i, \hat{y}^{(t-1)} + f_t(x_i)\right)\right) + \Omega(f_t)$$

Figure 6.3: XGBoost

6.5 Ensemble : Voting Classifier

Ensemble Learning is a subset of Machine Learning which is predominantly used when several different models provide mediocre accuracy and the developer wishes to improve on it. Here, we use Voting Classifier to carry out ensemble learning on our model. Voting Classifier lets the developer use multiple classifiers/models with their own parameters. These constituent classifiers are called base learners and the generalization abilities of the Ensemble learner are much stronger than that of the base learners. Such base learners are also called weak learners. The most appealing part of using ensemble learning is its ability to boost the performance and accuracy of the model from that of the weak learners. Such models convert the performance of base learners which is only slightly better than random guessers to strong learners which can make very accurate predictions. Despite all this, most theoretical research or work is done on base learners since those weak learners which are used in practice are not necessarily weak since using not-so-weak base learners most often results in above-par performance. Also, Voting classifier lets the developer use weights so as to emphasize on the performance of particular base learners more than that of others. There are two types opf voting under voting classifier and we use the hard voting since not all the base learners that we use have the inbuilt ability to predict the class with the highest class probability(i.e. They have a $predict_probe$ method).

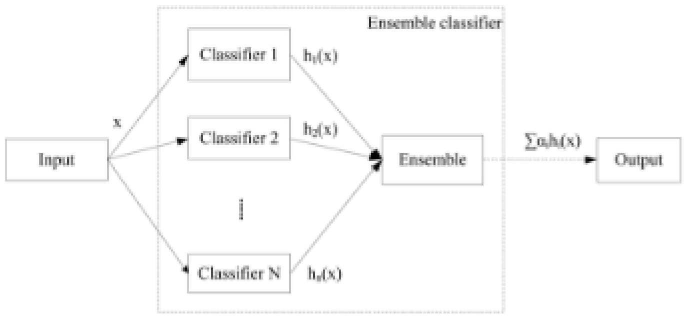


Figure 6.4: Voting Classifier

6.6 Logistic Regression

One of the predictive analyses, like all regression analyses, is Logistic Regression. It is a classification algorithm. It is the most appropriate regression analyses to conduct when the dependent variable can take only binary values i.e. if it is dichotomous. Logistic Regression is used to analyse relationships between a binary dependent variable with an ordinal/nominal/interval independent variable. It is a special case of linear regression. Through fitting data to a logistic function, it predicts the probability of occurrence of an event. For example, a type of question that a logistic classifier can answer is How does the probability of having a heart attack(yes or no) depend on factis like body weight, calorie intake, fat intake and age ADABoost AdaBoost, stands for Adaptive Boosting. This is a machine learning algorithm that was developed by Yoav Freund and Robert Schapire. The other leaning algorithm i.e. the weak learners are combined into a weighted sum that represent the final output of the ADAboost. AdaBoost is contextually adadaptive in the sense that the weak learners are rectified in the way that the instances misclassified by the previous classifier.

Noisy data and outliers do affect the ADAboost algorithm. It's still better than the other algorithms in the context of the overfitting problem.

The individual models that used for prediction can be weak, but till the time performance of each weak algorithm is slightly better than random guessing, the final model will turn out to become a strong learner.

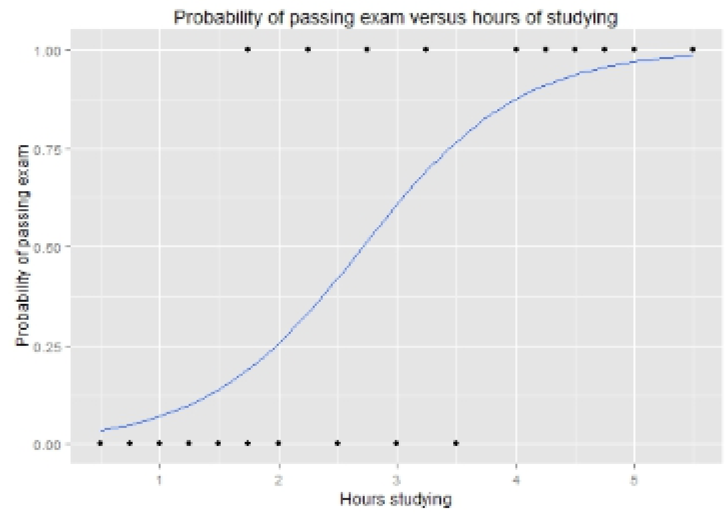


Figure 6.5: Logistic Regression

6.7 Support Vector Machine

One of the most commonly used supervised machine learning classifiers is SVM(Support Vector Machine) classifier. The essence of the algorithm behind the classifier is that given a set of labelled training data, it outputs a suitable/optimal hyperplane which categorizes the examples/data. The nearest neighboring data points should be farthest from the hyperplane for each feature-dimension. This makes the algorithm more optimal.

$$K(\mathbf{x}, \mathbf{x'}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x'}\|^2}{2\sigma^2}\right)$$

Figure 6.6: Kernel RBF Function

SVM uses kernel trick to classify non linear data. Kernels like Radial Basis Function(popularly known as RBF) are used to map the inputs to a higher dimension to make the data points linearly separable since higher the dimension, the greater is the probability of finding a hyperplane that separates the dataset into their corresponding categories. x and x represented as feature vectors in an input space, the RBF kernel on x and x is defined as

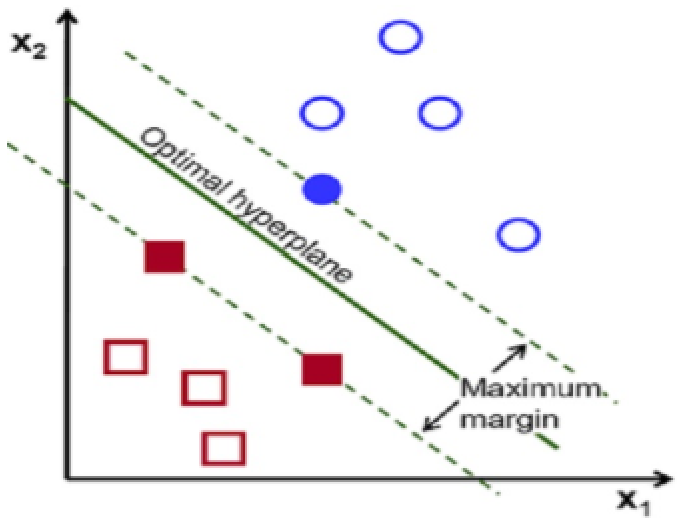


Figure 6.7: SVM classification

Support Vector Machines try to optimize the separating hyperplane such that it divides the data points to their categories in the best possible way. SVMs do this by maximizing the distance between the two margins. A margin is the perpendicular distance between the nearest data points on either side of the hyperplane. The nearest data points on either side of the hyperplane are called support vectors.In the above figure the points with their inner color filled are the support vectors for the given hyperplane.

Chapter 7

Implementation

7.1 Scraping the Dataset

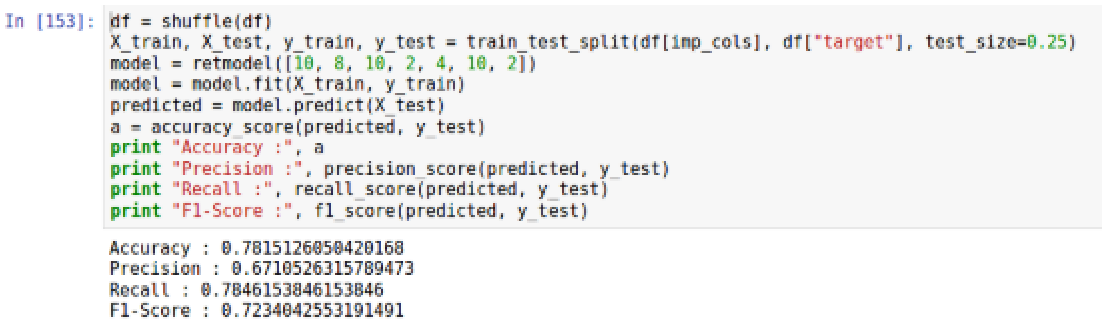
Scraping from excel to .txt file using python scripts. Excel datasheet provides the Blinkist link for the books which have been used to create the dataset and the next column shows anecdotes present in that particular book. Using the link and the login credentials of Blinkist provided to us by chosetothinq.com, we scrape the online content and download the book as a .txt file. Using BeautifulSoup. Once the books are avaiulabnle in the text format, we can tokenize them one by one into sentences to create the test and train data. Tokenization of the whole text into sentences is important since we train and test the model to check whether the given sentence can be a part of an anecdote or not.

7.2 Identifying the start of anecdote

Next, we use the Named Entity Recognition(NER) library from Stanford to detect proper nouns since anecdotes are independent plots of stories based on a real person/incident. According to the research we have done, more than 80% of the anecdotes have a proper noun at the start of the anecdote i.e. as a part of the first sentence of the anecdote. Hence, we have written a rule based algorithm to check for such proper nouns using the POS(Part-of-speech) tagging provided by nltk.

7.3 Machine Learning Model

About the ML model for the project, we have tried different models, however given the amount of data, our finalised model gives 75-85% accuracy depending on the shuffling of the training data. We have decided to use the concept of Ensemble Learning under which we have used Voting Classifier which incorporates the pros of all the classifiers that we use which are Naive Bayes, Random Forest, Extra Tree, SVM, Adaboost, XGBoost, Logistic Regression. Voting Classifier works according to the weights we as developers provide it with in reference to the classifiers mentioned above. It gives a cumulative Accuracy score of 75-85% for a set of weights as shown in the screenshot below.



```
In [153]: df = shuffle(df)
          X_train, X_test, y_train, y_test = train_test_split(df[imp_cols], df["target"], test_size=0.25)
          model = retmodel([10, 8, 10, 2, 4, 10, 2])
          model = model.fit(X_train, y_train)
          predicted = model.predict(X_test)
          a = accuracy_score(predicted, y_test)
          print "Accuracy :", a
          print "Precision :", precision_score(predicted, y_test)
          print "Recall :", recall_score(predicted, y_test)
          print "F1-Score :", f1_score(predicted, y_test)

          Accuracy : 0.7815126050420168
          Precision : 0.6710526315789473
          Recall : 0.7846153846153846
          F1-Score : 0.7234042553191491
```

Figure 7.1: Attributes of Emsemble Model

7.4 Identifying the start

Now, the first step in actually finding out the anecdote is identifying the start of the it which as mentioned above is done using Named Entity Recognition and POS tagging provided by nltk. We use the output from this rule based algorithm and refer the models prediction for each sentence in order to find the start of the anecdote. If it gets a green light from both the algorithms, then it is classified as the start of an anecdote. In addition to the NER and machine learning model, we have hard-coded detection of a few sample phrases like Long time ago, For example, For instance, etc. in order to detect the start of the anecdote, thereby helping us increase the accuracy of our overall algorithm.

7.5 Identifying the body and end of the anecdote

Next part is detection of the body of the anecdote in order to find its end. This is done using the Ensemble Learning model and a few heuristics used for cleaner output. Each sentence is feed to the model in order to get the models prediction if it can be a part of the anecdote or not. Furthermore, Heuristics are added to smoothen the rough edges around the output prediction of sentences. For example, for a given set of 6 sentences the output given by our model is 111011, then the actual probability of the 4th sentence to be an anecdote is high since both the sentences around it are anecdotes.The chances of the 4th sentence being an anecdote are lower when the output is 110011 since its surrounded by one non anecdotal sentence and one probable anecdotal sentence. Similarly, for 11000000011, it's almost impossible for the 6th sentence to be an anecdote. Thus this exponentially increasing probability of a sentence to be anecdotal depends on its neighbours too. Our heuristics take that into consideration and tweak the output by the model to get the final output to be displayed on the web-app. Also, to find the end of an anecdote, for the output 111000, 4th sentence is obviously the end of the anecdote which starts at the first sentence since the 4th sentence output is followed by all 0s i.e non anecdotal sentences.

7.6 User Interface

We have a User interface in place as the front end which starts with authentication(Login) page wherein only users with a set of credentials are allowed to access the Automatic Anecdote Detection portal. The GUI in the project is build using Django. Django has a very modular code. Each component that we build on Django can be reused. Also an important feature of Django is that it provides binding between inputs and outputs. Here are a few screenshots of the web application used for Automatic Anecdote Detection.

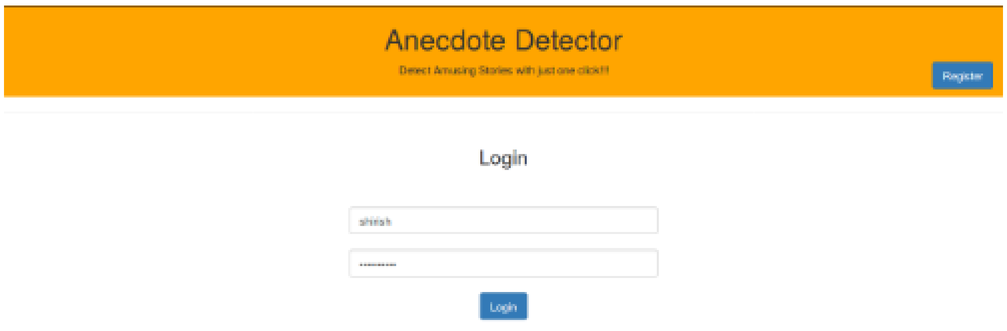
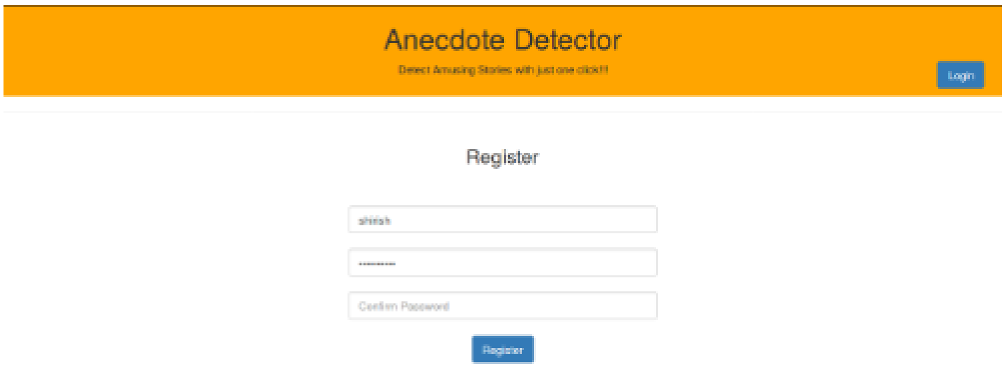


Figure 7.2: Login Screen



Anecdote Detector

Detect Amusing Stories with just one click!!!

Login

Register

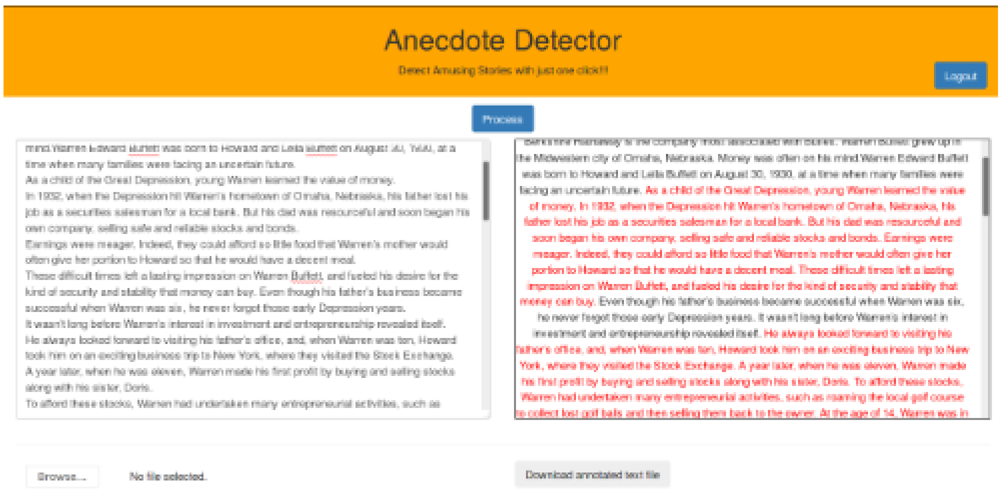
shirish

••••••••••

Confirm Password

Register

Figure 7.3: Registration Screen



Anecdote Detector

Detect Amusing Stories with just one click!!!

Logout

Process

mind.Warren Edward Buffett was born to Howard and Leila Buffett on August 30, 1930, at a
time when many families were facing an uncertain future.
As a child of the Great Depression, young Warren learned the value of money.
In 1932, when the Depression hit Warren's hometown of Omaha, Nebraska, his father lost his
job as a securities salesman for a local bank. But his dad was resourceful and soon began his
own company, selling safe and reliable stocks and bonds.
Earnings were meager. Indeed, they could afford so little food that Warren's mother would
often give her portion to Howard so that he would have a decent meal.
These difficult times left a lasting impression on Warren Buffett, and fueled his desire for the
kind of security and stability that money can buy. Even though his father's business became
successful when Warren was six, he never forgot those early Depression years.
It wasn't long before Warren's interest in investment and entrepreneurship revealed itself.
He always looked forward to visiting his father's office, and, when Warren was ten, Howard
took him on an exciting business trip to New York, where they visited the Stock Exchange.
A year later, when he was eleven, Warren made his first profit by buying and selling stocks
along with his sister, Doris.
To afford these stocks, Warren had undertaken many entrepreneurial activities, such as

Berkshire Hathaway is the company most associated with Buffett. Warren Buffett grew up in
the Midwestern city of Omaha, Nebraska. Money was often on his mind.Warren Edward Buffett
was born to Howard and Leila Buffett on August 30, 1930, at a time when many families were
facing an uncertain future. As a child of the Great Depression, young Warren learned the value
of money. In 1932, when the Depression hit Warren's hometown of Omaha, Nebraska, his
father lost his job as a securities salesman for a local bank. But his dad was resourceful and
soon began his own company, selling safe and reliable stocks and bonds. Earnings were
meager. Indeed, they could afford so little food that Warren's mother would often give her
portion to Howard so that he would have a decent meal. These difficult times left a lasting
impression on Warren Buffett, and fueled his desire for the kind of security and stability that
money can buy. Even though his father's business became successful when Warren was six,
he never forgot those early Depression years. It wasn't long before Warren's interest in
investment and entrepreneurship revealed itself. He always looked forward to visiting his
father's office, and, when Warren was ten, Howard took him on an exciting business trip to New
York, where they visited the Stock Exchange. A year later, when he was eleven, Warren made
his first profit by buying and selling stocks along with his sister, Doris. To afford these stocks,
Warren had undertaken many entrepreneurial activities, such as roaming the local golf course
to collect lost golf balls and then selling them back to the owner. At the age of 14, Warren was in

Browse... No file selected.

Download annotated text file

Figure 7.4: Home Screen

Chapter 8

Result & Analysis

The aim of the project is to print the different anecdotes that appear in the given text. The input to the algorithm is a text file that the user inputs through the website and gets the list of anecdotes as the result displayed on the website. The accuracy of the machine learning model is around 78% ranging from 75% to 85% with precision 67%, recall 78% and f1-score as 72%. The model uses a combination of statistical and rule based approach in finding the anecdotes from the given text file. The model successfully determines the anecdotes from the text file with a standard and intuitive way of interactive experience on the website.

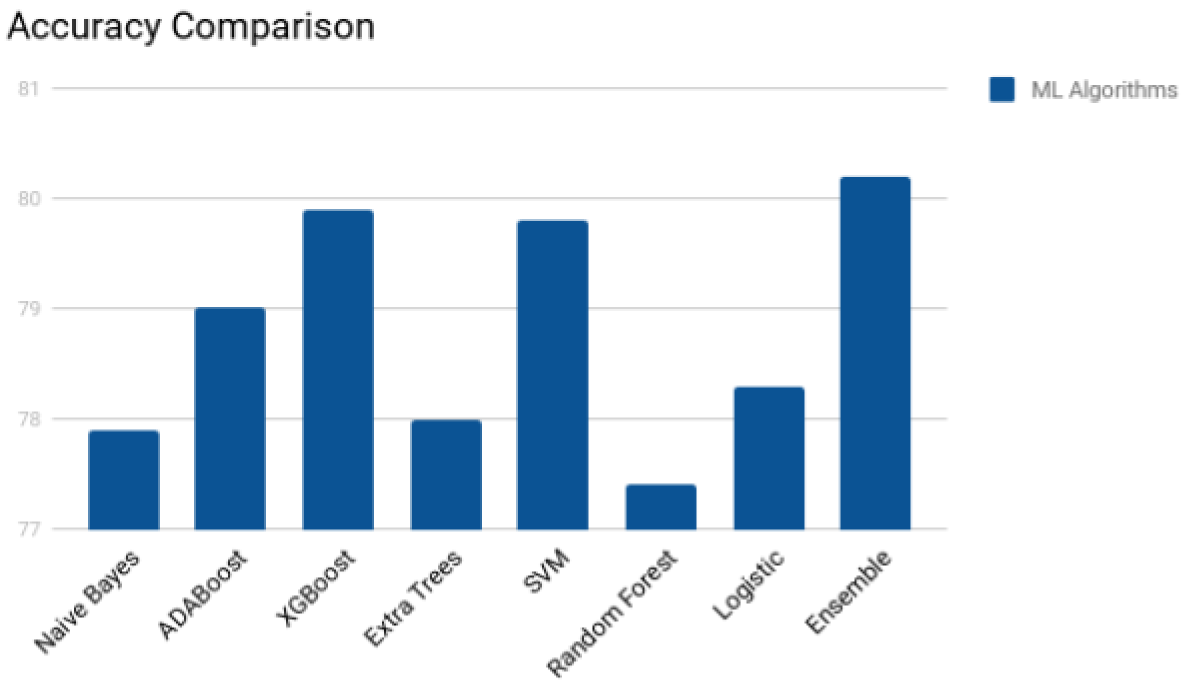


Figure 8.1: Accuracy Comparison

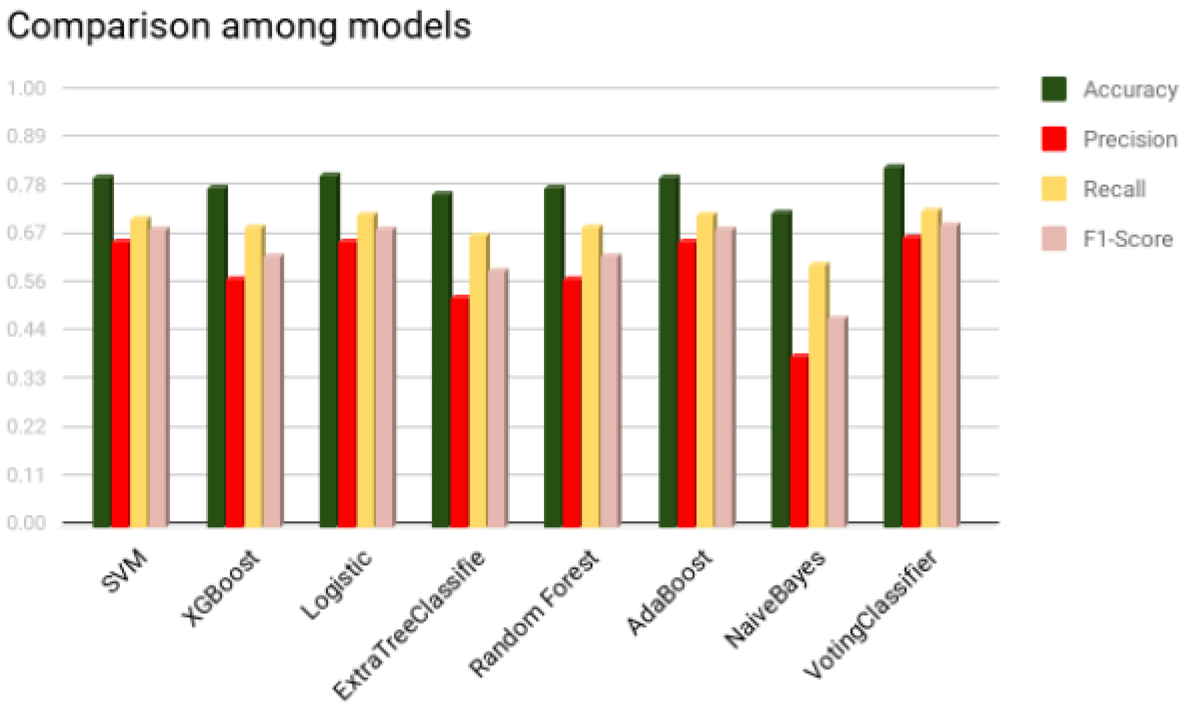


Figure 8.2: Overall Comparison

Chapter 9

Conclusion

Anecdote Detector is a research project. We have used number of models and designed an algorithm to make the computer do the task of anecdote detection. Understanding the approach taken by the project to solve the problem of anecdote detections will create a subdomain for the engineers, computer scientists and computer enthusiasts in the field of Natural Language Processing and Machine Learning. Non-technical or lay person will realize the potential of NLP and ML after going through this project. With the current project the manual process of analyzing text and detecting these anecdotes can be automated. This saves a whole lot of time of doing the monotonous task. Automatic Anecdote Detection will provide us with equivalent, if not better, efficiency and accuracy in finding the anecdotes.

9.1 Product Scope

Automatic Anecdote Detection is going to be used at Choose to Thinq. We are working along with them so that the manual tedious task of the anecdote detection get automated. 'Choose to Thinq' is a consulting company which inspires people to increase the productivity, make the workplace free of stress etc by giving real life examples from the self help books. Our work of Anecdote Detection will help the company 'Choose To Thinq' to get its task done by minimizing the human error. This project will save the companys finances and human efforts.

9.2 Future Scope

Automatic Anecdote Detector being a project first of its kind can be used and modified to find quotes, humour etc. Our algorithm can be tweaked to find customizable snippets from the unstructured text given as an input. The security can be customized according to the needs of the administrator and a single log in screen could be changed accordingly. The target and feature sets can also be changed in the future for varying needs such as to identify poetry, idioms, quotes, etc. from a given file of random text or story. Also, currently the application allows the users to upload only text files(.txt) as input. A major future scope for the project would be to include inputs from other file types like PDF, DOC, etc.

Bibliography

[1] Steven Bird, Ewan Klein Edward Lopar. *Natural Language Processing with Python.*

[2] Igor A. Bolshakov and Alexander Gelbukh, *Text Segmentation into Paragraphs Based on Local Text Cohesion*

[3] Thanh Ngoc Dao (thanh.dao@gmx.net) and Troy Simpson (troy@ebswift.com), *Measuring Similarity between sentences*

[4] Thomas G. Dietterich, *Ensemble Methods in Machine Learning*

[5] Ludmila I. Kuncheva and Juan J. Rodrguez, *A weighted voting framework for classifiers ensembles*

[6] R. M. Valdovinos and J. S. Snchez, *Combining Multiple Classifiers with Dynamic Weighted Voting*

[7] Logistic Regression
`https://www.datacamp.com/community/tutorials/logistic-regression-R`.

[8] L. Torlay, M. Perrone-Bertolotti, E. Thomas & M. Baciu *Machine learningXGBoost analysis of language networks to classify patients with epilepsy*

[9] Naive Bayes
`https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/`

[10] Support Vector Machines
`https://en.wikipedia.org/wiki/Support`$_vector_machine.$