Sentiment Analysis of Online Product Reviews and Comparison of Supervised Machine Learning Algorithms

Created By: Jasmeet Singh Sasan

Abstract

The content generated on social networking websites and forums comprised of human emotions about different features and aspects of products and services. This project is primarily focused on the sentiment analysis of online product reviews. In this project, two research questions have been addressed. Firstly, what type of emotion or sentiments have been expressed by a reviewer in the review. Secondly, the work primarily focused on the comparison of different supervised algorithms' performances in the binary classification task. Thirdly, we have compared the performance of algorithms as the size of the data increases with the same set of features. Finally, we have further compared the performance of different algorithms based on the types of features used to train the algorithm. The study uses 10-fold cross-validation method to avoid the overfitting of data. The study includes basic introduction to Sentiment Analysis, topic modeling, supervised and unsupervised machine learning algorithms, feature selection, part-of-speech tagging and classification of text reviews. The project has implemented the classic and existing state-of-the-art algorithms, and compared their performances in terms of accuracy, precision, recall and F-measure. This project would be useful for the beginners and other researchers working in the areas like Information Retrieval, Information Extraction and Web Mining. At the end, findings have been discussed along with the limitation of the study and its future scope.

Introduction and Motivation

Online social media has become an integral part of human life with its ever increasing usefulness. It plays a critical role and is considered as an important factor in everyday decisions. Forums, social networking and microblogging websites has given freedom to people to express their views regarding anything related directly or indirectly to their lives. The widespread availability of social media has enabled users to share their reviews, recommendation, ratings and feedbacks, about the products or services they have experienced. The easily available online reviews are very helpful to people in many ways for example to those people who are yet to experienced new products and services. It helps people in distinguishing good and bad before making any decision about their investments in anything. The huge text content generated from these website is immensely useful in data mining activities too. It is being studied constantly by researchers for numerous reasons, for example, to learn the psychology of human to better understand the human behavior and their actions. This large amount of data is particularly useful and interesting to the organizations who wants to make sure that their products and services are gaining positive reviews and ratings in the online social world.

With the wide-spread of internet in the past decade, people are more likely to share their opinions and experiences on online social media platforms like Facebook, Twitter, e-commerce websites and blogs which in turn has given birth to massive amount of text data readily available on web. People consume this user-generated data on a daily basis and make their decision based on the opinions and reviews given by others. On the other hand, companies are also utilizing this opinionated user-generated data to gauge and assess the performance of their products.

In the recent years, there has been an exponential increase in the research studies on text mining, machine translation, information retrieval and other Natural Language Processing (NLP) related tasks. There are several reasons which are contributing towards boost in these research domains: one,

enormous amount of data is easily accessible on the internet, two, there exists a mature set of tools and technologies such as Python**,** R, Matlab, Weka etc. which are required and readily available to carry out an efficient research.

An extensive amount of research is focused towards opinion mining also known as Sentiment Analysis to detect the sentiments of a given text document, Sentiment analysis is a classification task in which the text document is classified as positive, negative or neutral. A text document can be anything related to text such as product review, movie reviews, political reviews, news articles, and blogs and so on.
This project contributes to the field of Sentiment Analysis by analyzing the performance of state-of-the-art machine learning algorithms on large datasets.

Sentiment Analysis
Sentiment analysis (or more specifically, sentiment polarity analysis) can be defined as the mapping of text to one of the labels (elements) taken out of a finite predefined set or placing it on the continuum from one end to the other (Pang and Lee, 2008). These are 3 basic elements as positive, neutral and negative. There are several other derived from the basic elements such as 'relevant or irrelevant'; 'in favor of or against'; 'like or dislike'; 'good or bad'. The elements can also be expressed in terms of numbers on a Likert scale of 1 to 10 with 1 as most negative emotion and 10 as most positive emotion. Liu and Zhang has defined it as Sentiment analysis or opinion mining is the computational study of people's opinions, appraisals, attitudes, and emotions toward entities, individuals, issues, events, topics and their attributes."

Sentiment Analysis (SA) is a field of Natural Language Processing (NLP) with a primary objective of extracting subjective information from the text data by developing algorithmic models and theories. The text data with subjective information often contains expressions of opinions and viewpoints. SA touches every aspect of NLP yet confined in several ways. In this project, we have compared the performance of notable and widely used classification algorithms on text data. For this purpose, we have chosen the problem of Sentiment Analysis (SA) of online product reviews.

To measure the results significantly, we have used six different categories of online product review. The contribution of this project is two-fold. This allowed us to measure the impact of text-domain on the classification accuracy of an algorithm and evaluate which algorithm performs better for a particular domain. This further helped us in comparing the performance of different algorithms in a holistic way on text data.

Proposal and Contribution:
This project has contributed to the large scale sentiment classification literature by creating a model framework to compare the performance of the state-of-the-art and classic Machine Learning algorithms in conjunction with combination of multiple Feature Selection techniques. We have created a new approach of combining lexical and non-lexical features for the classification task.

Background and Related Research
In this section, we have discussed some of the influential studies in sentiment analysis over the past decade. As the literature of SA is vast, we have focused on the key studies.

The purpose of this section is to build a background for this project by presenting the previous research work on similar topics. As the literature of SA is vast, we have focused on the key studies. We have discussed some of the influential studies in sentiment analysis over the past decade. We have pointed out some of the basic approaches taken by researchers in the recent years to solve the sentiment classification problem with machine learning approaches. We have also described the most frequent terminologies associated with sentiment analysis.

Agarwal et. Al has examined the twitter text data to analyze the sentiments using features such as POS – specific prior polarity. The authors studied the problem of tedious feature engineering in sentiment analysis by exploring the features generated from a tree kernel. In the study, the authors have used manually annotated data which provided a benefit "in terms of language use and content." They have used Unigram model as the baseline, and their tree kernel and feature based models outperformed the baseline by 4% using SVM classifier. They noted that features generated from the combination of prior polarity of words and their parts-of-speech tags boosts the performance of the models.

Turney (2002) has proposed an unsupervised learning algorithm which works in two steps to classify a text review as recommended or not. Given a text review as input, firstly the algorithm uses a part-of-speech tagger to identify phrases in the input text that contain adjectives or adverbs. Secondly, using PMI-IR algorithm, it estimates the semantic orientation of each extracted phrase. The binary output is based on the average semantics orientation of the phrases extracted from the input text. Semantic orientation is the only feature included in this work which clearly shows that it could be a valuable feature if used in conjunction with other features to achieve high accuracy of sentiment classification.

The study by Hu and Liu (2004) has a major contribution in the field of sentiment analysis. They propose a feature based opinion summarization technique which has three major component. For a given text review, firstly they extracted the opinion words which are called as product features using data mining and NLP techniques such as POS tagging. Frequent feature identification and infrequent feature identification techniques are used to determine the set of features or aspects of products which occurs frequently in the text reviews. Secondly, semantic orientation of extracted opinion words is identified as positive or negative utilizing the WordNet's adjective synonym and antonym sets. Next step extracts the semantic orientation of the opinion sentence using the dominant orientation of the opinion words in each sentence. They also take care of special cases where words like "however", "but" occurs, and the negation words such as "not", "no", "never" with a certain word distance threshold. Lastly, a rank based summary generation technique is used to determine the overall of polarity of a text review based of previous step.

Barbosa and Feng (2010) research the problem of subjectivity detection and sentiment classification from twitter text data and propose a 2-step study which differentiates the text as subjective or objective and based on that performs polarity detection on subjective tweets which classifies them in one of the category: positive, neutral or negative. Variety of features like unigram, bigram, POS tags are used along with meta-information about the words, prior polarity, and tweet syntax features such as retweet, hashtag, reply, link, punctuation, emoticons, upper case word. Their analysis shows that prior polarity is an important feature with an assumption that polarity of the word is associated with the polarity of the

sentence. They compared their work with Unigram (Pang et al, 2004) and ReviewSA (Pang and Lee, 2004) and build the classifier TwitterSA. They report that SVM performs best on Unigram and TwitterSA with maximum confidence score.

Pang, Lee, and Vaithyanathan (2002) has applied the machine learning techniques to the sentiment classification problem of online reviews. They experiment with three standard algorithms such as Naive Bayes, Maximum Entropy and Support Vector Machines on the Movie Review dataset. A variety of features such as Bigrams, Unigrams, Adjectives, POS, word position, negation and their combinations are primarily considered in preparing the models. Presence of features and their frequency in the corpus also played a role in determining the accuracies of models. They report that SVM outperforms other algorithms in the classification task. Also, unigram is the most influential feature among all other features from the performance perspective.

Kim and Hovy (2004) introduced a system which experiments several models to determine the sentiments of text reviews at word and sentence level for a certain topic. They create a word sentiment classifier which uses seed lists of positive and negative words, and cumulatively generate the polarity of a sentence based on the individual polarity of all sentiment-bearing words in it. They defined four windows to define the length of the sentences region. The sentence classifier is built on three different models, firstly, by counting the polarity of words in a sentence without considering polarity strength of the words. Second is based on the Harmonic Mean including the strength of polarity, and third is based on the Geometric Mean. The experiment result are tested against human baseline and first model reports the best performance.

Gamin has approached the sentiment classification problem using a ML technique by training an SVM classifier on noisy customer feedback data. In the study, a combination of linguistic features and surface features such as word ngrams, function word frequencies and POS ngrams are used to build the classifier. The author noted that inclusion of linguistic analysis features improves the performance of classifier. Also, they concluded that the large feature vectors boosts the performance of sentiment classification when combined with a feature reduction technique based on log likelihood ratio.

Abbasi et. al (2008) designed a syntactic and stylistic features based Sentiment Classification model demonstrating the utility of identifying key features in order to achieve high classification accuracy. An Entropy weighted genetic algorithm (EWGA) is designed for the purpose of efficient feature selection while SVM is used for the classification with high level of accuracy. Their results shows that best classification accuracy is achieved by using a combination of syntactic and stylistic features in conjunction with EWGA feature selection method which statistically outperforms when used individually.

Pang and Lee (2004) proposed a machine-learning model to classify the sentiments at the document level. They noted that document-level polarity classification is a special case of text categorization with sentiments. Authors have used SVM as the default classifier in this study.

Machine Learning

According to Arthur Samuel (1959), machine Learning is a field of computer science that "gives computers the ability to learn without being explicitly programmed." Machine learning deals with the study of existing and new algorithms for the purpose of prediction based on the historical data. It can also be considered as an arm of artificial intelligence with primary focus on computational learning and pattern recognition.
In this project, we have focused on the classification problem of machine learning. Classification is a prediction problem where the main aim is to correctly predict the class of an instance.

Supervised Machine Learning Algorithms

Supervised learning algorithms refers to the class of algorithms where each instance of data has an output associated with it. A learning model is created on the training dataset using supervised algorithms. The accuracy of the learned model is checked on another subset of data called test dataset. Error is calculated as the difference between the predicted and actual values of the test dataset. Lower the error value, better the model. These algorithms can be used for classification problems with discrete outcome and the prediction of continuous dependent outcome.

This project is a binary classification study where goal is to correctly classify a text instance as positive or negative. Here, we have used nine most popular and widely used supervise learning algorithms. Below is a brief description of each of these algorithms.

Maximum Entropy

Maximum Entropy is a key algorithm in the NLP research area. It is widely being used in machine translation, Part of Speech tagging, parsing. “Model everything that is known and assume nothing about what is unknown.” As we have already learned about entropy in the previous section. The maximum entropy states that, subject to precisely stated prior data, which must be a proposition that expresses testable information, the probability distribution which best represents the current state of knowledge is one with the largest information theoretical entropy.

In most practical cases, the stated prior data or testable information is given by a set of conserved quantities associated with the probability distribution is question. Generally, it uses Lagrange method to solve the dual optimization problem in order to find the maximum value of parameter λ corresponding to maximum entropy.

SVM

Support Vector Machine is the primary and standard supervised learning algorithm used in majority of the text classification studies. It has outperformed the performance of many algorithms [NR] on different problem domains and datasets. Fundamentally, SVM creates one or more separating hyperplane (also known as decision boundary) in a high dimensional space which are used for the classification, and attempts to find the optimal solution by maximizing the margin around the separating hyperplanes. The margin can be defined as the distance between a hyperplane and the nearest training

data point of a class. In order to have lower classification errors, the margin should be as high as possible. SVM can efficiently perform both linear and non-linear classification tasks. The two commonly used kernels for SVM are the polynomial kernel and the radial basis function (RBF) kernel.

Mathematically, linear SVM is a convex quadratic optimization problem which can be solved using several algorithms. Empirically it works very well.

Naïve Bayes

Naïve Bayes (NB) is a widely used classification algorithm primarily used for clustering and classification tasks across different domains of machine learning. The algorithm is based on deriving the inferences using the Theory of Probability of Bayes under a strong assumption that all events (features) are conditionally independent of each other for a given class.

In plain English, the posterior probability of an event is a function of its prior probability and maximum likelihood with an assumption that all events are conditionally independent of each other. Parameter estimation is done using the maximum likelihood method". The main advantage of NB is that it requires very small amount of training data in order to estimate the parameters for classification tasks. The model finds the probability of a class given a feature.

Naive Bayes represents a distribution as a mixture of components, where within each component all variables are assumed independent of each other. Given enough components, it can approximate an arbitrary distribution arbitrarily closely. Inference time is linear in the number of components and the number of query variables. When learned from data, naive Bayes models never contain more components than there are examples in the data, guaranteeing that inference will be tractable [1]

A Bayesian network encodes the joint probability distribution of a set of n variables, {X1, . . . ,Xn}, as a directed acyclic graph and a set of conditional probability distributions (CPDs). Each node corresponds to a variable, and the CPD associated with it gives the probability of each state of the variable given every possible combination of states of its parents. The set of parents of Xi, denoted i, is the set of nodes with an arc to Xi in the graph. The structure of the network encodes the assertion that each node is conditionally independent of its non-descendants given its parents. The joint distribution of the variables is thus given by P(X1, . . . ,Xn) = Qn i=1 P(Xi|i). For discrete domains, the simplest form of CPD is a conditional probability table, but this requires space exponential in the number of parents of the variable. A more scalable approach is to use decision trees as CPDs, taking advantage of context-specific independencies (Friedman & Goldszmidt, 1996).

Classification Trees

In computer science, a tree is mostly described with three main components as the root node which (carries most of the information), the decision (a test to split the node further into child nodes), the branch (carries the weight from decision test), child nodes (derived from the parent node) and the leaf nodes (the terminal nodes of tree). Figure-x is an example of a tree. A Decision Tree is a supervised learning algorithm which partitions the data into several disjoint pieces on the basis of certain rules.

Breiman brought the concept of Classification and Regression Tree, popularly known as CART. Classification tree are used when the outcome variable is categorical in nature, and regression tree is used when output variable is a continuous and real in nature. The basic idea is to split the bigger problem into smaller pieces using a hierarchical decision scheme. It is a multistage approach to break a complex decision into a set of smaller decisions. The smaller sets are recursively split into subsets. These are popularly known as multistage classifiers. "It allows the rejection of the class label at intermediate stages." For any classification task, there exist a set of predictor variables called features vector and their corresponding class labels called response variable. The decision tree is constructed by recursively partitioning the data based on a set of tests applied to one of more feature values at each branch or node, and assigning class label to the leaf nodes.

Random Forest

Ensemble learning algorithms have received increasing interest because their enhanced accuracy over other classifiers. Ensemble classifiers are important for high dimensional and complex data sets. Also, these algorithms do not make a lot of assumptions such as normality which boosts their effectiveness. Furthermore, they don't require too much tuning to obtain the best performance because of limited parameters involved unlike algorithms like Support vector machine and neural network. The primary hypothesis behind the theory of ensemble classifier is that a set of classifiers outperform a single Random Forest is an important ensemble classifiers which has received significant success in a wider research applications areas and has many advantages over other classifiers such as higher efficiency with large data, capable to handle large number of input variables of the order of thousands, tells the most important variables in the classification, it generates an unbiased estimate of the generalization error, it is relatively robust to outliers and noise in the training data. From computational standpoint, it takes less time in training the model than the other tree based classifiers such as boosting or bagging. [1] Random Forest is an ensemble classifier which uses classification tree as the base classifier, {h(x,THETAk), k =1,2,...} , where x is the input vector and {theta k} are the independent and identically distributed random vectors. {Breiman, 2001; Hastie et al, 2009}

Random Forest uses many classification trees to train a data set and then combines the predictions of all trees fitted individually based on a criteria such as count or voting. The first step in the Random Forest algorithm is to select a certain number of bootstrap observations from the training dataset. The remaining observations which are excluded from the bootstrap samples are called out-of-bag (oob). The bootstrap observations from the training data is sampled and a classification tree is fitted to each sample. At each node, only a small number of randomly selected variables are available for binary partitioning. Note that, it uses a random split of input predicting variables, instead of best split of such input variables. The number of variables at each node is the square root of the total number of variables. The trees are allowed to grow fully without pruning which makes it computationally efficient when compared to a classification tree and each tree is used to predict the oob observations. The predicted class of an oob observation will be the one with majority of the votes for that observation from all sample classification trees, and the ties are split randomly. The generalization error is calculated as an unbiased estimate from the proportion between the misclassified observations and the total number of oob observations.

Furthermore, as the number of trees is increased, the error converges hence avoids overfitting of the data. Since the split of input variables is random, it decreases the correlation between the trees as the individual trees are less strengthened. Additionally, to maximize the dissimilarity between the classes in decision trees techniques like gain-ratio, Gini Index and Chi-square are typically used.

In supervised learning, Random Forest is a significant player in feature selection by identifying the most important and best performing features of the data. This is especially important where data is highly dimensional. RF finds the best performing variables in classification by replacing one random input variable while keeping rest as same. A change in the accuracy on the oob observations is a measure of variable importance by measuring the change in oob error and Gini Index.

Hyper-parameters: Random forest requires two tuning parameters to build a predictive model: number of classification trees and number of input variables required at each node to grow the tree. It is important to optimize these two parameters in order to get the best performance. As discussed earlier, higher number of trees reduces the generalization error and avoids overfitting of data. Additionally, lowering the number of input variables to be used at each node reduces the strength of individual trees hence decreases the correlation between trees and increases classification accuracy.

Bagging

Previous studies has shown that, ensemble of algorithms are more accurate than an individual classifier. Bagging is a classification tree based classifier, also known as bootstrap aggregation. In this classifier, the training data is sampled multiple times with a random distribution. For example, a training data set, with N rows, is randomly sampled and a classification tree is generated. This process is repeated multiple times with different randomly sampled training data sets and the final classifier is the result of the aggregated voted results of individual classifiers. Each training set is based on the randomly selected samples. In bagging, in each trial to create the training data, a set of instances are being replaced by other instances therefore an instance may appear more than once In each trial, the sampling is done with replacement keeping the sample size of the training data as same. In each sample, the set of instances which is replaced is called out-of-bag data while the set of replacing instances is known as in-bag data. The main advantage of this classifier is that it avoids overfitting of the model. The intuition behind this algorithm is the idea that a single classification tree has a fixed choice of training data, however, if the training data is sampled multiple times with replacement (called bootstrapping), and then the classification or regression tree are grown without pruning and results are averaged then higher accuracy can be obtained by reducing the variance in the output error. Also, Brieman noted that bagging allows "poor predictors to turn into worst predictors." Interpretability is the primary disadvantage of this bagging because it requires the averaging of a large number of trees which can't be interpreted individually.

Boosting

Boosting is yet another variant of ensemble decision tree algorithm where appropriate weight is assigned to the training examples or instances of high importance. These weights are repeatedly readjusted for each instances in each trial in order to reduce the bias. All training instances have a

certain impact on the classifier which is determined by its weight. The higher the weight, the more relevant that instance is. Unlike bagging, in boosting, same sample is repeatedly used with improved weights based on the classification accuracy of instances in each iteration. Sum of the votes of each classifier is the result of final classifier.

Data

In this study, we have used the online product text review data from e-commerce website Amazon.com gathered by Wang, Lu and Zhai [1]. It consists of text reviews for six different categories of electronics product reviews. All the analysis have been performed on four sizes of these categories. The raw data scraped by the authors is in JSON file format. Each category is a collection of several files and each file contains one or more reviews. Each review consist of content and ratings, and metadata which includes review ID, author, title and date.

Table-x shows the categories, dataset size and count of positive and negative instances. Here, each review is termed as an instance of the dataset. Each instance consists of an overall product rating between 1 and 5 with 1 as lowest and 5 as the highest. As this is a binary sentiment classification study, we considered only positive and negative reviews. Here, we have made two assumption. First, all reviews with a rating of 1 and 2 are considered negative while the reviews with a rating of 4 and 5 are positive in nature. We have omitted the reviews with rating of 3 because they could be their non-deterministic nature. Second, we have only considered the instances that has a minimum length of 10 terms.

S. No	Category Name	Data Set Sizes	Positive-Negative Counts
1.	Mobile	2K, 8K, 4K, 8K	P=1000, N=1000
2.	Laptop	2K, 8K, 4K, 8K	P=1000, N=1000
3.	Camera	2K, 8K, 4K, 8K	P=1000, N=1000

4.	Tablet	2K, 8K, 4K, 8K	P=1000, N=1000
5.	Video Surveillance	2K, 8K, 4K, 8K	P=1000, N=1000
6.	TV	2K, 8K, 4K, 8K	P=1000, N=1000

Table-1

Table-1 also shows that each dataset consist of equal number of negative and positive reviews. This study involves 24 different datasets of different sizes and categories.

Research Model

As shown in the figure-1, the framework of this project consists of 4 main components as: Data Pre-processing, Feature Engineering, Model Building and Model Evaluation. Next, we will be discussing each component in detail.

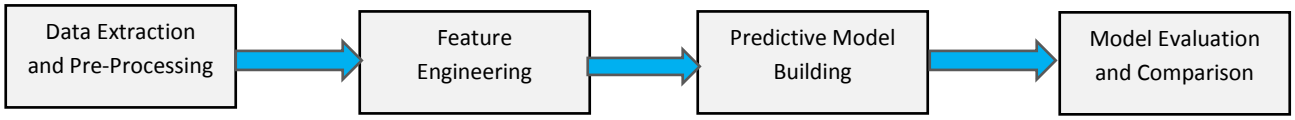

Figure-1

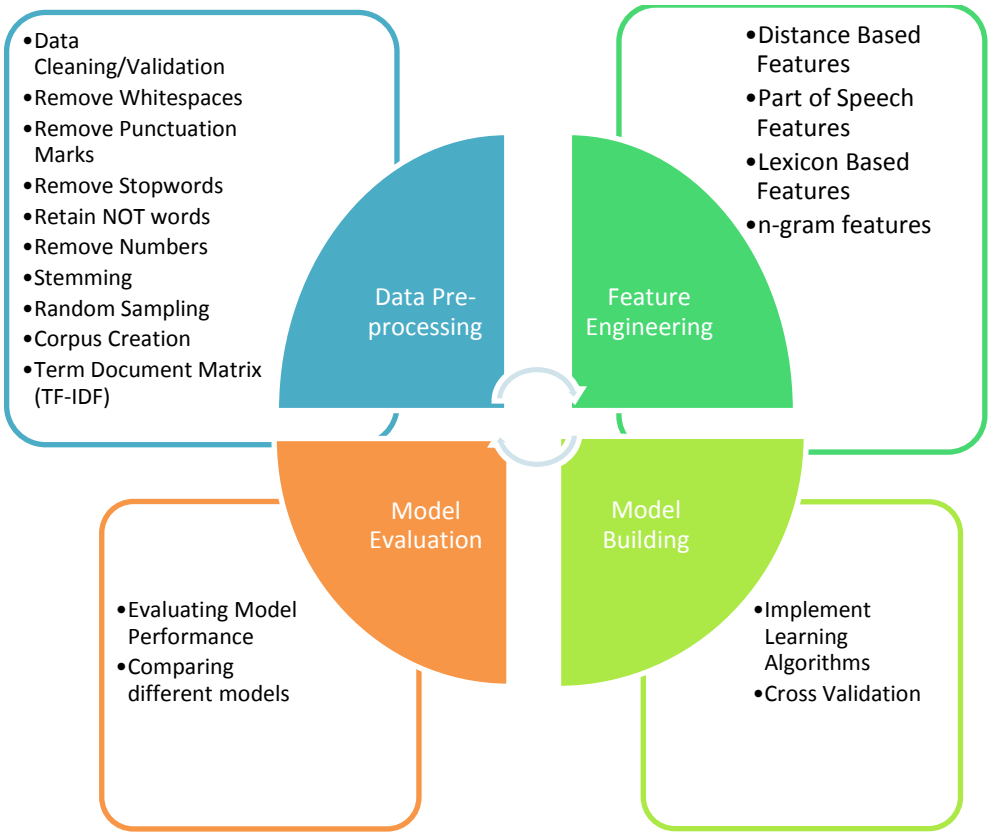

Figure-2

Data Pre-processing

Data pre-processing is one of the key steps in all machine learning studies. It is a required step in order to prepare the dataset which will be used to prepare the features in the next stage and further consumed by the learning algorithms for the prediction purpose. Data serve as the fuel in a predictive model.

In NLP, there are several pre-processing techniques used in order to remove unwanted data elements and get data in the desired shape and format. Pre-processing involves several kind of data manipulation operations which are performed on each instance of the data. We have performed the standard operations on every dataset for consistency which are commonly used in sentiment analysis studies. Additionally, we have used some custom data operations such as retaining negative words like “not”, “haven’t”. These data operations are discussed briefly below. Figure-5 shows some of the key transformations.

a. **UTF-8:**

In this project, we have considered only the reviews of English language. This operation removes all the unwanted characters and symbols which are not part of UTF-8 ASCII character dataset. This is an important part of data cleaning activity.

b. **Lower Case:**
   This operation converts all the characters to the lower-case. Therefore, words with similar spelling will be identical in nature.
c. **Remove Numbers:**
   This operation removes the numbers from the data.
d. **Remove Punctuations:**
   This operation removes all punctuation symbols from the data.
e. **Remove Stop-words:**
   Stop word removal is an important step of pre-processing. Here we
f. **Remove Whitespaces:**
   This operation removes the unnecessary white spaces from the data. This may include leading and trailing spaces in the text as well.
g. **Retain NOT Words:**
   In this project, we decided to retain the negation words such as not, never etc.
h. **Stemming:**
   Stemming is performed to keep the stem of words such as learn is the stem of learning and learn

Figure-3 shows all the element of data pre-processing

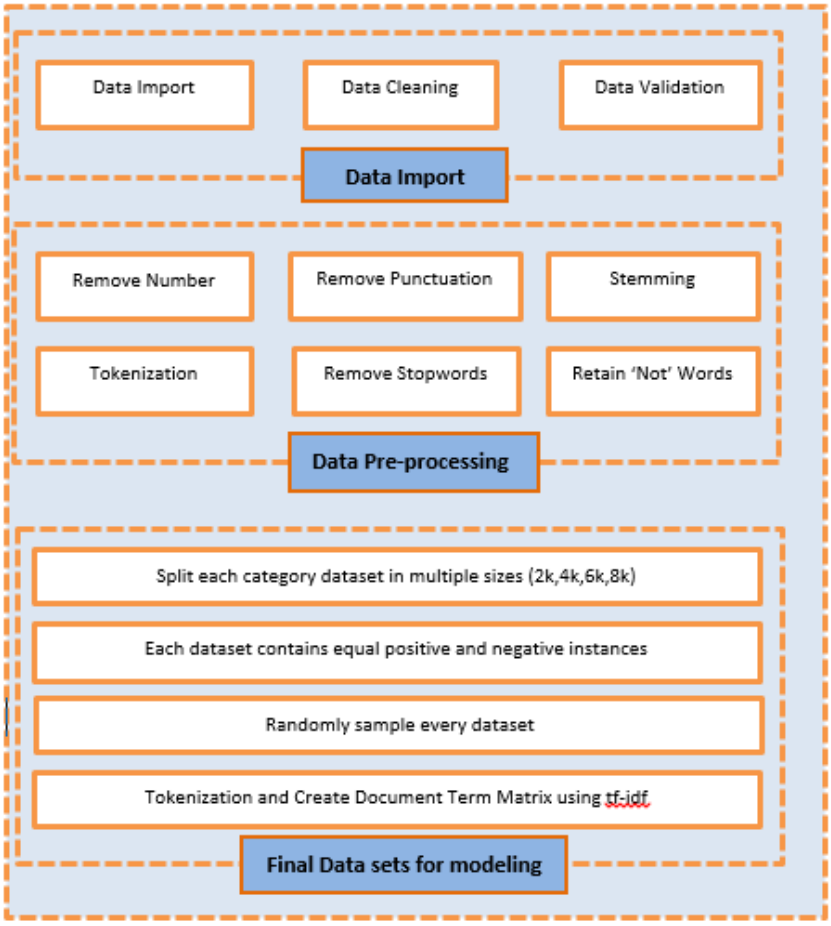

Figure-3

Figure-4 shows all the categories of text data used in this project and their sizes.

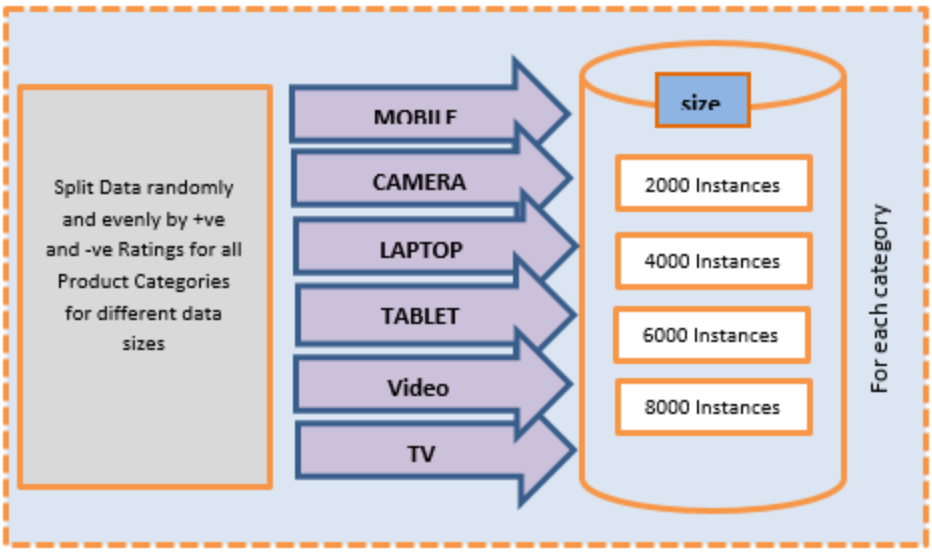

Figure-4

Standard Pre-processing Function	Description
toLower	Converting all the words of corpus to lower-case
removeNumber	removing number from the coprus
retain_not_words	retaining words containing not
removePunctuation	removing punctuation marks from the coprus
removeWhiteSpaces	removing white spaces from the coprus
removeControlCharacters	removing control character from the corpus

Figure-5

- **Feature Engineering**

Feature can be defined as a relevant attributes of the data which helps in predicting the outcome. Features are also called predictors. Feature engineering is the process of creating new features. In text mining, there are two basic types of features involved as semantic and syntactic.

Semantic Features: Features in which polarity of words is taken into account.

Syntactic Features: Features in which structure and form of sentence is taken into account.

Figure-6 shows different eight set of features created in this project. These feature sets is a combination of semantic and syntactic features.

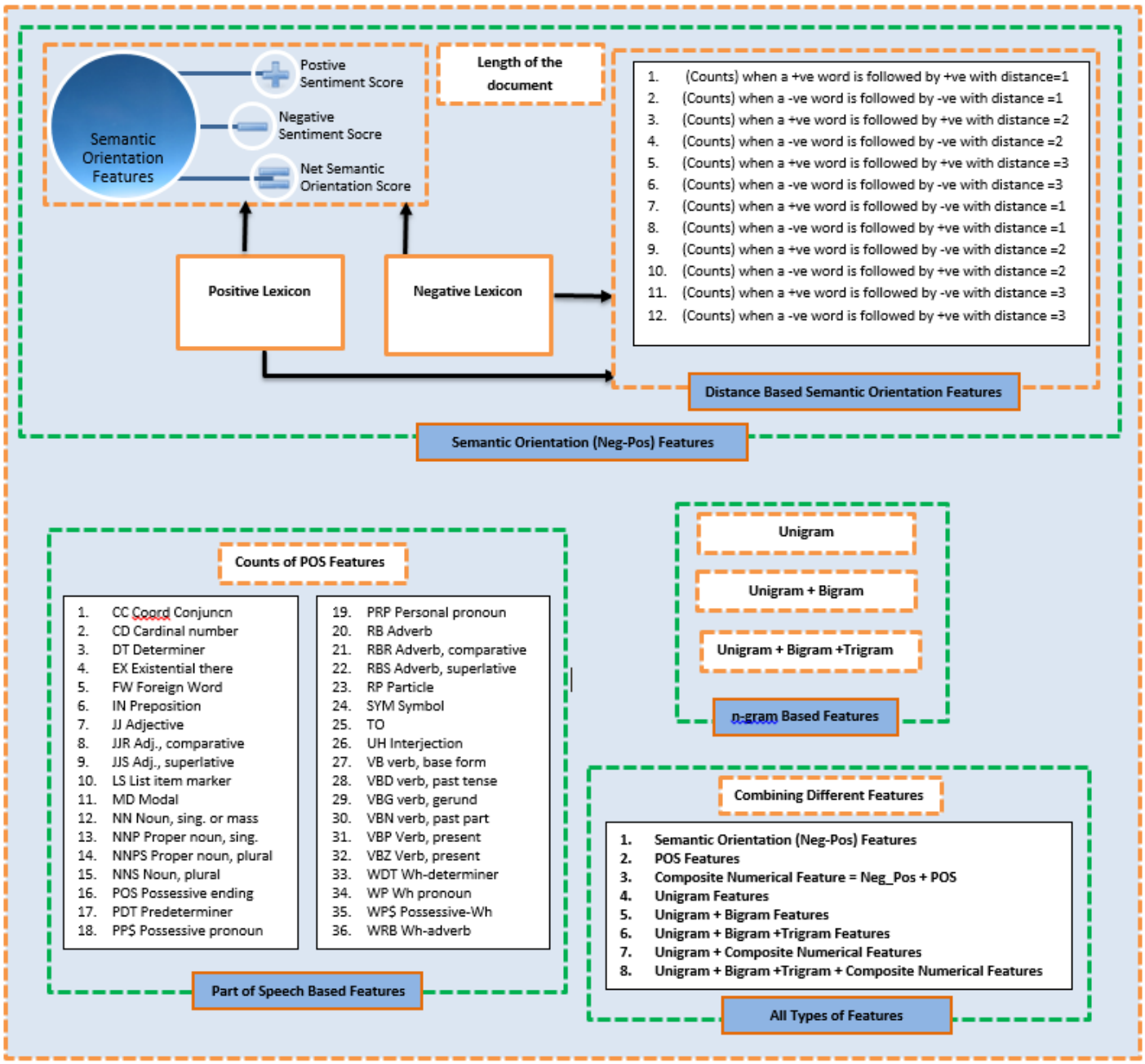

Figure-6

Semantic Features

Table below shows the features created based on the semantic orientation of words. Features such as count of positive words, negative words and net semantic orientation scores have been used in the early research on Sentiment Analysis.

Distance Based Features

Table-2 below also contains distance based features based on the semantic orientation of words. For instance, a negative word followed by negative word has a distance of one.

Example: “I do not like this card”

In the example above, a negative word (not) is followed by a positive word (like) therefore the senti_ctr_neg_pos feature will have a value of 1.

Example: “This phone has best camera yet worst device overall.”

In the example above, a positive word (not) is followed by a negative word (like) after 2 non-polar words therefore the senti_ctr_pos_neg feature will have a value of 1.

Feature name	Description	Type
Positive_word_score	Total Number of Positive Words in the document	Continuous
Negative_word_score	Total Number of Negative Words in the document	Continuous
Net_SO_Score	Net Semantic-Orientation Score (It is sum of first two features)	Continuous
Length_of_String	Total Number of words in the text document	Continuous
senti_ctr_pos_pos_1	Number of occurences in a text document when a positive word is followed by a positive word immediately	Continuous
senti_ctr_neg_neg_1	Number of occurences in a text document when a negative word is followed by a negative word immediately	Continuous
senti_ctr_pos_pos_2	Number of occurences in a text document when a positive word is followed by a positive word within a distance of two	Continuous
senti_ctr_neg_neg_2	Number of occurences in a text document when a negative word is followed by a negative word immediately within a distance of two	Continuous
senti_ctr_pos_pos_3	Number of occurences in a text document when a positive word is followed by a positive word within a distance of three	Continuous
senti_ctr_neg_neg_3	Number of occurences in a text document when a negative word is followed by a negative word immediately within a distance of three	Continuous
senti_ctr_pos_neg_1	Number of occurences in a text document when a positive word is followed by a negative word immediately	Continuous
senti_ctr_neg_pos_1	Number of occurences in a text document when a negative word is followed by a positive word immediately	Continuous
senti_ctr_pos_neg_2	Number of occurences in a text document when a positive word is followed by a negative word within a distance of two	Continuous
senti_ctr_neg_pos_2	Number of occurences in a text document when a negative word is followed by a positive word immediately within a distance of two	Continuous
senti_ctr_pos_neg_3	Number of occurences in a text document when a positive word is followed by a negative word within a distance of three	Continuous
senti_ctr_neg_pos_3	Number of occurences in a text document when a negative word is followed by a positive word immediately within a distance of three	Continuous

Table-2

n-gram

Table-2 below shows the list of n-gram based features. In unigram features, each word is considered as a feature. Similarly, in bigram an trigram, set of two and three words respectively are treated as features.

Feature name	Description	Type
unigram	Unigram features based on weighted tf-idf algorithm	Continuous
unigram + bigram	Unigram and bigram features of a document based on weighted tf-idf algorithm	Continuous
unigram + bigram + trigram	Unigram, bigram and trigram features of a document based on weighted tf-idf algorithm	Continuous

Table-3

Part of Speech (POS)

Table-4 below shows the part of speech based features. It is basically count of each POS in the text instances.

Feature name	Description	Type
CC	Count of Coordinating conjunction	Continuous
CD	Count of Cardinal number	Continuous
DT	Count of Determiner	Continuous
EX	Count of Existential there	Continuous
FW	Count of Foreign word	Continuous
IN	Count of Preposition or subordinating conjunction	Continuous
JJ	Count of Adjective	Continuous
JJR	Count of Adjective, comparative	Continuous
JJS	Count of Adjective, superlative	Continuous
LS	Count of List item marker	Continuous
MD	Count of Modal	Continuous
NN	Count of Noun, singular or mass	Continuous
NNS	Count of Noun, plural	Continuous
NNP	Count of Proper noun, singular	Continuous
NNPS	Count of Proper noun, plural	Continuous
PDT	Count of Predeterminer	Continuous
POS	Count of Possessive ending	Continuous
PRP	Count of Personal pronoun	Continuous
PRP$	Count of Possessive pronoun	Continuous
RB	Count of Adverb	Continuous
RBR	Count of Adverb, comparative	Continuous
RBS	Count of Adverb, superlative	Continuous
RP	Count of Particle	Continuous
SYM	Count of Symbol	Continuous
TO	Count of to	Continuous
UH	Count of Interjection	Continuous
VB	Count of Verb, base form	Continuous
VBD	Count of Verb, past tense	Continuous
VBG	Count of Verb, gerund or present participle	Continuous
VBN	Count of Verb, past participle	Continuous
VBP	Count of Verb, non-3rd person singular present	Continuous
VBZ	Count of Verb, 3rd person singular present	Continuous
WDT	Count of Wh-determiner	Continuous
WP	Count of Wh-pronoun	Continuous
WP$	Count of Possessive wh-pronoun	Continuous
WRB	Count of Wh-adverb	Continuous

Table-4

Feature Set

Table-5 below shows the eight final feature sets which have been used in training the models.

Feature name	Description	Type
Unigram	Unigram Features only	Continuous
Unigram_Bigram	Unigram and Bigram Features	Continuous
unigram_bigram_trigram	Unigram, bigram and trigram Features	Continuous
unigram_composite	A combination of Unigram features, part-of-speech count feaures and Lexicon Based Sentiment Polarity and Distance-Based Shifting Features	Continuous
unigram_bigram_trigram_composite	A combination of Unigram features, bigram features, trigram feature, part-of-speech count feaures and Lexicon Based Sentiment Polarity and Distance-Based Shifting Features	Continuous
Neg_Pos	Lexicon Based Sentiment Polarity and Distance-Based Shifting Features	Continuous
Composite_Numerical	A combination of part-of-speech count feaures and Lexicon Based Sentiment Polarity and Distance-Based Shifting Features	Continuous
Part_Of_Speech	Part-Of-Speech Count Features	Continuous

Table-5

Model Building

Model building is the process in which a learning model is prepared on the data containing selected and extracted features. The entire dataset is split into two part as test and train. The ratio of train and test can varied as per the requirement but is usually kept as 70 to 30. The model is learned using a supervised algorithm on test dataset. Model's accuracy is evaluated using test dataset.

Another popular model validation technique is k-fold Cross-validation which is widely used to avoid the over-fitting problem. In this technique, the data is randomly sampled and partitioned into k subsamples with k-1 subsamples as training set and one subsample as validation set. This process is repeated until all k-folds are treated as validation dataset. In this project, we have used 10-fold cross-validation.

Since this study is highly computation intensive, the models were built using Amazon Web Services' (AWS) cloud computing technology called Elastic Compute Cloud (EC2).

Table-6 shows the hyper-parameters setting of all algorithms which were usually kept to their default settings. Hyper-parameter tuning is a complex and time-consuming task and beyond the scope of this project.

Figure-7 summarizes the model building. As discussed in the data section above, there are 6 categories of data set were tested and each dataset has 4 sizes. Furthermore, there are 8 feature sets. Therefore, 24 datasets were used by 9 algorithms on 8 feature sets hence **1728** models were built.

Algorithm	Hyper-parameter
Naïve Bayes	Default
SVM	Kernel= "Radial"
Neural Net	Hidden Layer=1
Random Forest	Mtry= square root of the number of features ; Number of trees= 300;
Boosting	Default
Bagging	Default
SLDA	Default
Tree	200
Maximum Entropy	Default

Table-6

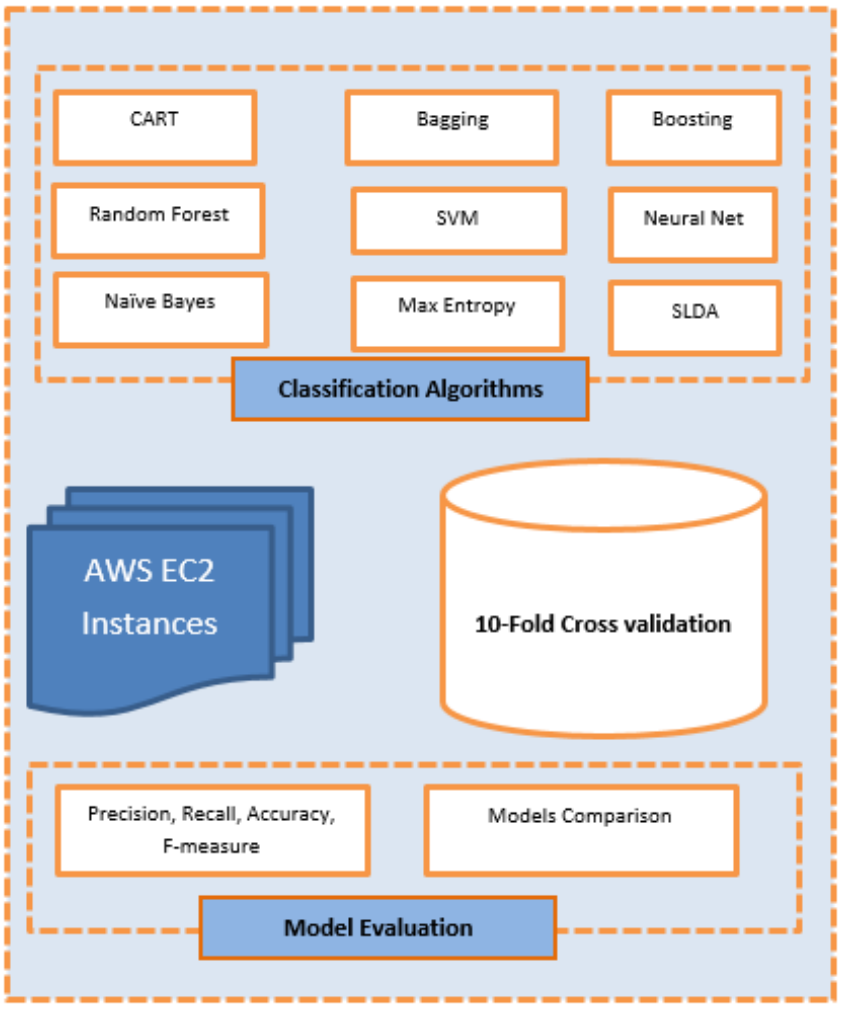

Figure-7

Model Evaluation

Model evaluation is the last stage of this research model. It is required in order to evaluate the performance of different algorithms. Prediction of classification problems are primarily evaluated by four key measures as discussed below.

Precision

Precision is the ratio of correctly predicted positive values to the total positive predicted values. It is also known as positive predicted value.

Recall

Recall also known as sensitivity is the ration of correctly predicted positive values to the actual total positive values in the data.

Accuracy

Accuracy is the most important performance metric of model evaluation in binary classification task. It is a measure of correctly predicted values to the total population.

F-measure

F-measure also known as F-score is the harmonic mean of recall and precision and is widely used in the binary classification problems to evaluate the performance of model. Higher the F-score, better the model.

Experiment Results

Performance of models has been primarily evaluated based on their F-score and accuracy as discussed in the previous section. Table- x shows the model evaluation results for Camera dataset with size 2k. Results for all datasets are available in the file provided with this report.

The performance of models can be analyzed in 4 way as discussed below:

1. By Feature-sets
2. By Algorithms
3. By Dataset-size
4. By Category

In this project, the first finding is that Unigram-Bigram- Trigram-Composite feature with Maximum Entropy has achieved highest accuracy of 96.367% among all other features in all categories of dataset. Secondly, Boosting algorithm has achieved highest accuracy and F-score with composite-numerical features as shown in figure. Moreover, Neural net has performed worst in all cases because of only 1 hidden layer. As expected, higher accuracy is achieved as the size of the dataset is increased in all the

categories. Also, all algorithms except Neural Net and Naïve Bayes have achieved accuracy of more than 80% across all categories and dataset sizes. It is also observed that recall and precision has improved as the size of dataset is increased in 4 out of 6 categories

Here, we have discussed the results of only one category. Please find all other results in the tableau dashboard available here.

Table-7 below shows the mean accuracy of all features for Camera category for 2k dataset and for all algorithms.

meanAccuracy	BG	BOOSTING	MX	NaiveBaye	NNET	RF	SLDA	SVM	TREE
Composite_Numerical	0.820511	0.879473	0.828234	0.623398	0.499406	0.831092	0.825427	0.591118	0.821409
Neg_Pos	0.799706	0.878372	0.816338	0.708019	0.499406	0.819159	0.788288	0.565753	0.819513
Part_Of_Speech	0.658993	0.883014	0.701098	0.536782	0.499406	0.677314	0.629635	0.622064	0.634184
Unigram	0.787779	0.894699	0.957904	0.792108	0.500239	0.857459	0.838966	0.830273	0.727286
Unigram_Bigram	0.792794	0.903564	0.957538	0.815976	0.500636	0.863603	0.853934	0.839983	0.731674
Unigram_Bigram_Trigram	0.799278	0.891339	0.957595	0.823255	0.499416	0.862272	0.850216	0.840287	0.735759
Unigram_Bigram_Trigram_Composite	0.844742	0.886308	0.964962	0.818225	0.499611	0.867832	0.817295	0.637876	0.825285
Unigram_Composite	0.839207	0.888187	0.963527	0.791075	0.499611	0.866177	0.724086	0.697378	0.825285

Table-7

Table-8 below shows the mean F-score of all features for Camera category for 2k dataset and foi all algorithms.

meanFscore	BG	BOOSTING	MX	NaiveBaye	NNET	RF	SLDA	SVM	TREE
Composite_Numerical	0.821556	0.850997	0.827002	0.700191	0.666245	0.829583	0.818163	0.677898	0.818549
Neg_Pos	0.799953	0.862461	0.814553	0.750145	0.666245	0.821867	0.788002	0.66129	0.81125
Part_Of_Speech	0.656651	0.666018	0.700799	0.627579	0.666245	0.67345	0.606347	0.66755	0.625918
Unigram	0.789818	0.850802	0.957333	0.782378	0.665813	0.854748	0.832921	0.827362	0.750514
Unigram_Bigram	0.79297	0.85935	0.958186	0.801846	0.666145	0.864133	0.84911	0.841024	0.752407
Unigram_Bigram_Trigram	0.80156	0.851417	0.956629	0.809443	0.663074	0.860683	0.844845	0.84166	0.756064
Unigram_Bigram_Trigram_Composite	0.845477	0.863962	0.96477	0.804839	0.665404	0.866601	0.809945	0.677359	0.827994
Unigram_Composite	0.840611	0.869957	0.964332	0.782233	0.665404	0.865374	0.716423	0.714862	0.827994

Table-8

Table-9 below shows the mean precision of all features for Camera category for 2k dataset and foi all algorithms.

meanPrecision	BG	BOOSTING	MX	NaiveBaye	NNET	RF	SLDA	SVM	TREE
Composite_Numerical	0.815	0.845	0.832	0.587	0.5	0.828	0.843	0.559	0.831
Neg_Pos	0.797	0.84	0.82	0.655	0.5	0.812	0.801	0.541	0.846
Part_Of_Speech	0.659	0.692	0.701	0.544	0.5	0.68	0.649	0.592	0.639
Unigram	0.779	0.886	0.955	0.81	0.5	0.871	0.869	0.835	0.694
Unigram_Bigram	0.793	0.903	0.952	0.869	0.5	0.876	0.883	0.84	0.696
Unigram_Bigram_Trigram	0.792	0.879	0.95	0.873	0.497	0.87	0.878	0.842	0.696
Unigram_Bigram_Trigram_Composite	0.837	0.863	0.967	0.868	0.5	0.872	0.84	0.609	0.809
Unigram_Composite	0.83	0.865	0.965	0.813	0.5	0.867	0.736	0.67	0.809

Table-9

Table-10 below shows the mean recall of all features for Camera category for 2k dataset and for all algorithms.

meanRecall	BG	BOOSTING	MX	NaiveBaye	NNET	RF	SLDA	SVM	TREE
Composite_Numerical	0.829	0.858	0.823	0.883	1	0.832	0.796	0.863	0.809
Neg_Pos	0.805	0.887	0.81	0.879	1	0.833	0.778	0.854	0.78
Part_Of_Speech	0.655	0.651	0.701	0.823	1	0.668	0.572	0.767	0.614
Unigram	0.803	0.821	0.96	0.758	1	0.84	0.801	0.821	0.818
Unigram_Bigram	0.794	0.823	0.965	0.745	1	0.854	0.819	0.845	0.82
Unigram_Bigram_Trigram	0.815	0.828	0.964	0.758	1	0.854	0.817	0.844	0.829
Unigram_Bigram_Trigram_Composite	0.856	0.867	0.963	0.753	1	0.864	0.784	0.766	0.85
Unigram_Composite	0.853	0.878	0.964	0.76	1	0.866	0.702	0.77	0.85

Table-10

Classification Evaluation by Algorithm

By Category and Size 2k

Camera:

In the figure-8 and figure-9 below, we can clearly observed that maximum entropy is the clear winner among algorithms and unigram_bigram_trigram_composite is the best set of feature with highest accuracy and F-score.

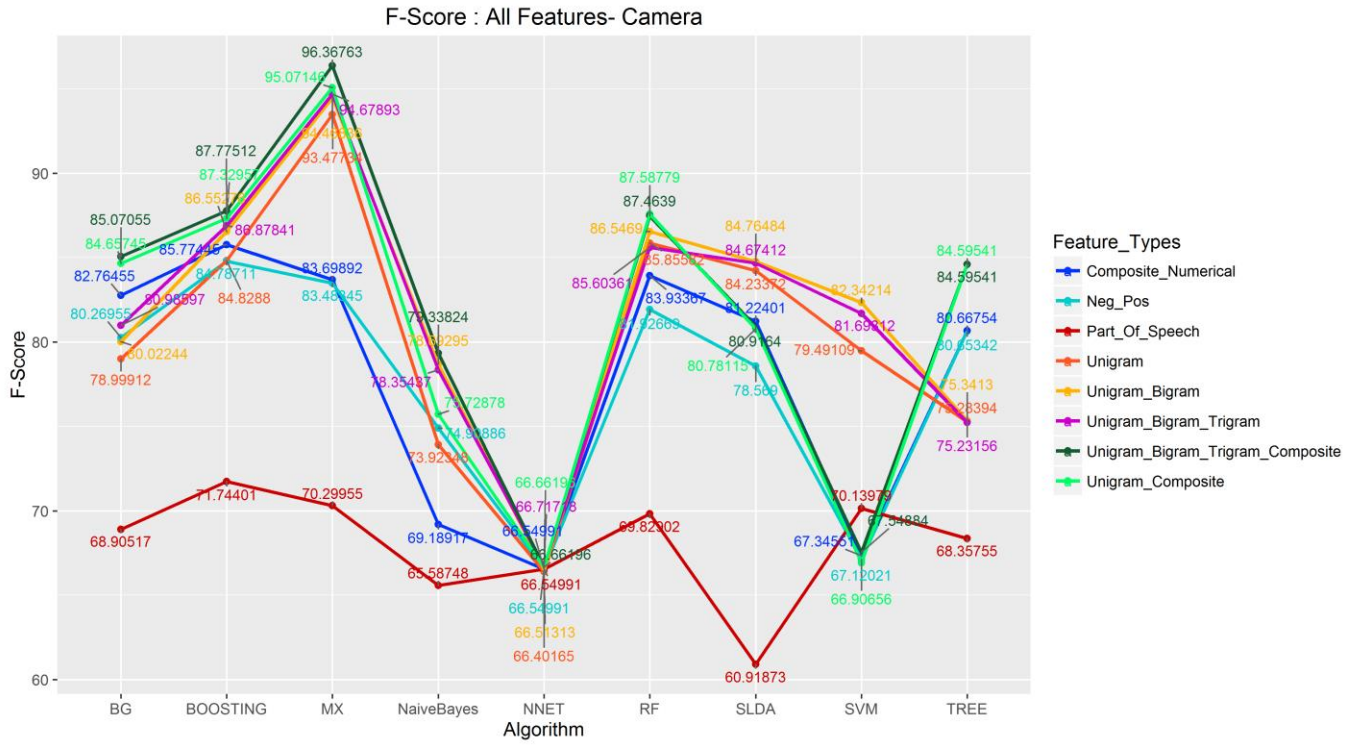

Figure-8

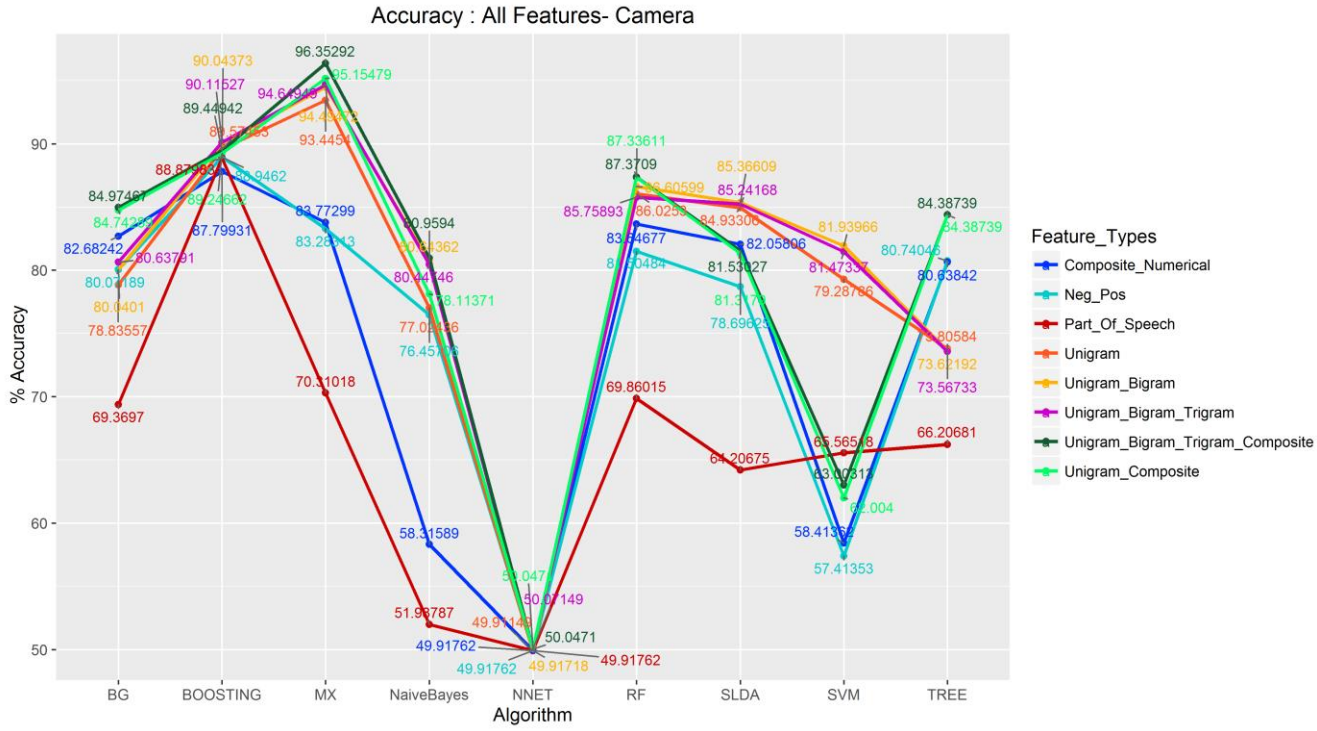

Figure-9

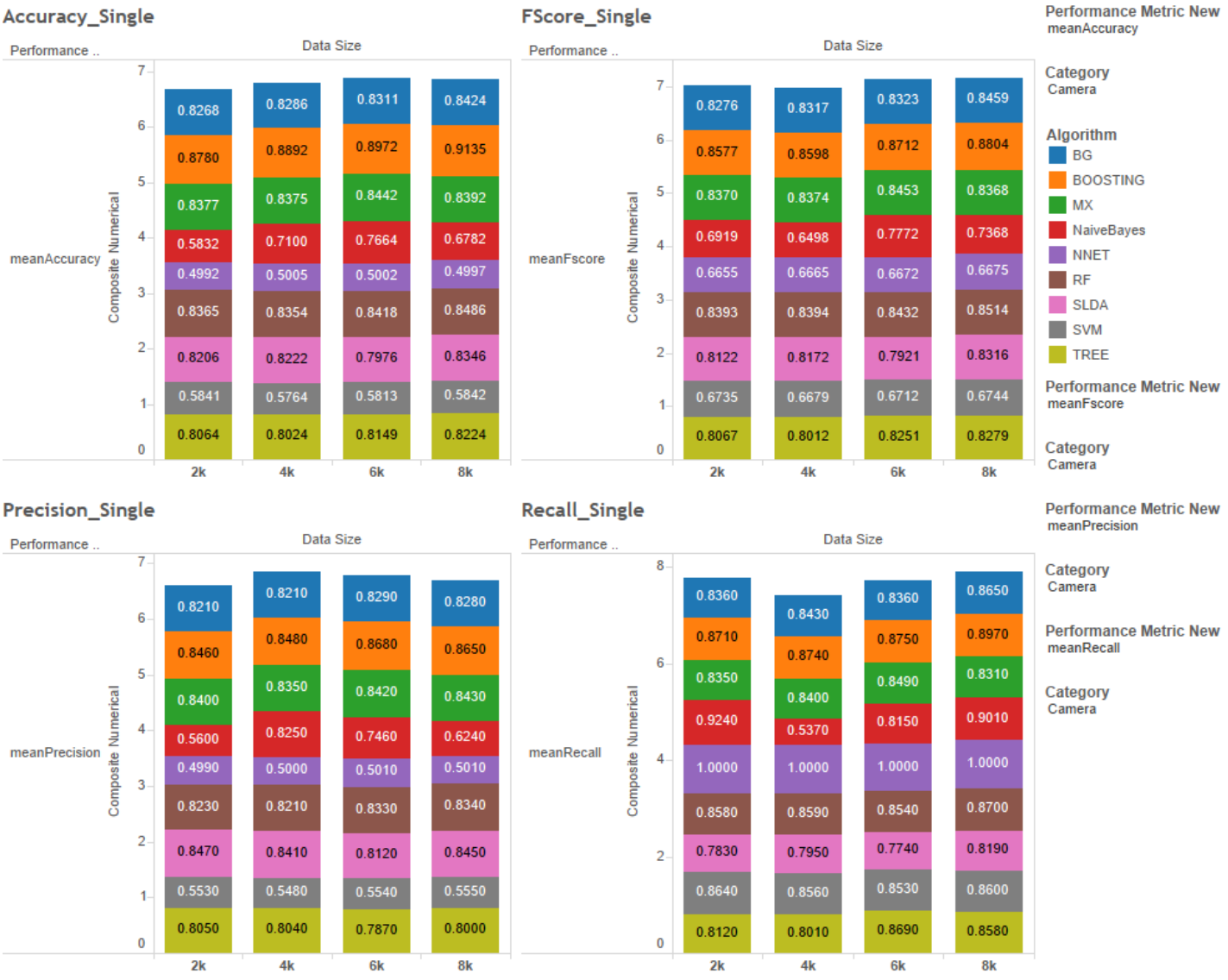

Figure-10

Reference:

Taboada, Maite, et al. "Lexicon-based methods for sentiment analysis." *Computational linguistics* 37.2 (2011): 267-307.

Pang, Bo, and Lillian Lee. "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts." *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2004.

Pang, Bo, and Lillian Lee. "Opinion mining and sentiment analysis." *Foundations and trends in information retrieval* 2.1-2 (2008): 1-135.

Dash, Manoranjan, and Huan Liu. "Feature selection for classification." *Intelligent data analysis* 1.3 (1997): 131-156.

Riloff, Ellen, Janyce Wiebe, and Theresa Wilson. "Learning subjective nouns using extraction pattern bootstrapping." *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, 2003.

Gamon, Michael. "Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis." *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, 2004.

Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan. "Thumbs up?: sentiment classification using machine learning techniques." *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. Association for Computational Linguistics, 2002.

Turney, Peter D. "Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews." *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002.

Yi, Jeonghee, et al. "Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques." *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*. IEEE, 2003.

Lowd, Daniel, and Pedro Domingos. "Naive Bayes models for probability estimation." *Proceedings of the 22nd international conference on Machine learning*. ACM, 2005.

Beineke, Philip, Trevor Hastie, and Shivakumar Vaithyanathan. "The sentimental factor: Improving review classification via human-provided information." *Proceedings of the 42nd annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2004.

Meyer, David, Kurt Hornik, and Ingo Feinerer. "Text mining infrastructure in R." *Journal of statistical software* 25.5 (2008): 1-54.

Tong, Simon, and Daphne Koller. "Support vector machine active learning with applications to text classification." *Journal of machine learning research*2.Nov (2001): 45-66.

Lee, Yuh-Jeng. "Smooth Support Vector Machines." *Preliminary Thesis Proposal Computer Sciences Department University of Wisconsin* (2000).

Berger, Adam L., Vincent J. Della Pietra, and Stephen A. Della Pietra. "A maximum entropy approach to natural language processing." *Computational linguistics* 22.1 (1996): 39-71.

Ratnaparkhi, Adwait. "A maximum entropy model for part-of-speech tagging." *Proceedings of the conference on empirical methods in natural language processing*. Vol. 1. 1996.

Loh, Wei-Yin. "Classification and regression tree methods." *Encyclopedia of statistics in quality and reliability* (2008).

Rish, Irina. "An empirical study of the naive Bayes classifier." *IJCAI 2001 workshop on empirical methods in artificial intelligence*. Vol. 3. No. 22. IBM New York, 2001.

Xu, Min, et al. "Decision tree regression for soft classification of remote sensing data." *Remote Sensing of Environment* 97.3 (2005): 322-336.

Quinlan, J. Ross. "Bagging, boosting, and C4. 5." *AAAI/IAAI, Vol. 1*. 1996.

Maclin, Richard, and David Opitz. "An empirical evaluation of bagging and boosting." *AAAI/IAAI* 1997 (1997): 546-551.

Prasad, Anantha M., Louis R. Iverson, and Andy Liaw. "Newer classification and regression tree techniques: bagging and random forests for ecological prediction." *Ecosystems* 9.2 (2006): 181-199.

Caruana, Rich, and Alexandru Niculescu-Mizil. "An empirical comparison of supervised learning algorithms." *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006.

Rodriguez-Galiano, Victor Francisco, et al. "An assessment of the effectiveness of a random forest classifier for land-cover classification." *ISPRS Journal of Photogrammetry and Remote Sensing* 67 (2012): 93-104.

Cutler, D. Richard, et al. "Random forests for classification in ecology." *Ecology* 88.11 (2007): 2783-2792.

Liaw, Andy, and Matthew Wiener. "Classification and regression by randomForest." *R news* 2.3 (2002): 18-22.

Breiman, Leo. "Random forests." *Machine learning* 45.1 (2001): 5-32.

Loh, Wei-Yin. "Classification and regression tree methods." Encyclopedia of statistics in quality and reliability (2008).

Pal, Mahesh, and Paul M. Mather. "Decision tree based classification of remotely sensed data." Paper presented at the 22nd Asian Conference on Remote Sensing. Vol. 5. 2001.

Hongning Wang, Yue Lu and ChengXiang Zhai. Latent Aspect Rating Analysis without Aspect Keyword Supervision. The 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'2011), P618-626, 2011.

Agarwal, Apoorv, et al. "Sentiment analysis of twitter data." Proceedings of the workshop on languages in social media. Association for Computational Linguistics, 2011.

Wilson, Theresa, Janyce Wiebe, and Paul Hoffmann. "Recognizing contextual polarity in phrase-level sentiment analysis." Proceedings of the conference on human language technology and empirical methods in natural language processing. Association for Computational Linguistics, 2005.

Hu, Minqing, and Bing Liu. "Mining and summarizing customer reviews." Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2004.

Barbosa, Luciano, and Junlan Feng. "Robust sentiment detection on twitter from biased and noisy data." Proceedings of the 23rd International Conference on Computational Linguistics: Posters. Association for Computational Linguistics, 2010.

Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan. "Thumbs up?: sentiment classification using machine learning techniques." Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10. Association for Computational Linguistics, 2002.

Kim, Soo-Min, and Eduard Hovy. "Determining the sentiment of opinions." Proceedings of the 20th international conference on Computational Linguistics. Association for Computational Linguistics, 2004.

Abbasi, Ahmed, Hsinchun Chen, and Arab Salem. "Sentiment analysis in multiple languages: Feature selection for opinion classification in Web forums." ACM Transactions on Information Systems (TOIS) 26.3 (2008): 12.

Pang, Bo, and Lillian Lee. "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts." Proceedings of the 42nd annual meeting on Association for Computational Linguistics. Association for Computational Linguistics, 2004.