Sentiment Analysis on Product Reviews

INSE6180: Security and Privacy Implications of Data Mining

By

Pazim Goyal

40069412

Abstract:

A Sentiment is a view of or attitude toward a situation or event hence an opinion. Sentiment Analysis is basically opinion extraction. The term sentiment analysis can be used to distinguish between two different concepts one being extracting and understanding the sentiments being expressed in text documents whereas the other being the task of assigning class labels to the documents. In this project my goal is to analyses the text reviews from an online source and convert useful data into a point matrix, then perform the desired analysis on the data thus obtained. The paper elaborately discusses three supervised machine learning algorithms: K-Nearest Neighbour(K-NN), Naïve Bayes' and Support vector machine (SVM) and compares their overall accuracy as well as time taken to compute the results.

Introduction:

Data mining is practice of examining large dataset in order to generate new information. There are several data mining tools such as clustering, classification, regression etc. those could be used for sentiment analysis. Sentiment analysis is one of the important aspects of data mining as the data could be classified into classes i.e. positive or negative class. Sentiment analysis uses natural language processing, text analysis, to systematically identify, extract, quantify and study affective states and subjective information. In essence, it is the process of data mining the emotional tone behind a series of words, used to gain an understanding of the opinions hence the term opinion mining. it is extremely useful in social media monitoring as it allows an overview of the wider public opinion. The ability to extract incites from social data is a practice that is being widely adopted by organizations across the world. Opinion analysis or sentiment analysis primarily uses data mining processes and techniques to extract and capture data for analysis in order to discern the subjective opinion of a document or collection of documents like review, news, social media feeds, it is a type of data mining that measures the inclination of people's opinion, the general publics sentiments or reactions towards certain products, people, or ideas revile the contextual polarity of the information. E.g. "This is a good phone. I love it" this is a positive sentiment as it contains words like good, love. These words are mostly found in positive reviews and rarely found in negative reviews. In this paper we are using three supervised machine learning algorithms Naïve Bayes', K-nearest Neighbour (KNN) and Support Vector Machine (SVM).

Related Work:

few related works are as following

Naïve Bayes: https://nlp.stanford.edu/IR-book/html/htmledition/naive-bayes-text-classification-1.html

https://web.stanford.edu/~jurafsky/slp3/slides/7_NB.pdf

KNN: https://www.sciencedirect.com/science/article/pii/S1877705814003750

SVM: http://www.ijesi.org/papers/Vol(4)11/G411033037.pdf

Proposed work:

Data Source or Dataset: Amazon is an e-commerce website which gather millions of reviews for all the products, all these review datasets are available online for research and other purposes. For this project we are using dataset which contain reviews about Phones and its Accessories. These datasets had several attributes such as name of person who reviewed, timestamp, rating, review. But for this project we trimmed the data and only used attributes those are actual review and whether the result is positive or negative represented by 0 or 1 respectively.

- The datasets could be found at http://jmcauley.ucsd.edu/data/amazon/

The data is converted from JSON to CSV format for easy use. Final training datasets look like table below with 20000 reviews and testing dataset with 10000 reviews.

1	This is a good phone. I love it.
0	Don't buy this phone. Total wastage of money

Sample dataset table

Preprocessing: Preprocessing is one of the most important state in this project.

Stop-Words: Stop words are words like "is", "am", "the" are most commonly occurring words in dataset. The usually don't provide much information about the sentiment

Algorithm

i. Read the training.csv file from the storage
ii. Put the data in two columns one with reviews and other with class of review positive or negative, 1 or 0 respectively with names 'text' and 'rate'
iii. Read stop-words file from storage and save all the words in a python sets
iv. For each word in each document in text column add the word to a list if does not exist in stop-words set.
v. Add the list to a new list with corresponding class of review.

Multinomial Naïve Bayes': Multinomial naïve bayes is one of the best approach in text classification. It is fast, reliable and better then other classification algorithms in terms of speed and accuracy. It works simply on concept of probability. For this it uses bag of words. It is simply the reliable algorithm because it works on principle of probability. Calculating probability of each word in document and calculating the final result on basis of higher probability.

<u>Bag of Words Representation:</u> The bag-of-words model is a simplifying representation used in natural language processing. Each key is the word, and each value is the number of occurrences of that word in the given text document.

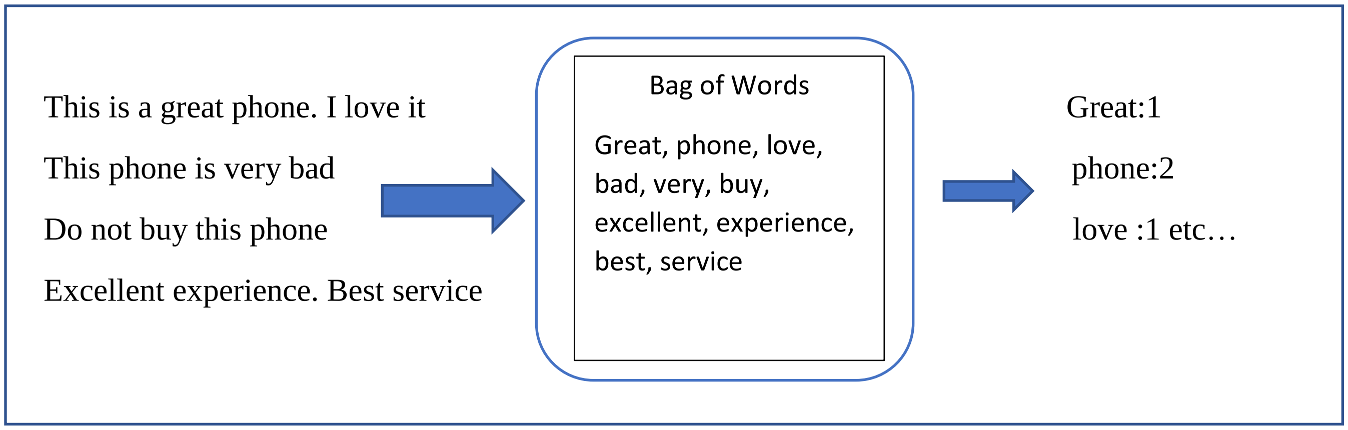

This is a great phone. I love it
This phone is very bad
Do not buy this phone
Excellent experience. Best service
Bag of Words
Great, phone, love, bad, very, buy, excellent, experience, best, service
Great:1
phone:2
love :1 etc…

Figure: e.g. bag of words

<u>Algorithm Multinomial Naïve Bayes'</u>

Training

i. read the preprocessed data
ii. create an two empty dictionaries (Key – value pairs) for each class i.e positive and negative
iii. for each word in each document of training dataset

    If word doesn't exist in dictionary, Add the word to corresponding dictionary for that class with value 1

    If word does exist in dictionary, increment the value by one.
iv. Calculate total unique words in training dataset

Testing

v. Calculate Prior for each class: $$\hat{P}(c_j) = \frac{doccount(C = c_j)}{N_{doc}}$$

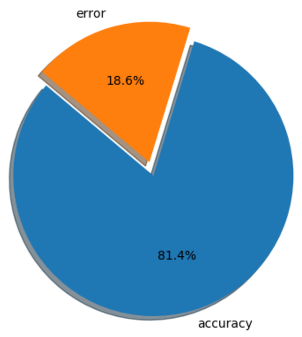
vi. For each word in testing document calculate probability of that word with respect to both the classes. Where count(w_i,c) is the count of the word with respect to class c and ∑count(w,c) count of all the words in that class. |V| is no of all the unique words in the dataset.

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

vii. Take the product of all the obtained probabilities along with the prior calculated above for both the classes.

viii. Classify the document with the class that has higher probability.

Results Naïve Bayes:



Accuracy: accuracy of naïve bayes comes to be 81.4%. This is a good accuracy for text classification problems due to presence of noise in datasets which cant be removed easily.

Time taken for this algorithm is around 15 to 20 seconds.

With increase in dataset its accuracy increases.

K-Nearest Neighbor: KNN is a very popular algorithm for text classification. This paper presents the possibility of using KNN algorithm with TF-IDF method and framework for text classification. the algorithm is based on the machine learning. Preprocessing and document preparation is followed by the learning phase. The algorithm determines the basic documents which will be comparing with each new document. Algorithm checks where a document is categorized by only looking at the training documents that are most similar to it. The algorithm assumes that it is possible to classify documents in the Euclidean space as points. Euclidean distance is the distance between two points in Euclidean space. The distance between two points in the plane with coordinates p=(x, y) and q=(a, b) can be calculated

$$d(p,q) = d(q,p) = \sqrt{(x-a)^2 + (y-b)^2}$$

TF-IDF Matrix

To provide text classification and searching the documents it is necessary to establish the weight matrix. The matrix contains the values of relations between each unique words and documents. It is the initial object in the algorithm to calculate the individual importance (weight) of each searched document. Each document is represented as a vector in n-dimensional vector space. We can imagine a matrix A with dimensions NxM, where N dimension is defined by a number of unique words in a sample of all documents. M represents the number of documents to be classified. Weight matrix can be characterized as a relational matrix of word - document. Dimension of the matrix is equal to product, the number of different unique words and the total number of documents. Each matrix element aij represents weight value of word i in the document j.

weight of each word is calculated as following:

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

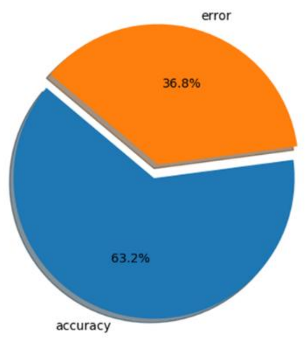
$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

For Classification simply calculate weight of each word in that document and create a row of weight similar to row in tf-idf matrix. Calculate the Euclidean distance between that row and every row of matrix. For given no of k calculate the class with more occurrence with least distance. Classify document with the predicted class.

Results KNN:

_Accuracy: accuracy of KNN comes to be 65%. This is the least accuracy found among all the algorithms applied on this dataset.

Time taken for this algorithm is several hours. As distance has to be calculated between each row of tf-idf matrix.



Support Vector Machine (SVM): Support Vector machines are supervised learning that analyze data used for classification. the text classifier that has been implemented has utilized the idea of supervised learning in order to classify unseen data instances. SVM uses virtual plain to distribute data between two plains. In order to classify the testing document each word in testing document is taken and found the plain it lies in. For all the words in the document, the class is classified the one which has more occurrence in particular plane.

Algorithm

Training

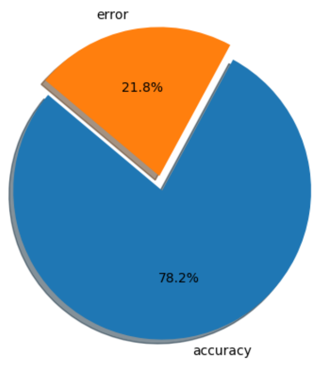
i. Read the preprocessed training data
ii. Create two empty dictionaries one for each class and an empty list for words
iii. For every word in each document in training data check the class of document
iv. If the word is not present in list add the document to the dictionary of corresponding class and set the value to be 1. Add the word to another dictionary and set the value to be 0
v. If the word is present in the list increase the value of the word for the corresponding class.

Testing

i. For document in testing dataset calculate sum for each word in both the dictionaries.
ii. Say the sums are sum1 and sum2
iii. If sum1>sum2 categories, the document to class of dictionary 1
iv. If sum2>sum1 categories, the document to class of dictionary 2

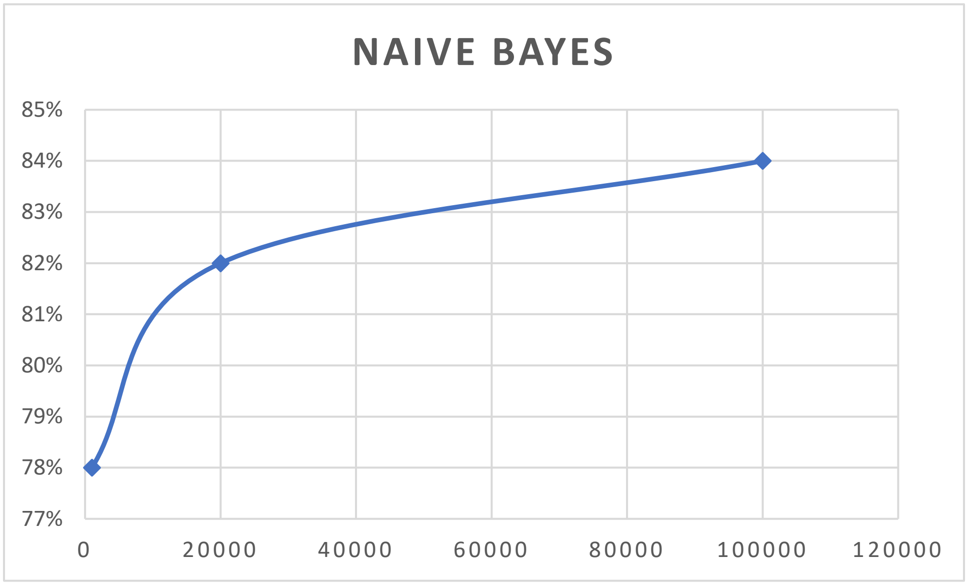
SVM Results:

_SVM is by far not the best algorithm as compared to naïve bayes but if compared to KNN it is better. Time taken for SVM is approximately 1 minute. Its accuracy lies around 78 %. But it may vary depending upon size of dataset.



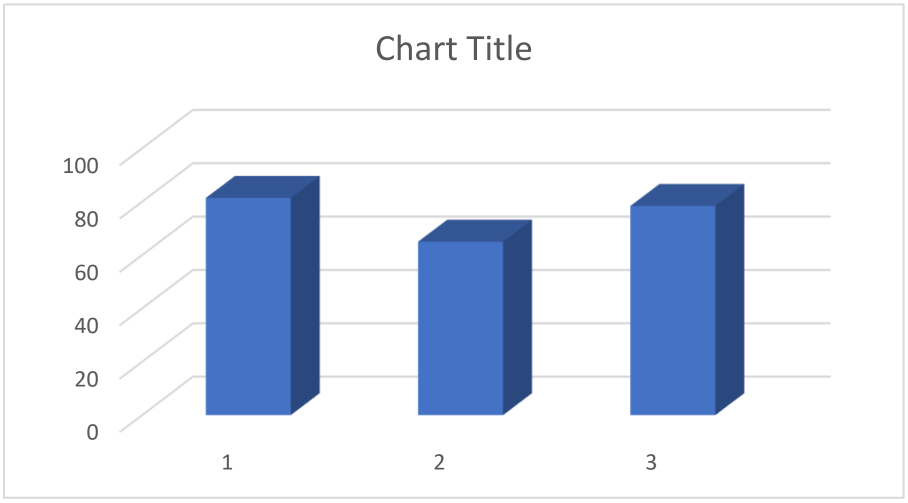
Conclusions:

1. Naïve Bayes is by far the best algorithm for sentiment analysis. It is fast and most accurate among all. It is most reliable algorithm. With increase in dataset its accuracy increases.



Naïve Bayes accuracy with increased dataset

2. KNN is the slowest algorithm among all. Its speed reduces with increase in training dataset hence not suitable for text classification with large datasets. It has low accuracy level.
3. SVM is neither good nor bad it is faster than KNN but slower than Naïve Bayes. Its accuracy will decrease with increase in datasets.



Accuracy comparison of Naïve Bayes , KNN , and SVM