**Syracuse University**

**IST-736 Assignment 4**

ThulasiRam RuppaKrishnan

IST 736

Professor Ami Gates

Contents

[Introduction 3](#_Toc15256938)

[Analysis and Models 4](#_Toc15256939)

[**About the data** 4](#_Toc15256940)

[**Models** 8](#_Toc15256941)

[Results 17](#_Toc15256942)

[Conclusion 24](#_Toc15256943)

## 

## **Introduction**

Artificial Intelligence, Machine Learning and Deep Learning (AI | ML | DL) are at the heart of digital transformation by enabling organizations to exploit their growing wealth of big data to optimize key business and operational use cases.

* **AI** (Artificial Intelligence) is the theory and development of computer systems able to perform tasks normally requiring human intelligence (e.g. visual perception, speech recognition, **Image recognition**, translation between languages, etc.).
* **ML** (Machine Learning) is a sub-field of AI that provides systems the ability to learn and improve by itself from experience without being explicitly programmed.
* **DL** (Deep Learning) is a type of ML built on a deep hierarchy of layers, with each layer solving different pieces of a complex problem. These layers are interconnected into a “neural network.” A DL framework is SW that accelerates the development and deployment of these models.

**Deception Recognition**

Deception is a socially pervasive psycholinguistic phenomenon from lies during legal trials to fabricated online product reviews. Its detection in human communication has long been of great interest in real-life situations involving law enforcement, national security, and business, etc. The techniques employed for the detection of deception are varied, ingenious, and often dramatic from the ancient Chinese method of spitting dry rice to the modern polygraph. Deception detection has also been the subject of investigation within psychology, social science, and linguistics, where it has mainly been based on qualitative and quantitative observations of gesture, facial expression and voice analysis. Nonetheless, very little scientific work has been done to date on the fundamental theoretical underpinnings of systems for automatically detecting deception in text. The proliferation in recent years of fake online reviews meant to deceive consumers has heightened the interest in automatic deception filtering systems.

Source: <https://core.ac.uk/download/pdf/54849184.pdf>



Deception is potentially disruptive in everyday communication, information seeking, and decision making. It is a message knowingly and intentionally transmitted by a sender to foster a false belief or conclusion by the perceiver. it is widespread phenomenon and often undetected especially in electronic environments where credibility assessments are complicated by the absence of many traditional cues such as verifiable credentials or face-to-face contact. The need arises for decision support tools capable of alerting users to potentially deceptive content.

Human Judgments and Automated Deception Detection In interpersonal psychology and communication studies, human respondents are often asked to distinguish deceptive statements from truthful ones. People are notoriously unreliable in this task. In a meta-analytical review of over 100 experiments with over 1,000 participants, DePaulo, and colleagues (1997) determined an unimpressive mean accuracy of 54%, slightly above chance. Recently attainable with natural language processing (NLP) and machine learning, automated approaches show promise in distinguishing a general sense of deception with success rates slightly higher than those of humans. The underlying mechanism is to identify reliable verbal cues which show linguistic differences when people deceive as compared to when they tell the truth. For instance, 3 out of 18 verbal cues derived from Statement Validity Analysis techniques (amount of detail reported, coherence, and admissions of lack of memory) used in law enforcement for credibility assessments tested as statistically reliable (Porter & Yuille,1996). Other studies report that, compared to truth-tellers, deceivers produce more total words and sense-based words show lower cognitive complexity, use more negative emotion words, use more extreme positive emotions and more general knowledge references, but use fewer certainty and hesitation words. When implemented in decision support tools, three standard classification algorithms (neural nets, decision trees, and logistic regression) achieved 74% accuracy (Fuller, Biros, & Wilson, 2009). Another approach adapted a pre-existing psycholinguistic lexicon (Linguistic Inquiry Word Count (LIWC)) and achieved an average 70% classifier accuracy in binary lie-truth text categorization (Mihalcea & Strapparava, 2009). With more evidence for reliable verbal cue combinations, human abilities to spot deception can be complemented, if not enhanced.

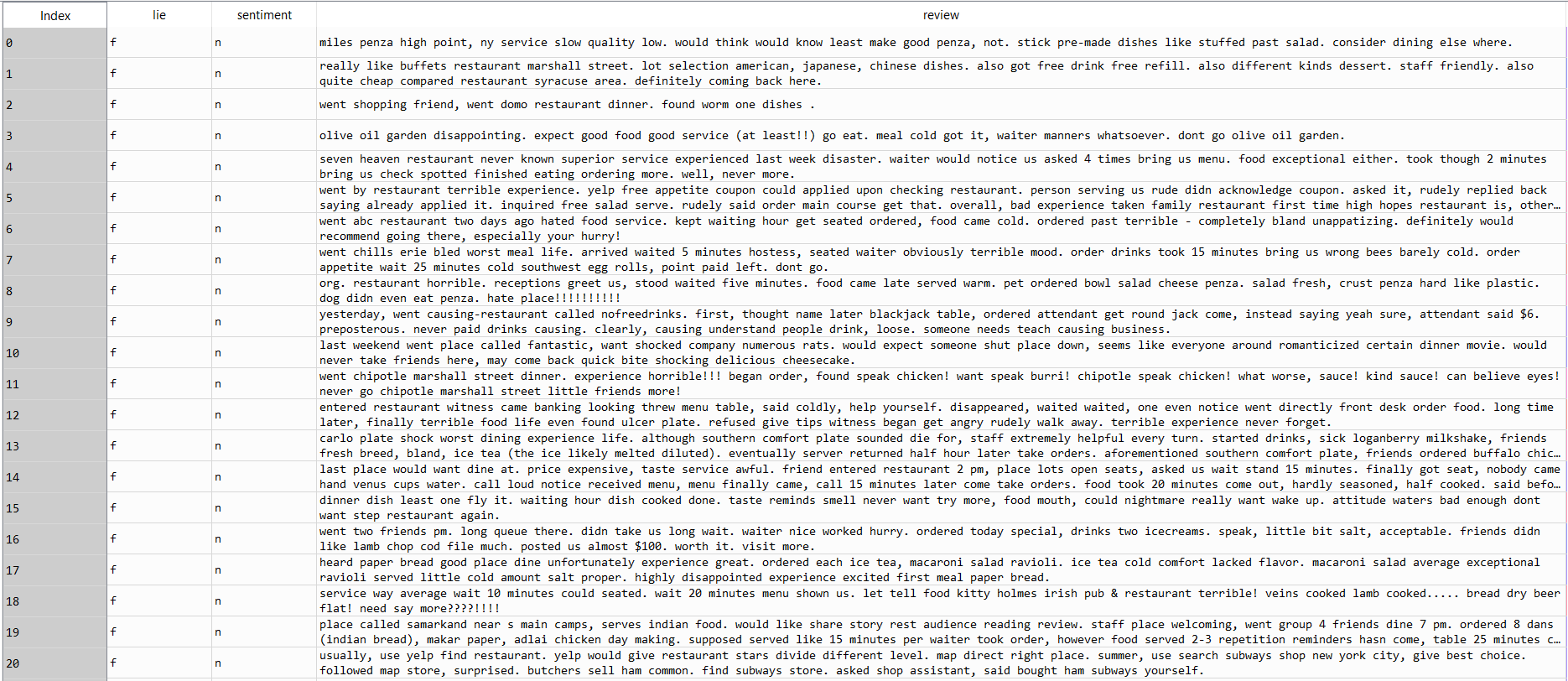
Source: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/meet.2011.14504801098>

## **Analysis and Models**

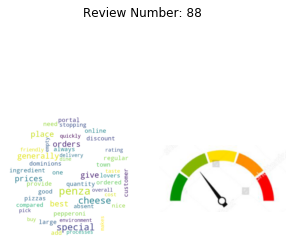
### **About the data**

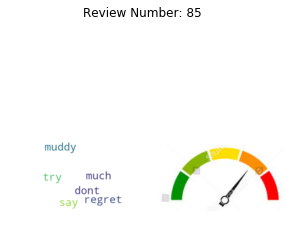
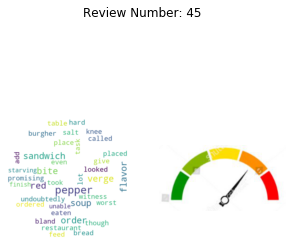
The dataset for deception analysis consists of 90 observations and 3 columns out of which one is labeled for fake/truth and the 2nd one is labeled for sentiment (positive/negative). The third variable(column) contains actual review submitted by public for a restaurant`. **Table 1.1** represents the sample dataset of reviews and its labels.

In the exploratory data analysis, word clouds are generated for every review and the positivity and negativity is tagged by using Sentiment Intensity analyzer. **Figure 1.1** shows word cloud and tagged sentiment intensity by each review. Also, **Figure 1.2 and 1.3** shows word count distribution for each review, and the labels are differentiated by color. Both fake and negative reviews are tagged in red and positive and true reviews are tagged in blue

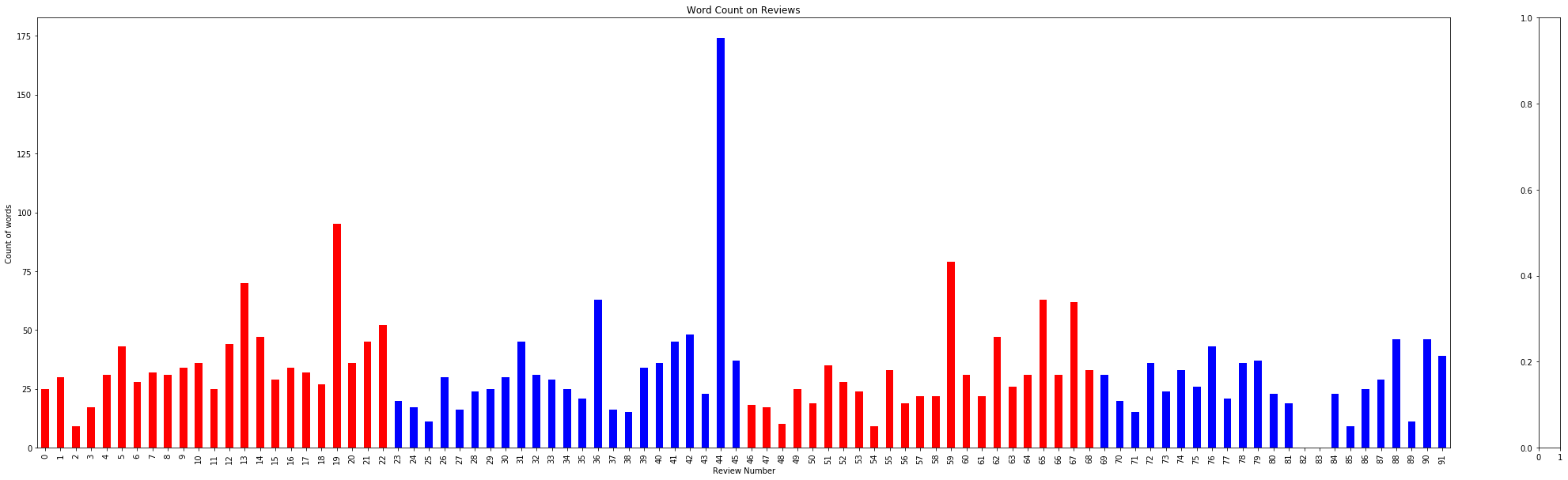


**Table 1.1 Input Data**

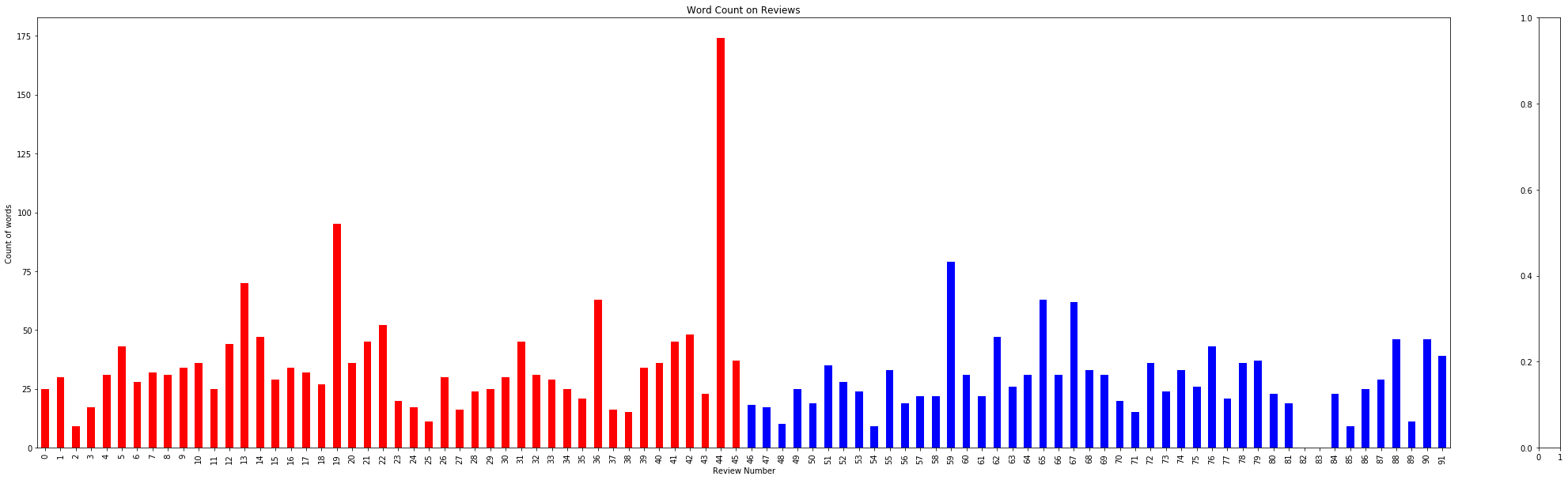
 

**Figure 1.1 Word Clouds**



**Figure 1.2 Word count distribution for fake and true reviews**



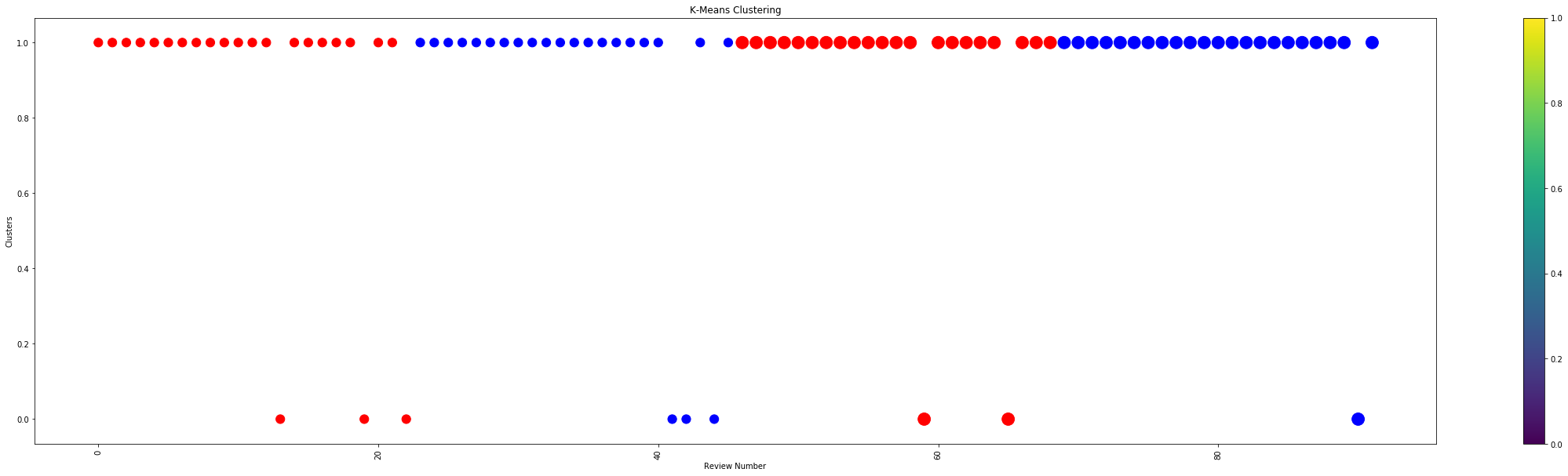
**Figure 1.3 Word count distribution for positive and negative reviews**

**K-means clusters**

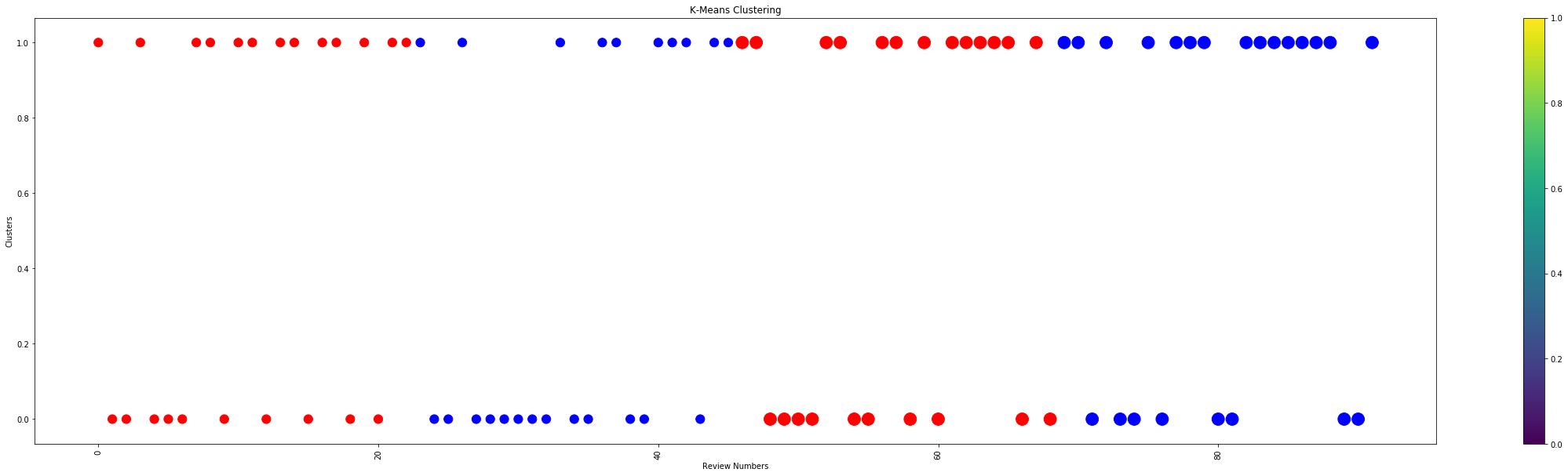
Clustering is performed after vectorization and the observation are shown in the below figures.

1. **Figure 1.4** represents the clusters formed by using word frequency count without normalization
2. **Figure 1.5** represents the clusters formed by using term frequencies
3. **Figure 1.6** represents the clusters formed by using TFIDF

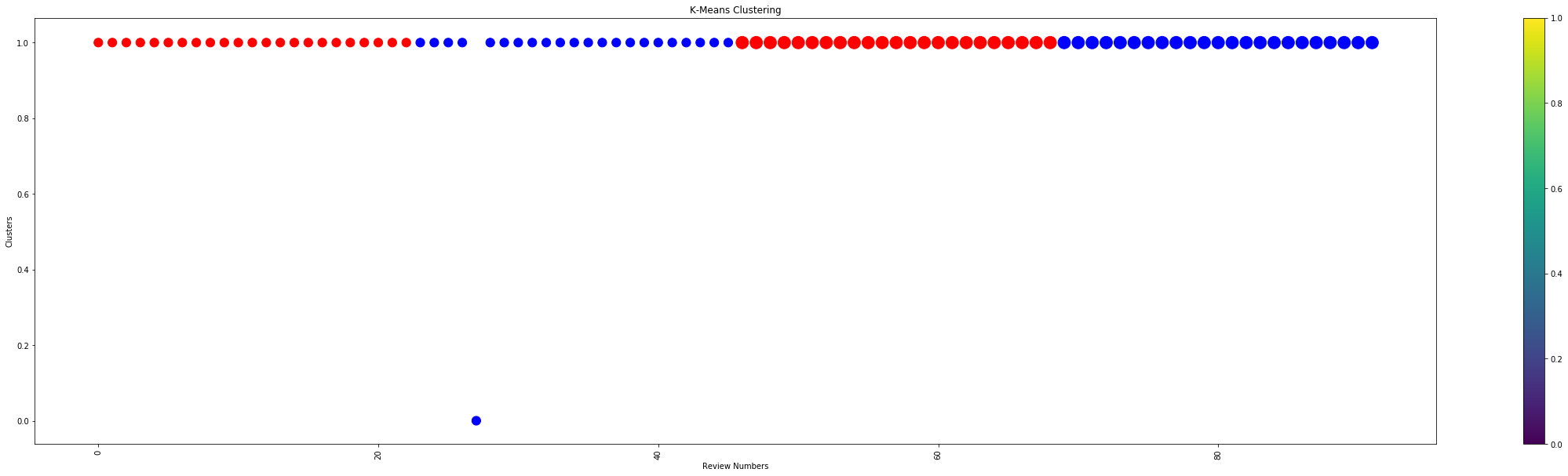
Clusters formed by using frequency count without normalization and TFIDF weightage doesn’t show healthy clusters and the term frequency clusters formed clusters with almost equal number of observations



**Figure 1.4 represents the clusters formed by using word frequency count**



**Figure 1.5 represents the clusters formed by using term frequencies**



**Figure 1.6 represents the clusters formed by using TFIDF**

### **Models**

In this exercise, models are developed using Naïve Bayes, SVM and Kmeans Clustering to compare their efficiency and accuracy in classifying deception and sentiments on a text document.

#### **Naïve Bayes Classification**

It is a classification technique based on Bayes’ Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a specific feature in a class is unrelated to the presence of any other feature. For example, a fruit may be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as ‘Naive’.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

**Bayes theorem**

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:



* P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes).
* P(c) is the prior probability of class.
* P(x|c) is the likelihood which is the probability of predictor given class.
* P(x) is the prior probability of predictor.

**Classification based on conditional probability**

To classify whether players will play or not based on weather condition using Naïve Bayes classification approach

Likelihood table Frequency Table are derived by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.

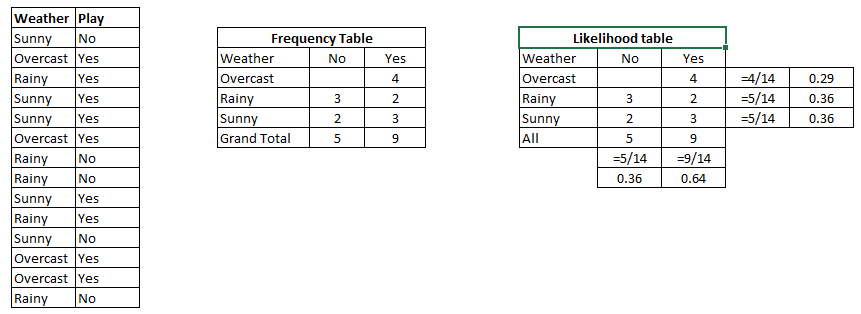
[](https://www.analyticsvidhya.com/wp-content/uploads/2015/08/Bayes_41.png)

Table 2.1

Using Naive Bayesian equation, the posterior probability for each class is calculated. The class with the highest posterior probability is the outcome of prediction.

Say if we want to find out if the Players will play when the weather is sunny?

To solve the above discussed method of posterior probability.

P (Yes | Sunny) = P (Sunny | Yes) \* P(Yes) / P (Sunny)

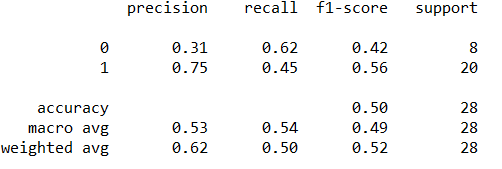
Here we have P (Sunny |Yes) = 3/9 = 0.33, P(Sunny) = 5/14 = 0.36, P(Yes)= 9/14 = 0.64

Now, P (Yes | Sunny) = 0.33 \* 0.64 / 0.36 = 0.60, which has higher probability.

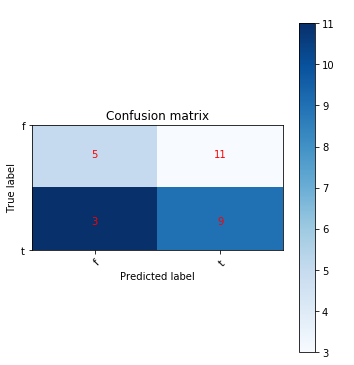
Naive Bayes uses a similar method to predict the probability of different class based on various attributes.

**Model 1.1: Lie Detection using Naïve Bayes (MultinomialNB)**

**Figure 2.1** shows the confusion matrix and the classification report of the Naïve Bayes MultinomialNB algorithm.



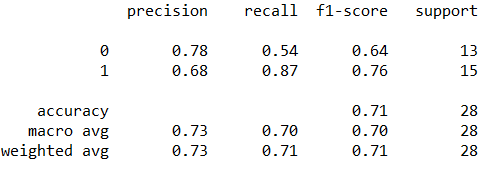
**Classification report**



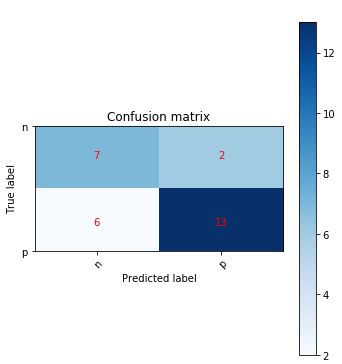
**Figure 2.1 Confusion Matrix for Naïve Bayes Model 1.1**

**Model 1.2: Sentiment Analysis using Naïve Bayes (MultinomialNB)**

**Figure 2.2** shows the confusion matrix and the classification\_report using Naïve Bayes (MultinomialNB) algorithm.



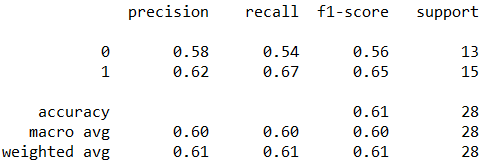
**Classification report**



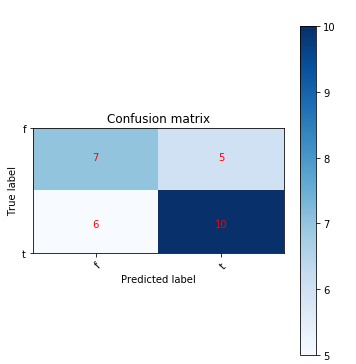
**Figure 2.2 Confusion Matrix for Naïve Bayes Model 1.2**

**Model 1.3: Lie Detection using Naïve Bayes (BernoulliNB)**

**Figure 2.3** shows the confusion matrix and the classification\_report of the Naïve Bayes BernoulliNB algorithm.



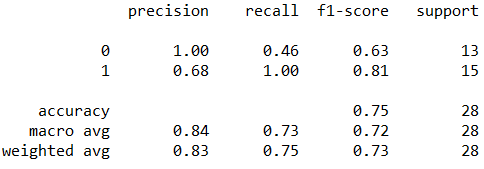
**Classification report**



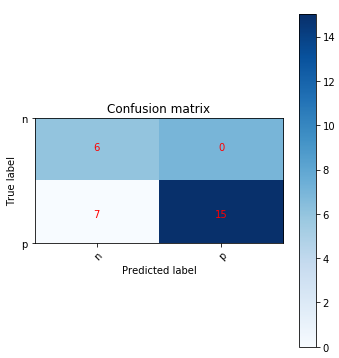
**Figure 2.3 Confusion Matrix for Naïve Bayes Model 1.3**

**Model 1.4: Sentiment Analysis using Naïve Bayes (BernoulliNB)**

**Figure 2.4** shows the confusion matrix and the classification\_report using Naïve Bayes (BernoulliNB) algorithm.



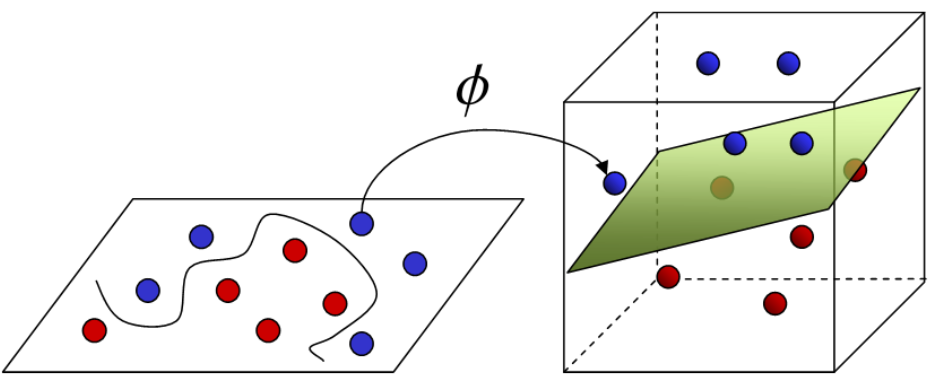
**Classification report**



**Figure 2.4 Confusion Matrix for Naïve Bayes Model 1.4**

#### **SVM (Support Vector Machine)**

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In two-dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.



**Kernel**

The learning of the hyperplane in linear SVM is done by transforming the problem using some linear algebra. This is where the kernel plays role. For linear kernel the equation for prediction for a new input using the dot product between the input (x) and each support vector (xi) is calculated as follows:

f(x) = B (0) + sum (ai \* (x, xi))

This is an equation that involves calculating the inner products of a new input vector (x) with all support vectors in training data. The coefficients B0 and ai (for each input) must be estimated from the training data by the learning algorithm. The polynomial kernel can be written as K(x,xi) = 1 + sum(x \* xi)^d and exponential as K(x,xi) = exp(-gamma \* sum((x — xi²)).

Polynomial and exponential kernels calculate separation line in higher dimension. This is called kernel trick

**Regularization**

The Regularization parameter (often termed as C parameter in python’s sklearn library) tells the SVM optimization how much you want to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points.

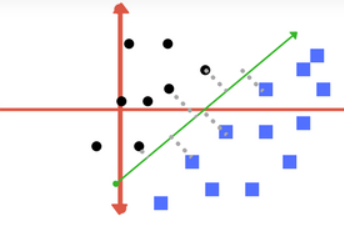
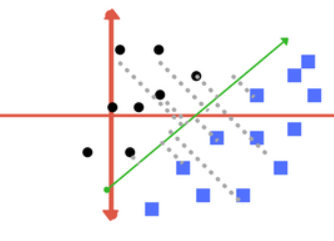
The images below (same as image 1 and image 2 in section 2) are example of two different regularization parameter. Left one has some misclassification due to lower regularization value. Higher value leads to results like right one.



**Left: low regularization value, right: high regularization value**

**Gamma**

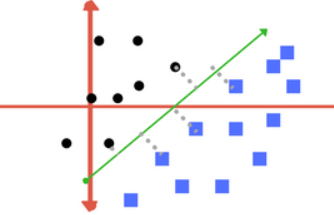
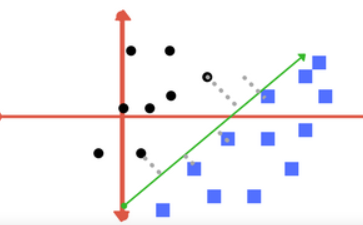
The gamma parameter defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’. In other words, with low gamma, points far away from plausible separation line are considered in calculation for the separation line. Whereas high gamma means the points close to plausible line are considered in calculation.

**High Gamma Low Gamma**

**Margin**

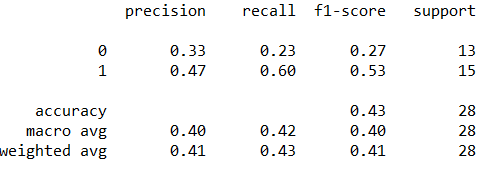
And finally, last but very important characteristic of SVM classifier. SVM to core tries to achieve a good margin. A margin is a separation of line to the closest class points. A good margin is one where this separation is larger for both the classes. Images below gives to visual example of good and bad margin. A good margin allows the points to be in their respective classes without crossing to other class.

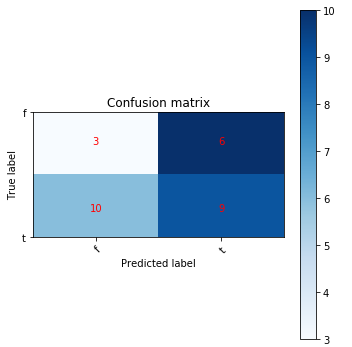
**Good Margin Bad Margin**

**Model 1.5: Lie Detection using SVM Linear algorithm**

**Figure 2.5** shows the confusion matrix and the classification report using SVM linear algorithm



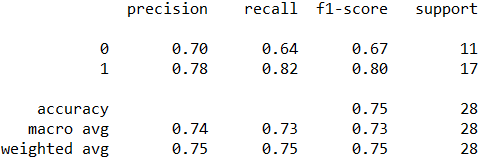
**Classification Report**



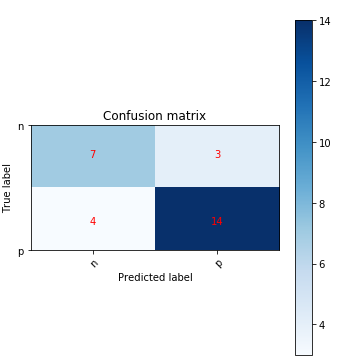
**Figure 2.5 Confusion Matrix for SVM-linear Model 1.5**

**Model 1.6: Sentiment Analysis using SVM linear algorithm**

**Figure 2.6** shows the confusion matrix and the classification report using SVM linear algorithm.



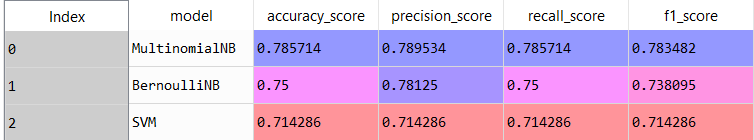
**Classification Report**



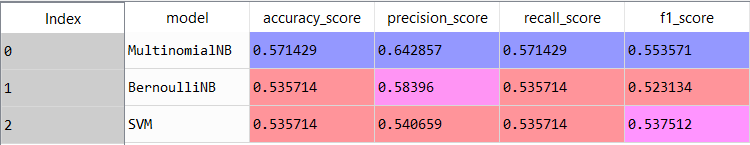
**Figure 2.6 Confusion Matrix for SVM-linear Model 1.6**

## **Results**

Accuracy, precision and recall using Naive Bayes and SVM are tabulated in **Table 3.1 and Table 3.2** for sentiment and Lie detection respectively



**Table 3.2 Model performance comparison for sentiment analysis**

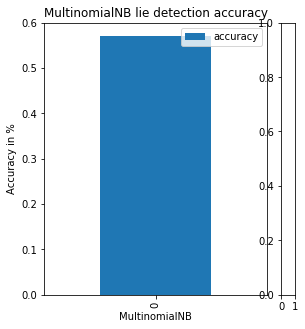


**Table 3.2 Model performance comparison for lie detection**

#### **Naïve Bayes Classification**

**Model 1.1: Lie Detection using Naïve Bayes MultinomialNB**

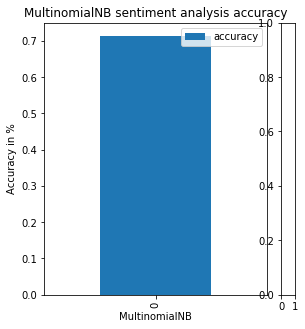
The accuracy of this model for predicting fake and true reviews is about 57% as show in **Figure 3.1**



**Figure 3.1 Accuracy of the Model 1.1**

**Model 1.2: Sentiment Analysis using Naïve Bayes MultinomialNB**

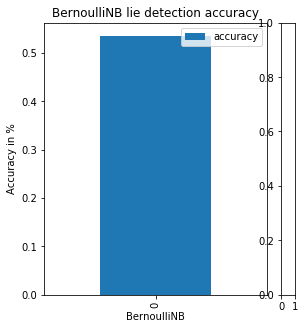
The accuracy of this model is about 78% as shown in **Figure 3.2** shows the distribution of prediction error for sentiment analysis.



**Figure 3.2 Accuracy of the Model 1.2**

**Model 1.3: Lie Detection using Naïve Bayes BernoulliNB**

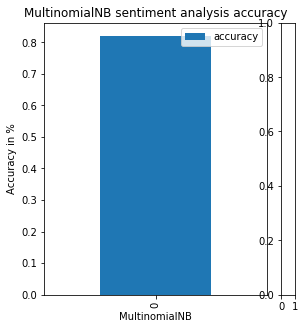
The accuracy of this model for predicting fake and true reviews is about 53% as shown in **Figure 3.1**



**Figure 3.3 Accuracy of the Model 1.3**

**Model 1.4: Sentiment Analysis using Naïve Bayes BernoulliNB**

The accuracy of this model is about 75%. **Figure 3.4** shows the distribution of prediction error for sentiment analysis.

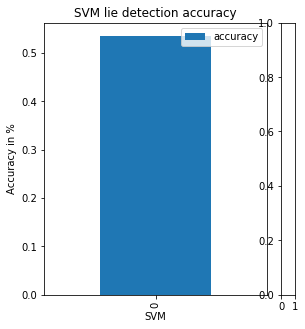


**Figure 3.4 Accuracy of the Model 1.4**

#### **Support Vector Machine**

**Model 1.5: Lie Detection using SVM linear algorithm**

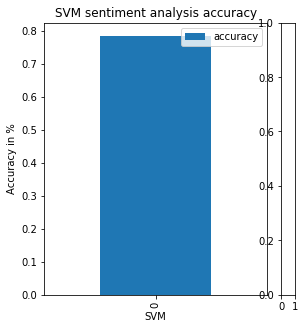
The accuracy of this model for predicting fake and true reviews is about 53% as shown in **Figure 3.5**



**Figure 3.5 Accuracy of the Model 1.5**

**Model 1.6: Sentiment Analysis using SVM linear algorithm**

The accuracy of this model is about 71%. **Figure 3.6** shows the distribution of prediction error for sentiment analysis.



**Figure 3.4 Accuracy of the Model 1.6**

#### **Algorithm performance comparison**

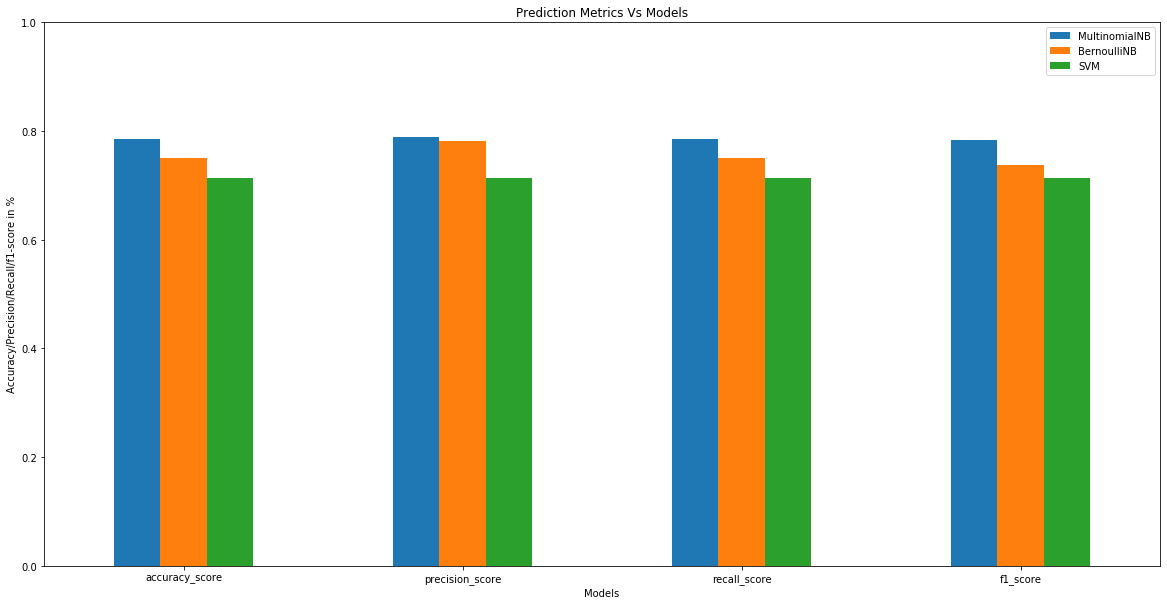
After performing 3 cross validation, the average time took to build the model and perform the test prediction did not show much variance as our dataset is small.

## **Conclusion**

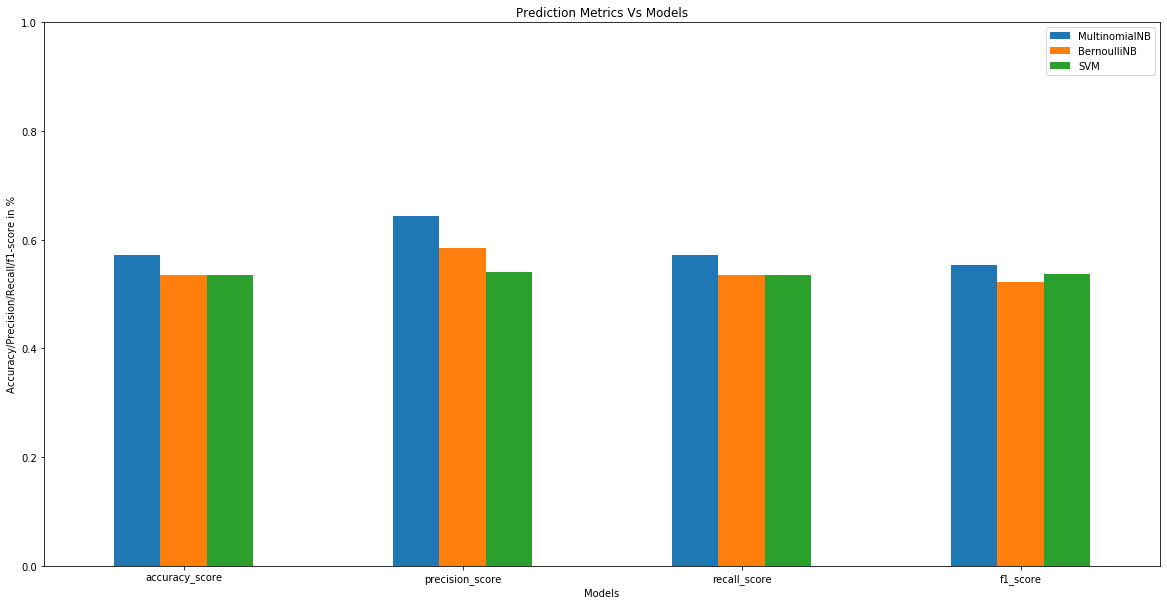
After analyzing the ability of various algorithm to detect lies and sentiment, it is observed that Multinomial Naïve Bayes outperforming Bernoulli and SVM models. The accuracy of Multinomial Naïve Bayes in detecting lies is 57% whereas the Bernoulli and SVM models has got 53%. Similarly, the accuracy of Multinomial Naïve Bayes in detecting sentiment is 78% whereas the Bernoulli and SVM models are 75% and 71% respectively.

Also, when comparing accuracy of all the models between lie detection and sentiment analysis. It is observed that sentiment analysis can be easily predicted with better accuracy than lie detection.

Please refer **Figure 4.1** which compares the accuracy of different models and its accuracy, precision and recalls.



**Figure 4.1 Prediction Metrics comparison for sentiment analysis between Models**



**Figure 4.2 Prediction Metrics comparison for Lie Detection between Models**

The above inference shows that Naïve Bayes model is better than SVM models for sentiment analysis and lie detection.