**Naïve Bayes – Algorithm**

**Training phase:**

Of all the text which are present in the training data, the probability of ‘positive’ (P0-percentage of all text which are positive)and ‘negative’(1-P0-percentage of all text which are negative). This is helpful because if you choose a comment randomly it will tell you what is the probability of that to be positive or negative.

Ex - Positive : 55%; Negative: 45%

Ex – The word “Happy”

PosHappy = Sum frequency of “happy” in positive comments

Sum frequency of “Happy” in entire corpus(Training dataset)

NegHappy = 1 - PosHappy

We will calculate Pos and Neg score for every word inside the training data and at the end of it we will have 2 info

* The probability that any randomly picked comment is Positive or a negative comment
* The Pos score and Neg score for every word present in the training data.

**Test Phase:**

|  |  |
| --- | --- |
|  | **Pos context** |
| **Happy** | **90%** |
| **Love** | **95%** |
| **Food** | **50%** |
| **Hate** | **10%** |
| **Bad** | **30%** |
|  |  |

**Ex: “Love the food”**

Pos of sentence = PosLove \* Posfood \* P0( Overall prob that the comment is positive)

In case like this we ignore words like the, a, an etc

= .95 \* .50 \* .55 = 0.26

Negativity score = (1 – PosLove ) \* ( 1 – Posfood ) \* ( 1 – P0 )

= .05 \* .5 \* .45 = 0.01

Pos score > Neg socre

This sentence is Positive

**Why it is called Naïve?**

PosComment = PosWord1 \* PosWord2 \* PosWord3 …..

Each words contributes independently

Because no term accounting for a two or more words appearing together ( Like if a phrase comes with the same context in same sentence the phrase does not at all affect the positivity or negativity of a sentence)

A false assumption – Naïve Bayes algorithm is not that powerful.

This is not true. Naïve Bayes is actually very Robust. Specially in the case when you have small amount of training data or you don’t have lot of info of domain knowledge to create very relevant feature.

**Actual Example:**

Dataset – Sentiment Labelled Sentences Dataset

The dataset has sentences from 3 sources

IMDB reviews, Yelp reviews, Amazon reviews

Each line in dataset

**review label**

Wasted two hours 0

0 is for negative sentences and 1 is for positive sentences

We will use CountVectorizer to convert sentence into frequency of each word in that senetence.

**Support Vector Machine Algorithm**

**Ad Detection**

**Features:**

We will use attributes of image such as the height or weight of the image, page url, Image url, Page text, image text, Image Caption text

**Training Phase:**

A large dataset of images which is already labelled as Ad/Non Ad

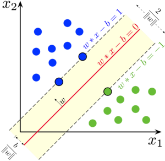
Represent all the images as points in an N-Dimensional Hypercube

**N-Dimensional Hypercube**

* A line is a 1-D shape -> Any point on a line can be represented using 1 number
* A square is a 2-D shape -> Any point in square can be represented using 2 numbers (x, y)
* A cube is a 3-D shape -> Any point in cube can be represented using 3 numbers (x, y, z)
* N-Dimensional Hypercube -> A set of N numbers represents a point in an N-Dimensional Hypercube

(2, 0, 0, 0, 0, 1, 0, 1, 0…….)

Tuples of N numbers



SVM finds a hyperplane that nearly separates the 2 sets of points

On one side of boundary you will only have Ad images and on the other hand of boundary you will have Non Ad images.

A hyperplane is just a surface that acts as a boundary in an N-Dimensional space.

**Test Phase:**

We get a new image, we will just check which side of the boundary this image falls on.

**Note:**

SVM can only be used for binary classification.

**Implementation:**

Internet Advertisements Data set