International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 5, pp: 273-280 www.ijaem.net ISSN: 2395-5252

Implementation of Tweet Sentiment through Naive Bayes Algorithm

Shrishti*, Shweta Dwivedi, Santosh Kumar
School of Computer Science, Maharishi University of Information Technology, Lucknow, 226013, INDIA†
School of Computer Science, Maharishi University of Information Technology, Lucknow, 226013, INDIA
School of Computer Science, Maharishi University of Information Technology, Lucknow, 226013, INDIA
Corresponding author: Shrishti

Date of Submission: 08-08-2020 Date of Acceptance: 24-08-2020

ABSTRACT: Twitter is a well-known smaller scale running on the web administration to compose a blog or remark that place client's impact status messages (called "tweets"). These tweets every now and then explicit conclusions contacting one of kind points. The intention in regards to that request is in similarity with construct a calculation so that execute correctly. Twitter messages might be in positive or negative, along regard as per an inquiry term or according to disposition of client. Our theory is to that sum we perform accomplishing unnecessary appropriateness over arranging feeling of Twitter messages by the utilization of AI strategies. Guileless Bayes Classifier to be specific that is a typical classifier with capacities in Natural Language Processing (NLP). Regardless of its effortlessness, that is capable as indicated by gain over normal execution of unmistakable obligations like sense examination. We expound thoughts in regards to it model or underneath put into impact it inside Python. For the most part, this sort over opinion examination is helpful in light of the fact that purchasers whosoever are endeavouring in congruity with research an item or administration, yet advertisers picking up information on masses decision for theirs organization.

Keywords: Sentiments, Naive Bayes, Tweets, Natural Language Processing, Entropy, Bigrams, Semantics, Open NLP.

I. INTRODUCTION

Sentiments are contemplations which are come in anyone's psyche immediately. So these are most alterable marvels of any ones days. That's why notions investigation is urgent piece of any strategy or strategies to catch the example of supposition. Opinion are sticks according to huge time factor risking, presently someone's feeling are gotten basic like on the off chance that he search 10 tunes in YOUTUBE, at that point at prepared 20 to 30 % information will coordinate with his last inquiry and his customary pursuit. Here we utilizing Naives Bayes to investigation opinions. As notions are consistently produced ancient rarities. Guileless Bayes given a class c, the nearness of an individual element of our record is autonomous on the others.

1.1. Sentiment Definition

Sentiment can be defined as "a personal positive or negative feeling." Here are some examples:

Table 1. Sample Sentiment and Tweet

Sentiment	Tweet
Positive	Ramesh: Rani is my new best friend.
Neutral	Rani: I know Ramesh
Negative	Rahul: Friendship with Rani is Very Difficult.

In the event that we have a few tweets and not satisfactory about their temperament, we can utilize the accompanying litmus test: If the tweet would ever show up as a paper title text or as a sentence in Wikipedia, it has a place in the nonpartisan class. For instance, the accompanying tweet would be set apart as unbiased in light of the fact that it is truth from a paper title text, despite the fact that it anticipates a general negative inclination about Good Morning :

A Naive Bayes Classifier is utilized here for breaking down slant. Essential math documentations which are utilizing here for

* Typeset names in 8 pt Times Roman, uppercase. Use the footnote to indicate the present or permanent address of the author.
† State completely without abbreviations, the affiliation and mailing address, including country. Typeset in 8 pt Times Italic.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 5, pp: 273-280 www.ijaem.net ISSN: 2395-5252

effectively group a survey as positive or negative. These are the two classes to which each report has a place. In increasingly numerical terms, we need to locate the most plausible class given a report, which is actually what the recipe passes on.

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d)$$

C is the arrangement of every single imaginable class, cone of these classes and the archive that we are presently characterizing. We read P(c|d) as the likelihood of class c, given record d. This condition can be modified by utilizing the notable Bayes' Rule, one of the most key guidelines in AI. Since we need to boost the condition we can drop the denominator, which doesn't rely upon class c.

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} \overbrace{P(d|c)}^{\text{likelihood}} \overbrace{P(c)}^{\text{prior}}$$

The modified type of our classifier's objective normally parts it into two sections, the probability and the earlier. You can think about the last as "the likelihood that given a class c, record d has a place with it" and the previous as "the likelihood of having an archive from class c". To go above and beyond we have to present the presumption that gives this model its name. Gullible Bayes supposition: given a class c, the nearness of an individual element of our report is free on the others. We think about every individual expression of our archive to be an element. In the event that we compose this officially we acquire:

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{f \in F} P(f|c)$$

The Naive Bayes presumption lets us substitute P(d|c) by the result of the likelihood of each component moulded on the class since it accept their freedom. We can roll out one greater improvement: boost the log of our capacity. The explanation behind this is simply computational, since the log space will in general be less inclined to undercurrent and progressively productive. We show up at the last plan of the objective of the classifier.

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} \log P(c) + \sum_{i \in positions} \log P(w_i|c)$$

So how precisely does this reformulation help us? How about we take a gander at each term independently.

- P(c) is basically the likelihood of experiencing a report of a specific class inside our corpus. This is effortlessly determined by simply partitioning the quantity of events of class c by the all-out number of archives.
- P(wi|c) is the likelihood of word wi happening in a report of class c. Again we can utilize the frequencies in our corpus to process this. This will essentially be the occasions word wi happens in reports of class c, isolated by the whole of the tallies of each word that shows up in records of class c.

All the terms processed here, implying that figure out the most probable class of this test record. There is just one issue that have to manage i.e. zero probabilities.

1.2. Smoothing

Envision that we are attempting to group an audit that contains the word 'stupendous' and that our classifier hasn't seen this word previously. Normally, the likelihood P(wi|c) will be 0, causing the second term of our condition to go to negative unendingness! This is a typical issue in NLP however fortunately it has a simple fix with smoothing. This strategy comprises in adding a steady to each include in the P(wi|c) recipe, with the most fundamental sort of smoothing being called include one (Laplace) smoothing, where the consistent is only 1.

$$\hat{P}(w_i|c) = \frac{count(w_i,c)+1}{\sum_{w \in V}(count(w,c)+1)} = \frac{count(w_i,c)+1}{\left(\sum_{w \in V} count(w,c)\right)+|V|}$$

This tackles the zero probabilities issue and represented later exactly the amount it impacts the exactness of our model.

II. PROCEDURE-DATA COLLECTION

Information assortment for the exploration isn't as straightforward as it might appear at intense idea. There are suspicions and choices to be made. There are three contrastingly gathered datasets: test

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 5, pp: 273-280 www.ijaem.net ISSN: 2395-5252

information, emotional preparing information, and goal (impartial) preparing information. There are no current informational collections of Twitter conclusion messages. We gathered our own arrangement of information. For the preparation information, we gathered messages that contained the emojis :) and :(through the Twitter API. The test information was physically. A lot of 75 negative tweets and 108 positive tweets were physically checked. A web interface instrument was worked to help in the manual grouping task.

2.1 Twitter API

Twitter gives two APIs: REST and Streaming. REST API comprises of two APIs: one just called the REST API and another called SEARCH API. The contrast between Streaming API and REST APIs are: Streaming API bolsters seemingly perpetual association and gives information in practically ongoing.

2.2 Training Data

There are two datasets that are utilized for the preparation of a classifier: abstract information and nonpartisan information. Abstract information is information that includes positive and additionally negative notion while unbiased information is information that doesn't show opinion. The accompanying information was gathered to be utilized to prepare a classifier.

2.3 Subjective Data Collection

Abstract information in this setting is information that contains negative or potentially positive emojis. While it is conceivable to gather enough negative and positive information in a couple of sequential days, the emotional information was gathered in four non-successive days to empower irregularity. Negative and positive tweets are additionally gathered simultaneously as opposed to gathering them in various days. Gathering them simultaneously was viewed as acceptable in light of the fact that it can catch some unpretentious contrasts among negative and positive Twitter posts for a few.

2.4 Neutral Tweets

For impartial tweets, tweets are gathered from Twitter records of 20 significant papers, for example, Times of India, The Hindu, Dainik Jagrans and others that a news title text is a nonpartisan post. They were gathered on days not the same as when the emotional information was gathered. The proposed classifier is actualized as a Naive Bayes Classifier class. The calculation is part into two fundamental parts, for example, preparing and characterizing.

III. TRAINING

In this stage, we furnish the classifier with an (ideally) huge corpus of text, indicated as D, which figures all the tallies important to register the two terms of the reformulated. Inside the circle we simply follow the request as given in the pseudocode.

for each class $c \in C$ # Calculate $P(c)$ terms
N_{doc} = number of documents in D
N_c = number of documents from D in class c
$logprior[c] \leftarrow \log \frac{N_c}{N_{doc}}$
$V \leftarrow$ vocabulary of D
$bigdoc[c] \leftarrow$ **append**(d) **for** $d \in D$ **with** class c
for each word w in V # Calculate $P(w|c)$ terms
$count(w,c) \leftarrow$ # of occurrences of w in $bigdoc[c]$
$loglikelihood[w,c] \leftarrow \log \frac{count(w,c)+1}{\sum_{w' \text{ in } V}(count(w',c)+1)}$
return $logprior, loglikelihood, V$

Program 1 - Pseudocode for Naive Bayes training

When implementing, although the pseudocode starts with a loop over all classes, we will begin by computing everything that doesn't depend on class c before the loop. This is the case for N_doc, the vocabulary and the set of all classes.

1. First, count the number of documents from D in class c.
2. Calculate the log prior for that particular class.
3. Make a loop over our vocabulary so that can get a total count for the number of words within class c.
4. Finally, compute the log-likelihoods of each word for class c using smoothing to avoid division-by-zero errors.

2.5 Classifiers

A few distinct classifiers were utilized. A Naive Bayes classifier was worked without any preparation. Outsider libraries were utilized for Maximum Entropy and Support Vector Machines. The accompanying table sums up the outcomes.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 5, pp: 273-280 www.ijaem.net ISSN: 2395-5252

Table 6. Accuracy results from various classifiers

Accuracy	Keyword	Multinomial Naive Bayes Unigram	Multinomial Naive Bayes Unigram using MI	MaxEnt OpenNlp	MaxEnt Stanford Classifier (Sigma= 10)
Unigram	0.426229508196721	0.808743169398907	0.841530054644809	79.23	0.61
Unigram Limited with feature threshold = 100	-	0.808743169398907	-	-	0.63
Unigram + POS	-	0.748633879781421	0.781420765027322	-	0.683
Bigram	-	0.759562841530054	0.737704918032786	0.7322	0.667

Preparing size likewise affects execution. Figure 1 shows the impact of preparing size on precision. At the point when the preparation is done we have all the fundamental qualities to make a forecast. This will essentially comprise in taking another (inconspicuous) record and processing the probabilities for each class that has been seen during preparing.

```
for each class c ∈ C
    sum[c] ← logprior[c]
    for each position i in testdoc
        word ← testdoc[i]
        if word ∈ V
            sum[c] ← sum[c] + loglikelihood[word,c]
return argmax_c sum[c]
```

Pseudocode for the classification part

2. Naïve Bayes Algorithm

Naive Bayes is a simple model for classification. It is straightforward and functions admirably on text categorization. A multinomial Naive Bayes algorithm is adopted here which assumes each feature is conditional independent to other features given the class.

$$P(c \mid t) = \frac{P(c)P(t \mid c)}{p(t)}$$

where c is a specific class and t is text which is to classify. P(c) and P(t) is the prior probabilities of this class and this text. And P(t|c) is the probability the text appears given this class. Here, the value of class c might be POSITIVE or NEGATIVE, and t is just a sentence.

The goal is choosing value of c to maximize P(c | t):

Where P(wi|c) is the likelihood of the ith include in text t seems given class c. We have to prepare boundaries of P(c) and P(wi|c). It is basic for getting these boundaries in Naive Bayes model. They are simply greatest probability estimation (MLE) of every one. When making expectation to another sentence t, we ascertain the log probability log P(c) + ∑i logP(wi|c) of various classes, and take the class with most noteworthy log probability as forecast.

In practice, it needs smoothing to maintain a strategic distance from zero probabilities. Something else, the probability will be 0 if there is a concealed word when it making forecast. We just use include 1 smoothing in this examination work and it functions admirably.

a) Feature selection

For unigram include, there are normally 260,000 unique highlights. This is an exceptionally huge number. It makes model higher fluctuation. (Since increasingly convoluted model has higher fluctuation). So it will require substantially more preparing information to abstain from overfitting. Our preparation set contains many thousands sentences. Be that as it may, it is as yet an enormous number of highlights for our preparation set. It is useful on the off chance that we dispose of some pointless highlights. We attempt 3 distinctive element determination calculations.

b) Frequency-based feature selection

This is the least difficult approach to do highlight determination. We simply pick highlights

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 5, pp: 273-280 www.ijaem.net ISSN: 2395-5252

(unigram words for our situation) for each class with high recurrence event in this class. By and by, if the quantity of events of an element is bigger than some limit (3 or 100 in our trials), this element is a decent one for that class. As we found in the outcome table, this essentially calculation increments about 0.03 of precision.

c) Mutual Information

The idea of mutual information is, for each class C and each feature F, there is a score to measure how much F could contribute to making correct decision on class C. The formula of MI score is,

$$MI(C;F) = \sum_{ef \in \{1,0\}} \sum_{ec \in \{1,0\}} P(C=ec, F=ef) \log \frac{P(C=ec, F=ef)}{P(C=ec)P(F=ef)}$$

In practice, add-1 smoothing is used for each Count(C = ec, F = ef) to avoid divided by zero. The code is below.

```
double n = polarityAndFeatureCount.totalCount() + 4;
for(String feature: featureCount.keySet())
{
for(int polarity : polarityCount.keySet())
{
double n11 = polarityAndFeatureCount.getCount(polarity, feature) + 1;
double n01 = polarityCount.getCount(polarity) - polarityAndFeatureCount.getCount(polarity, feature) + 1;
double n10 = featureCount.getCount(feature) - polarityAndFeatureCount.getCount(polarity, feature) + 1;
double n 00 = n - (n11 + n01 + n10); double n1dot = n11 + n10;
double n0dot = n - n1dot; double ndot1 = n11 + n01; double ndot0 = n - ndot1;
double miScore = (n11 / n) * Math.log((n * n11) / (n1dot * ndot1))+ (n01 / n) * Math.log((n * n01) / (n0dot * ndot1))+ (n10 / n) * Math.log((n * n10) / (n1dot * ndot0))+ (n00 / n) * Math.log((n * n00) / (n0dot * ndot0));
mi.setCount(polarity, feature, miScore);
}
}
```

$$\chi^2(F,C) = \frac{N(N_{11}N_{00} - N_{10}N_{01})^2}{(N_{11}+N_{01})(N_{11}+N_{10})(N_{10}+N_{00})(N_{01}+N_{00})}$$

only top k features with highest scores will be picked after calculating MI score for feature set to test. Although if k is small, the model is too simple that data is under fitting. But if k is large, the model is too complicated that data is over fitting. The best number of features in our unigram case is about 40,000. As k grow up to 20,000, the accuracy and F score are also grow up quickly. This is because in this area, the model is high bias. So it is helpful to add features to avoid under fitting data. When the number is larger than 100,000, the accuracy and F score decrease gradually. Since the large number of features makes model so complicated that there are not enough training sentence to avoid over fitting.

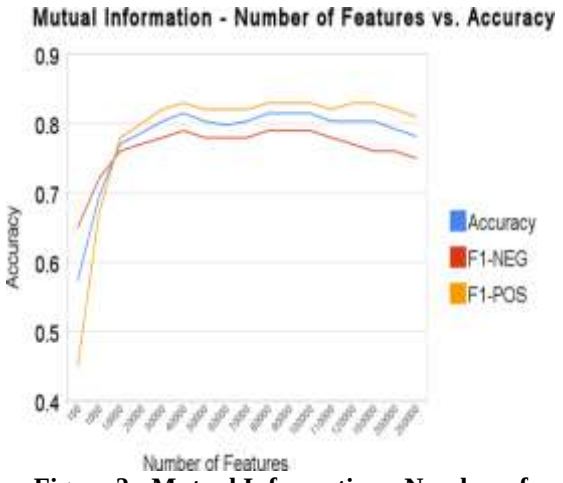

Figure 2 - Mutual Information - Number of Features vs. Accuracy

a) X^2 Feature Selection

The possibility of X^2 Feature determination is comparative as common data. For each component and class, there is additionally a score to gauge if the element and the class are autonomous to one another. It utilizes X^2 test, which is a measurement technique to check if two occasions are autonomous. It accept the element and class are free and computes X^2 esteem. The huge score infers they are not free. For instance, the basic estimation of 0.001 is 10.83. This implies, in the event that they are autonomous to one another, at that point the likelihood this score bigger than 10.83 is just 0.001. On the other hand, in the event that the score is bigger than 10.83, at that point it is far-fetched the element and the class autonomous. The bigger the score is, the higher reliance they have. So we need save highlights for each classes with most noteworthy X^2scores. The equation of X^2score is,

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 5, pp: 273-280 www.ijaem.net ISSN: 2395-5252

2.6 Maximum Entropy

The idea behind MaxEnt classifiers is that we should prefer the most uniform models that satisfy any given constraint. MaxEnt models are feature based models. We use these features to find a distribution over the different classes using logistic regression. The probability of a particular data point belonging to a particular class is calculated as follows:

$$p(c \mid d, \vec{\lambda}) = \frac{\exp\left[\sum_i \lambda_i f_i(c, d)\right]}{\sum_{c'} \exp\left[\sum_i \lambda_i f_i(c', d)\right]}$$

Where, c is the class, d is the data point we are looking at, and λ is a weight vector.

MaxEnt makes no independence assumptions for its features , unlike Naïve Bayes . This means we can add features like bigrams and phrases to MaxEnt without worrying about feature overlapping. There are two packages for the MaxEnt implementation such as the Stanford Classifier and the OpenNLP package.

2.7 Performance

The Stanford Classifier bundle gave terrible outcomes for the default boundary settings. Over various preparing sizes (Figure 1) it improved a piece, yet was a great deal more awful than different classifiers. the smoothing constants were changed, yet it never got extremely near the NB classifier regarding precision. As appeared in Figure 3, changed sigma (smoothing) values didn't contribute a lot to higher precision.

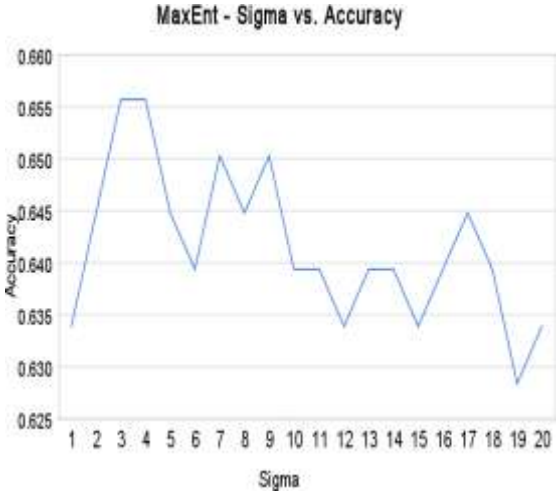

Figure 3. Sigma (the smoothing parameter) vs accuracy

Basic unigram feature selection for NB	Presence	Frequency-based(3)	Frequency-base(Presence, 100)	MI (Presence, 40000)	Chi2 (Presence, 40000)
Accuracy	0.7650273224043	0.78142076502732	0.79781420765027	0.81420765027322	0.80327868852459
F1-NEG	0.73	0.75	0.79	0.79	0.78
F1-POS	0.79	0.81	0.81	0.83	0.83

After testing for different smoothing values and trying different functions in place of Conjugate Descent, we decided to try OpenNLP's MaxEnt classifier since time was running short.

MaxEnt from OpenNLP did perform considerably better. As one can see from Figure 1, MaxEnt performs similar to how the NB performs. Since it doesn't significantly improve performance and takes very long to train and test, we decided to pursue NB for some other experiments.

IV. NAÏVE BAYES ERROR ANALYSIS

1) Example 1

Naive Bayes's independence assumption sometimes causes havoc in classification. This is most notable for negative words like "not" that precede adjectives. Here is an example:

As you may have noticed, not too happy about the GM situation, nor AIG.

The actual sentiment is negative, but Naive Bayes predicted positive. The unigram model has a probability on the word "happy" for the positive class, which doesn't take into account the negative word "not" before it.

2) Example 2

In some cases, the proposed language model was simply not rich enough. For example, the Naive Bayes classifier failed on the following example:

"Cheney and Bush are the real culprits - http://fwix.com/article/939496"

The genuine estimation is negative, however the Naive Bayes classifier anticipated positive. The explanation is that "guilty parties" just happened once in preparing information, as a positive assessment. The stemming words may help here in light of the fact that "guilty party" shows up in the preparation corpus: 1 time in the positive class and multiple times in the negative class. The Porter Stemmer in the unigram highlight extractor to help with this circumstance, yet it wound up cutting down by and large precision by 3%..

V. RESULT AND IMPLEMENTATION

Now that is some accuracy, Smoothing makes the proposed model good enough to correctly classify at least 4 out of 5 reviews, a very nice result. The training and predicting both together take at most 1 second which is a relatively low runtime for a dataset with 2000 reviews.

Following Improvements can be made for future:

1. Our algorithms classify the overall sentiment of a tweet. Depending on whose perspective you're seeing the tweet from the polarity may change.
2. Part of Speech (POS) tagger - The POS tagger took about 3 hours to train and hence we could not run too many tests on it. It did improve the accuracy in case of Maxent and could have been helpful to NB with some more variations but we didn't have enough time to conduct these tests.
3. Domain-specific tweets - Our classifiers produce around 85% accuracy for tweets across all domains. This means an extremely large vocabulary size. If limited to particular domains (such as movies) we feel our classifiers would perform even better.
4. Support Vector Machines - SVM performed the best when classifying movie reviews as positive or negative. An important next step would be to further explore SVM parameters for classifying tweets.
5. Handling neutral tweets - In real world applications, neutral tweets cannot simply be ignored. Proper attention needs to be paid to neutral sentiment. There are some approaches that use a POS tagger to look at adjectives to determine if a tweet contains an sentiment.
6. Dealing with words like "not" appropriately- Negative words like "not" have the magical affect of reversing polarity. Our current classifier doesn't handle this very well.
7. Ensemblemethods- Asingleclassifiermaynotbethebestapproach.Itwouldbeinterestingtoseewhattheresultsare for combining different classifiers. For example, we thought about using a mixture model between unigrams and bigrams. More sophisticated ensemble methods, like boosting, could be employed.
8. Using cleaner training data. Our training data does not have the cleanest labels. The emoticons serve as a noisy label. There are some cases in which the emoticon label would normally not make sense to a human evaluator. For example user ayakyl tweeted, "agghhhh :) looosing my mind!!!!" If we remove the emoticon from this phrase, it becomes "agghhhh looosing my mind!!!!" in which a human evaluator would normally assess as negative.

VI. CONCLUSION

From the above examination work, it is seen, even an exceptionally fundamental execution of the Naive Bayes calculation can prompt shockingly great outcomes for the errand of supposition investigation. Notice that the proposed model is basically a paired classifier, implying that it very well may be applied to any dataset that has two classifications. There are a wide range of uses for it, running from spam discovery to Bitcoin exchanging dependent on supposition. With a precision of 82%, there is actually a great deal that you could do, all you need is a named dataset and obviously, the bigger it is, the better.

ACKNOWLEDGMENTS

Authors are appreciative to the Vice-Chancellor, Maharishi University of Information Technology Lucknow for giving the magnificent office in the registering lab of Maharishi college of Information Technology, Lucknow, India. Much appreciated are likewise because of University Grant Commission, India for help to the University.

REFERENCES

[1]. Saroj Kumar, Santosh Kumar, "World Wide Web - Cloud Boundaries", International Journal of Computer Sciences and Engineering, Vol.7, Issue.6, pp.483-490, 2019.

[2]. Saroj Kumar,Ankit Kumar Singh,Priya Singh,Abdul Mutalib Khan,Vibhor Agrawal Mohd Saif Wajid, "Sentiment Analysis Based on A.I. Over Big Data", Publisher Name Springer, Singapore, Print ISBN978-981-10-1677-6, Online ISBN978-981-10-1678- 3, https:// link.springer.com/ chapter/10.1007/978-981-10-1678-3_61.

[3]. Saroj Kumar, Priya Singh, Shadab Siddiqui "Cloud security based on IaaS model prospective"
[4]. Ankit Kumar Singh, Saroj Kumar and Abhishek Rai "Secure Cloud Architecture Based on YAK and ECC" International Journal of Computer Applications, ISBN 0975 – 8887 Volume 90, Number 19 (March 2014), pp. 29–33, © International Journal For Computer Application, http://www.ijca.org.
[5]. Ankit Kumar Singh, Saroj Kumar and Abhishek Rai " CLOUD SERVICE ARCHITECTURE FOR EDUCATION SYSTEM UNDER OBJECT ORIENTED METHODOLOGY" International Journal of Research in Engineering and Technology, eISSN: 2319-1163 | pISSN: 2321-7308 Volume: 03 Special Issue: 10 | NCCOTII 2014 | Jun-2014, Available @ http://www.ijret.org.
[6]. Akhilesh Kumar, Saroj Kumar " ENERGY SAVING MODEL AND APPLICATION FOR SMART PHONES" International Journal of Research in Engineering and Technology, eISSN: 2319-1163 | pISSN: 2321-7308 Volume: 03 Special Issue: 10 | NCCOTII 2014 | Jun-2014, Available @ http://www.ijret.org .
[7]. Ashutosh Gaur, Saroj Kumar " PROPOSED MAC PROTOCOL FOR REDUCE ENERGY CONSUMPTION OVER WSN NETWORK" International Journal of Research in Engineering and Technology, eISSN: 2319-1163 | pISSN: 2321-7308 Volume: 03 Special Issue: 10 | NCCOTII 2014 | Jun-2014, Available @ http://www.ijret.org
[8]. Saroj Kumar, and Priya Singh "Cloud Computing Applications Architecture - ArchJava" International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, Volume 2, Issue 6, June 2012, www.ijetae.com.
[9]. B.Jansen, M. Zhang, K. Sobel, A. Chowdury. The Commerical Impact of Social Mediating Technologies: Micro-blogging as Online Word-of-Mouth Branding, 2009.
[10]. B.Manning and H. Schuetze. Foundations of Statistical Natural Language Processing. 1999.
[11]. B. Pang, L. Lee, S. Vaithyanathan. Thumbs up? Sentiment Classification using Machine Learning Techniques, 2002.
[12]. B. Pang and L. Lee. "Opinion Mining and Sentiment Analysis" in Foundations and Trends in Information Retrieval, 2008.
[13]. B. Pang and L. Lee. "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts" in Proceedings of ACL, 2004.
[14]. J. Read. Using Emotions to Reduce Dependency in Machine Learning Techniques for Sentiment Classification, 2005.
[15]. P. Turney. "Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews" in Proceedings of the 40th Annual Meeting of the Association for Computatoinal Linguistics (ACL), 2002.