Opinion Mining and Sentiment Analysis

Submitted in the partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER ENGINEERING

Under Supervision Of:
Mr. Sarfaraz Masood
Assistant Professor
Department of Computer Engg.
Jamia Millia Islamia, New Delhi

Submitted By:
Abhishek Raj Chauhan (10-CSS-02)
Prakhar Dhama (10-CSS-48)

DEPARTMENT OF COMPUTER ENGINEERING
FACULTY OF ENGINEERING AND TECHNOLOGY
JAMIA MILLIA ISLAMIA, NEW DELHI-110025
YEAR: 2013-2014

BONAFIDE CERTIFICATE

Certified that this project report, “Opinion Mining and Sentiment Analysis” is bonafide work of Abhishek Raj Chauhan (10-CSS-02) and Prakhar Dhama (10-CSS-48) in partial fulfilment of Graduate degree of B.Tech Computer Engg. in Jamia Millia Islamia during the year 2013-2014.

The project was carried out under my supervision. The project has not been submitted for the award of any Degree or Diploma as far as my knowledge is concerned.

Mr. Sarfaraz Masood
Assistant Professor
Department of Computer Engg.
Faculty of Engineering and Technology
Jamia Millia Islamia
New Delhi

ACKNOWLEDGEMENT

We are greatly indebted to our supervisor and guide Mr. Sarfaraz for his invaluable, technical guidance, great innovative ideas and overwhelming moral support during the course of the project. We are grateful to our H.O.D. Professor M.N. Doja for his invaluable support through the project.

We are thankful to the Department of Computer Engg. And the entire faculty members especially Professor Md. Sufyan Beg, Mr. Sarfaraz Masood, Mr. Tanvir Ahmad, Mr. Bashir Alam, Md. Amjad, Md. Zeeshan Ansari, Mr. Faiyaz Ahmed, Mr. Jawaharlal, Mr. Shahzad, Mr. Danish Rizvi and Mr. Musheer Ahmad for their teaching, guidance and encouragement. We are also thankful to our classmates and friends for the valuable suggestions for their active support.

Abhishek Raj Chauhan (10-CSS-02)
Prakhar Dhama (10-CSS-48)
Department of Computer Engg.
Jamia Millia Islamia
New Delhi

TABLE OF CONTENTS

ABSTRACT

The objective of this project is to mine opinions which indicate positive or negative sentiments. The task is technically challenging and practically very useful. For example, businesses always want to find public or consumer opinions about their products and services. Potential customers also want to know the opinions of existing users before they use a service or purchase a product. As a further convenience to the customers, we have also tried to identify the product aspects along with their individual attributes, which acts as a summary of the whole review. The algorithm used in this project builds upon the one introduced by [2]. It uses an unsupervised approach and takes the help of Natural Language Processing (NLP) techniques like Part-of-Speech (POS) tagging and Information Theory techniques like Pointwise Mutual Information (PMI). The algorithm has been implemented in C#.NET 4.5 and takes the help of OpenNLP library. Initial testing has revealed the algorithm to be quite accurate (around 80%), although extensive testing is yet to be conducted.

INTRODUCTION

The Web, apart from structured data(typically data records retrieved from underlying databases and displayed in Web pages following some fixed templates), also contains a huge amount of information in **unstructured texts**. Analysing these texts is of great importance as well and perhaps even more important than extracting structured data because of the sheer volume of valuable information of almost any imaginable type contained in text. In this project, we only focus on mining opinions which indicate positive or negative sentiments. The task is technically challenging and practically very useful. For example, businesses always want to find public or consumer opinions about their products and services. Potential customers also want to know the opinions of existing users before they use a service or purchase a product.

This area of study is called **opinion mining or sentiment analysis**. It analyses people's opinions, appraisals, attitudes, and emotions toward entities, individuals, issues, events, topics, and their attributes. Opinions are important because they are key influencers of our behaviours. Our beliefs and perceptions of reality, and the choices we make, are to a considerable degree conditioned on how others see and evaluate the world. For this reason, when we need to make a decision we often seek out the opinions of others. This is true not only for individuals but also for organizations.

With the explosive growth of social media (i.e., reviews, forum discussions, blogs, and social networks) on the Web, individuals and organizations are increasingly using the content in these media for their decision making. Nowadays, if one wants to buy a consumer product, one is no longer limited to asking one's friends and family for opinions as in the past because there are many user reviews of products on the Web. For an organization, it may no longer be necessary to conduct opinion polls, surveys, and focus groups in order to gather public opinions about its products and services because there is an abundance of such information publicly available. However, finding and monitoring opinion sites on the Web and distilling the information contained in them remains a formidable task because of the proliferation of diverse sites. Each site typically contains a huge volume of opinionated text that is not always easily deciphered in long forum postings and blogs. The average human reader will have difficulty identifying relevant sites and accurately summarizing the information and opinions contained in them. Moreover, it is also known that human analysis and evaluation of text information is subject to considerable biases, e.g., people often pay greater attention to opinions that are consistent with their own preferences. People also have difficulty, owing to their mental and physical limitations, producing consistent results when the amount of information to be processed is large. **Automated opinion mining** and **summarization systems** are thus needed, as subjective biases and mental limitations can be overcome with an objective opinion analysis system.

In the past decade, a considerable amount of research has been done in academia. There are also numerous commercial companies that provide opinion mining services. In this project, we first define the opinion mining problem. From the definition, we will see the key technical issues that need to be addressed. We then describe various key mining tasks that have been studied in the research literature and their representative techniques.

1.1 The Problem of Opinion Mining

In this section, we define an abstraction of the opinion mining problem. It enables us to see a structure from the complex and intimidating unstructured text. Moreover, for most opinion-based applications, it is essential to analyse a collection of opinions rather than only one because one opinion represents only the view of a single person, which is usually not sufficient for action. This indicates that some form of summary of opinions is needed. The abstraction should facilitate this summarization.

1.2 Problem Definitions

We use the following review segment on iPhone to introduce the problem (an id number is associated with each sentence for easy reference):

"(1) I bought an iPhone a few days ago. (2) It was such a nice phone. (3) The touch screen was really cool. (4) The voice quality was clear too. (5) However, my mother was mad with me as I did not tell her be- fore I bought it. (6) She also thought the phone was too expensive, and wanted me to return it to the shop. … "

The question is: what we want to mine or extract from this review? The first thing that we notice is that there are several opinions in this review.

Sentences (2), (3), and (4) express some positive opinions, while sentences (5) and (6) express negative opinions or emotions. Then we also notice that the opinions all have some targets. The target of the opinion in sentence (2) is the iPhone as a whole, and the targets of the opinions in sentences (3) and (4) are "touch screen" and "voice quality" of the iPhone, respectively. The target of the opinion in sentence (6) is the price of the iPhone, but the target of the opinion/emotion in sentence (5) is "me", not iPhone. Finally, we may also notice the holders of opinions. The holder of the opinions in sentences (2), (3), and (4) is the author of the review ("I"), but in sentences (5) and (6) it is "my mother." With this example in mind, we now formally define the opinion mining problem. We start with the *opinion target*. In general, opinions can be expressed about anything, e.g., a product, a service, an individual, an organization, an event, or a topic, by any person or organization. We use the term *entity* to denote the target object that has been evaluated. An entity can have a set of components (or parts) and a set of attributes. Each component may have its own sub-components and its set of attributes, and so on. Thus, an entity can be hierarchically decomposed based on the part-of relation. Formally, we have the following:

Definition (entity): An entity e is a product, service, person, event, organization, or topic. It is associated with a pair, e: (T, W), where T is a hierarchy of components (or parts), sub-components, and so on, and W is a set of attributes of e. Each component or sub-component also has its own set of attributes.

Example 1: A particular brand of cellular phone is an entity, e.g., iPhone. It has a set of components, e.g., battery and screen, and also a set of attributes, e.g., voice quality, size, and weight. The battery component also has its own set of attributes, e.g., battery life and battery size.

Based on this definition, an entity can be represented as a tree or hierarchy. The root of the tree is the name of the entity. Each non-root node is a component or sub-component of the entity. Each link is a part-of relation. Each node is associated with a set of attributes. An opinion can be expressed on any node and any attribute of the node.

Example 2: Following Example 1, one can express an opinion on the cellular phone itself (the root node), e.g., "I do not like iPhone," or on any one of its attributes, e.g., "The voice quality of iPhone is lousy." Likewise, one can also express an opinion on any one of the phone's components or any attribute of the component.
In practice, it is often useful to simplify this definition due to two reasons: First, natural language processing is a difficult task. To effectively study the text at an arbitrary level of detail as described in the definition is very hard. Second, for an ordinary user, it is too complex to use a hierarchical representation. Thus, we simplify and flatten the tree to two levels and use the term *aspects* to denote both components and attributes. In the simplified tree, the root level node is still the entity itself, while the second level nodes are the different aspects of the entity.

Definition (aspect): The aspects of an entity e are the components and attributes of e.

Definition (aspect name and aspect expression): An aspect name is the name of an aspect given by the user, while an aspect expression is an actual word or phrase that has appeared in text indicating an aspect.

Example 3: In the cellular phone domain, an aspect could be named voice quality. There are many expressions that can indicate the aspect, e.g., "sound," "voice," and also "voice quality" itself.

Aspect expressions are usually nouns and noun phrases but can also be verbs, verb phrases, adjectives, and adverbs. We call aspect expressions in a sentence that are nouns and noun phrases explicit aspect expressions. For example, "sound" in "The sound of this phone is clear" is an explicit aspect expression. We call aspect expressions of the other types, implicit aspect expressions, as they often imply some aspects. For example, "large" is an implicit aspect expression in "This phone is too large." It implies the aspect size. Many implicit aspect expressions are adjectives and adverbs, which also imply some specific aspects, e.g., expensive (price), and reliably (reliability). Implicit aspect expressions are not just adjectives and adverbs. They can be quite complex, e.g., “This phone will not easily fit in pockets.” Here, “fit in pockets” indicates the aspect size (and/or shape). Like aspects, an entity also has a name and many expressions that indicate the entity. For example, the brand Motorola (entity name) can be ex- pressed in several ways, e.g., “Moto,” “Mot,” and “Motorola.”

Definition (entity name and entity expression): An entity name is the name of an entity given by the user, while an entity expression is an actual word or phrase that has appeared in text indicating an entity.

Definition (opinion holder): The holder of an opinion is the person or organization that expresses the opinion.

An *opinion* (or regular opinion) is simply a positive or negative view, attitude, emotion, or appraisal about an entity or an aspect of the entity from an opinion holder. Positive, negative, and neutral are called opinion orientations. Other names for opinion orientation are sentiment orientation, semantic orientation, or polarity. In practice, neutral is often interpreted as no opinion. We are now ready to formally define an opinion.

Definition (opinion): An opinion (or regular opinion) is a quintuple,

(ei, aij, ooijkl, hk, tl),

where ei is the name of an entity, aij is an aspect of ei, ooijkl is the orientation of the opinion about aspect aij of entity ei, hk is the opinion holder, and tl is the time when the opinion is expressed by hk. The opinion orientation ooijkl can be positive, negative, or neutral or be expressed with different strength/intensity levels. When an opinion is on the entity itself as a while, we use the special aspect GENERAL to denote it.

Some important remarks about this definition are in order:

1. It should be stressed that the five pieces of information in the quintuple must correspond to one another. That is, the opinion ooijkl must be given by opinion holder hk about aspect aij of entity ei at time tl. Otherwise, we may assign an opinion to a wrong entity or wrong aspect, etc.

2. These five components are essential. Without any of them, it can be problematic in general. For example, one says “The picture quality is great,” but if we do not know whose picture quality, the opinion is of little use. However, we do not mean that every piece of information is needed in every application. For example, knowing each opinion holder is not necessary if we want to summarize opinions from a large number of people. Similarly, we do not claim that nothing else can be added to the tuple. For example, in some applications (e.g., marketing), the user may want to know the sex and age of each opinion holder.

3. This definition provides a basis for transforming unstructured text into structured data. The quintuple gives us the essential information for a rich set of qualitative and quantitative analysis of opinions. More specifically, the quintuple is basically a schema/relation for a database table. With a large set of opinion quintuples mined from text, the whole suite of database management systems (DBMS) and OLAP tools can be applied to slice and dice the opinions for all kinds of analyses.

4. Opinions (regular opinions) also have sub-types, e.g., direct opinions and indirect opinions. For direct opinions, opinions are expressed directly on entities or their aspects, e.g., "The voice quality of this phone is great." For indirect opinions, opinions on entities are expressed based on their effects on some other entities. This sub-type often occurs in the medical domain. For example, "After taking this drug, my hand felt much better" describes a desirable effect of the drug on "my hand," which indirectly gives a positive opinion to the drug. For simplicity, we will not distinguish these sub-types here.

5. In the original definition of an entity, it is a hierarchy/tree of components, sub-components, and so on. Every component can have its set of attributes. Due to simplification by flattening the tree, the quintuple representation can result in information loss. For example, "battery" is a component/part of a digital camera. In a camera review, one wrote "The battery for this camera is expensive." This does not say that the camera is expensive (which indicates the aspect price). If one does not care about any attribute of the battery, this sentence just gives a negative opinion to the battery, which is an aspect of the camera entity. How- ever, if one also wants to study opinions about different aspects of the battery, e.g., battery life, price, etc., the battery needs to be treated as a separate entity. The quintuple representation still applies, but the part-of relationship needs to be saved. Of course, conceptually one may also treat the quintuple as a nested relation rather than a flat relation.

We now put everything together to define a model of entity, a model of opinionated document, and the mining objective, which are collectively called the **aspect-based opinion mining**.

Model of entity: An entity ei is represented by itself as a whole and a finite set of aspects, Ai= {ai1, ai2, …, ain}. The entity itself can be expressed with any one of a final set of entity expressions OEi = {oei1, oei2, …, oeis}. Each aspect aij Ai of the entity can be expressed by any one of a finite set of aspect expressions AEij = {aeij1, aeij2, …, aeijm}.

Model of opinionated document: An opinionated document d contains opinions on a set of entities {e1, e2, …, er} from a set of opinion holders {h1, h2, …, hp}. The opinions on each entity ei are expressed on the entity itself and a subset Aid of its aspects.

Objective of opinion mining: Given a collection of opinionated documents D, discover all opinion quintuples (ei, aij, ooijkl, hk, tl) in D.

To achieve this objective, one needs to perform the following tasks:

Task 1 (entity extraction and grouping): Extract all entity expressions in D, and group synonymous entity expressions into entity clusters. Each entity expression cluster indicates a unique entity ei.

Task 2 (aspect extraction and grouping): Extract all aspect expressions of the entities, and group aspect expressions into clusters. Each aspect expression cluster of entity ei indicates a unique aspect aij.

Task 3 (opinion holder and time extraction): Extract these pieces of information from the text or structured data.

Task 4 (aspect sentiment classification): Determine whether each opinion on an aspect is positive, negative or neutral.

Task 5 (opinion quintuple generation): Produce all opinion quintuples (ei, aij, ooijkl, hk, tl) expressed in D based on the results of the above tasks. The difficulty of opinion mining lies in the fact that none of the above problems or tasks is a solved problem. To make matters worse, a sentence may not explicitly mention some pieces of information, but they are implied due to pronouns, language conventions, and contexts. What is also challenging is to ensure that the five pieces of information in an opinion correspond to one another as we discussed earlier. We now use an example blog to illustrate the tasks (a sentence id is associated with each sentence):

Example 4: Posted by: bigXyz on Nov-4-2010: (1) I bought a Motorola phone and my girlfriend bought a Nokia phone yesterday. (2) We called each other when we got home. (3) The voice of my Moto phone was un- clear, but the camera was good. (4) My girlfriend was quite happy with her phone, and its sound quality. (5) I want a phone with good voice quality. (6) So I probably will not keep it.

Task 1 should extract the entity expressions, “Motorola,” “Nokia,” and “Moto,” and group “Motorola” and “Moto” together as they represent the same entity. Task 2 should extract aspect expressions “camera,” “voice,” and “sound” and group “voice” and “sound” together as they are synonyms representing the same aspect. Task 3 should find the holder of the opinions in sentence (3) to be bigXyz (the blog author) and the holder of the opinions in sentence (4) to be bigXyz’s girlfriend. It should also find the time when the blog was posted, which is Nov-4-2010. Task 4 should find that sentence (3) gives a negative opinion to the voice quality of the Motorola phone but a positive opinion to its camera. Sentence (4) gives positive opinions to the Nokia phone as a whole and also its sound quality. Sentence (5) seemingly expresses a positive opinion, but it does not. To generate opinion quintuples for sentence (4), we also need to know what “her phone” is and what “its” refers to. All these are challenging problems. Task 5 should finally generate the following four opinion quintuples:

(Motorola, voice_quality, negative, bigXyz, Nov-4-2010)
(Motorola, camera, positive, bigXyz, Nov-4-2010)
(Nokia, GENERAL, positive, bigXyz's girlfriend, Nov-4-2010)
(Nokia, voice_quality, positive, bigXyz's girlfriend, Nov-4-2010)

1.3 Aspect-Based Opinion Summary

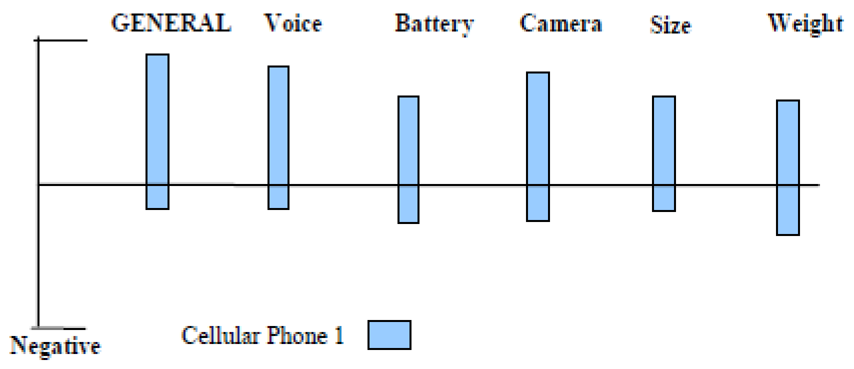
As mentioned previously, most opinion mining applications need to study opinions from a large number of opinion holders. One opinion from a single holder is usually not sufficient for action. This indicates that some form of summary of opinions is needed. Opinion quintuples defined above provide an excellent source of information for generating both qualitative and quantitative summaries. A common form of summary is based on aspects and is called **aspect-based opinion summary** (or feature-based opinion summary). Below, we use an example to illustrate this form of summary, which is widely used in industry.

Example 5: Assume we summarize all the reviews of a particular cellular phone, cellular phone 1. The summary looks like that in Fig. 1.1, and is called a **structured summary**. In the figure, GENERAL represents the phone itself (the entity). 125 reviews expressed positive opinions about the phone and 7 expressed negative opinions. Voice quality and size are two product aspects. 120 reviews expressed positive opinions about the voice quality, and only 8 reviews expressed negative opinions. The <individual review sentences> link points to the specific sentences and/or the whole reviews that give the positive or negative opinions. With such a summary, the user can easily see how existing customers feel about the phone. If he/she is interested in a particular aspect, he/she can drill down by following the <individual review sentences> link to see why existing customers like it and/or dislike it.

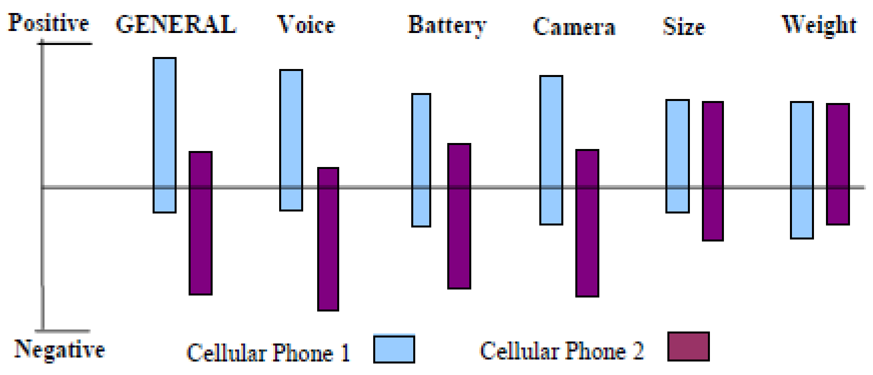
As mentioned earlier, the discovered quintuples can be stored in database tables. Then a whole suite of database and visualization tools can be applied to see the results in all kinds of ways to gain insights of the opinions in structured forms and displayed as bar charts and/or pie charts. For example, the aspect-based summary in Fig. 1.1 can be visualized using the bar chart in Fig. 1.2. In the figure, each bar above the X-axis shows the number of positive opinions on the aspect given at the top. The corresponding bar below the X-axis shows the number of negative opinions on the same aspect. Obviously, other visualizations are also possible. For example, one may only show the percentage of positive opinions.

Cellular phone 1:
Aspect: **GENERAL**
Positive: 125 <individual review sentences>
Negative: 7 <individual review sentences>
Aspect: **Voice quality**
Positive: 120 <individual review sentences>
Negative: 8 <individual review sentences>
Aspect: **Battery**
Positive: 80 <individual review sentences>
Negative: 12 <individual review sentences>
…

Fig. 1.1. An aspect-based opinion summary.



(A) Visualization of aspect-based summary of opinions on a cellular phone



(B) Visual opinion comparison of two cellular phones

Fig. 1.2. Visualization of aspect-based summaries of opinions

TECHNICAL DETAILS

2.1 Sentiment Classification Based on Supervised Learning

Sentiment classification(which classifies an opinion document (e.g., a product review) as expressing a positive or negative opinion or sentiment) can be formulated as a supervised learning problem with three classes, positive, negative, and neutral. Training and testing data used in the existing research are mostly product reviews, which is not surprising due to the above assumption. Since each review already has a reviewer-assigned rating (e.g., 1–5 stars), training and testing data are readily available. For example, a review with 4 or 5 stars is considered a positive review, a review with 1 or 2 stars is considered a negative review and a review with 3 stars is considered a neutral review.

Sentiment classification is similar to but also somewhat different from classic topic-based text classification, which classifies documents into pre- defined topic classes, e.g., politics, sciences, sports, etc. In topic-based classification, topic-related words are important. However, in sentiment classification, *topic-related words are unimportant*. Instead, **opinion words** (also called **sentiment words**) that indicate positive or negative opinions are *important*, e.g., great, excellent, amazing, horrible, bad, worst, etc.

Any existing supervised learning methods can be applied to sentiment classification, e.g., **naïve Bayesian classification**, and **support vector machines (SVM**). Pang et al. took this approach to classify movie reviews into two classes, positive and negative. It was shown that using uni-grams (a bag of individual words) as features in classification performed well with either naïve Bayesian or SVM.

Subsequent research used many more features and techniques in learning. As most machine learning applications, the main task of sentiment classification is to engineer an effective set of features. Some of the example features used in research and possibly in practice are listed below.

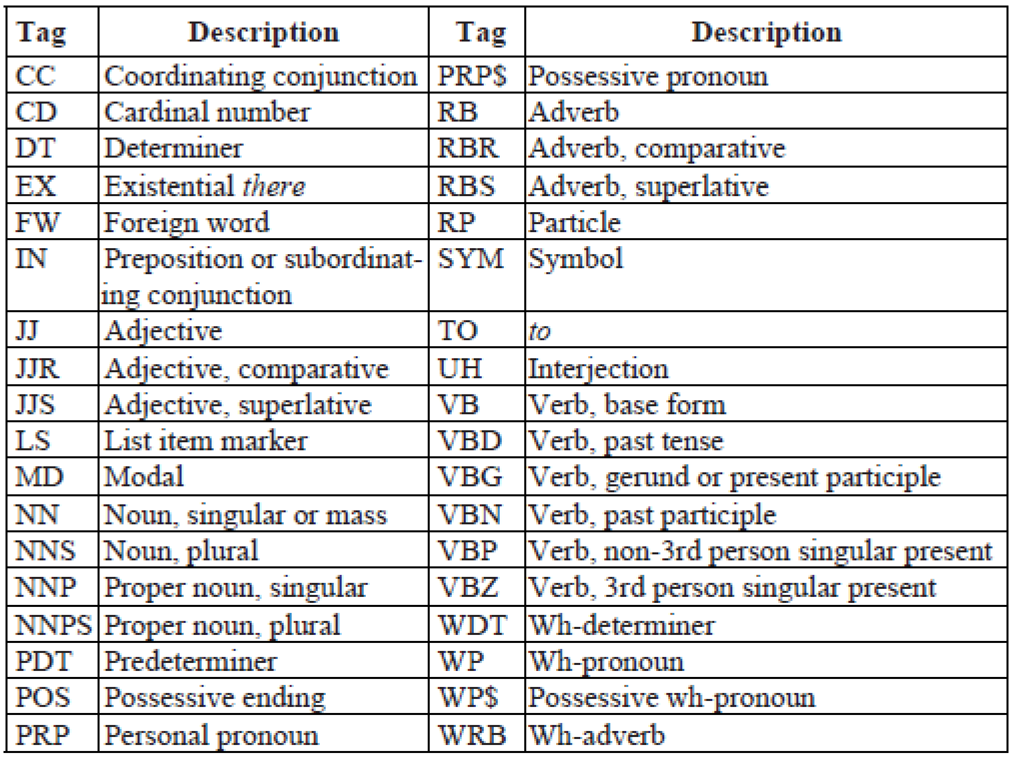
<ins>Terms and their frequency:</ins> These features are individual words or word n- grams and their frequency counts (they are also commonly used in traditional topic-based text classification). In some cases, word positions may also be considered. The TF-IDF weighting scheme from information retrieval may be applied too. These features have been shown quite effective in sentiment classification.

<ins>Part of speech:</ins> It was found in many researches that adjectives are important indicators of opinions. Thus, adjectives have been treated as special features.
Opinion words and phrases: Opinion words are words that are commonly used to express positive or negative sentiments. For example, beautiful, wonderful, good, and amazing are positive opinion words, and bad, poor, and terrible are negative opinion words. Although many opinion words are adjectives and adverbs, nouns (e.g., rubbish, junk, and crap) and verbs (e.g., hate and like) can also indicate opinions. Apart from individual words, there are also opinion phrases and idioms, e.g., cost someone an arm and a leg. Opinion words and phrases are instrumental to sentiment analysis for obvious reasons.

Rules of opinions: Although opinion words and phrases are important, there are also many other expressions that contain no opinion words or phrases but indicate opinions or sentiments.
Negations: Clearly negation words are important because their appearances often change the opinion orientation. For example, the sentence "I don't like this camera" is negative. However, negation words must be handled with care because not all occurrences of such words mean negation. For example, "not" in "not only … but also" does not change the orientation direction.
Syntactic dependency: Words dependency-based features generated from parsing or dependency trees are also tried by several researchers.

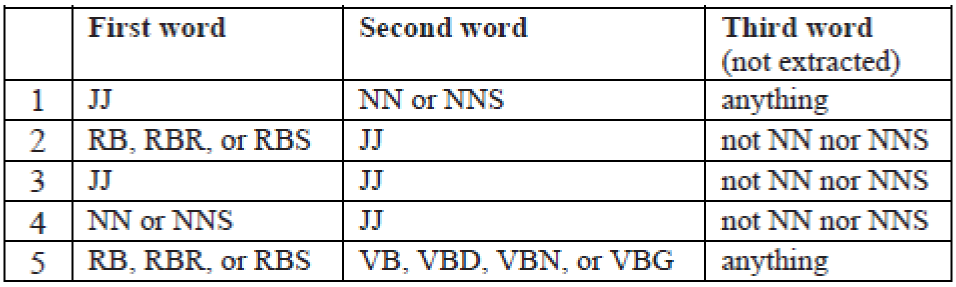
Negations: Clearly negation words are important because their appearances often change the opinion orientation. For example, the sentence "I don't like this camera" is negative. However, negation words must be handled with care because not all occurrences of such words mean negation. For example, "not" in "not only … but also" does not change the orientation direction.

Syntactic dependency: Words dependency-based features generated from parsing or dependency trees are also tried by several researchers.



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

Table 1.1 Penn Treebank Part-Of-Speech (POS) Tags



	First word	Second word	Third word (not extracted)
1	JJ	NN or NNS	anything
2	RB, RBR, or RBS	JJ	not NN nor NNS
3	JJ	JJ	not NN nor NNS
4	NN or NNS	JJ	not NN nor NNS
5	RB, RBR, or RBS	VB, VBD, VBN, or VBG	anything

Table 1.2 Patterns of tags for extracting two words phrases

2.2 Unsupervised Approach to Sentiment Classification

It is not hard to imagine that opinion words and phrases are the dominating indicators for sentiment classification. Thus, using unsupervised learning based on such words and phrases would be quite natural. The given method is such a technique. It performs classification based on some fixed syntactic phrases that are likely to be used to express opinions. The algorithm makes use of a natural language processing technique called **part-of-speech**

(POS) tagging. The part-of-speech of a word is a linguistic category that is defined by its syntactic or morphological behaviour. Common POS categories in English grammar are: noun, verb, adjective, adverb, pronoun, preposition, conjunction, and interjection. Then, there are many categories which arise from different forms of these categories. For example, a verb can be a verb in its base form, in its past tense, etc. In this project, we use the standard **Penn Treebank POS Tags** as shown in Table 1.1. POS tagging is the task of labelling (or tagging) each word in a sentence with its appropriate part of speech. The Penn Treebank site is at http://www.cis.upenn.edu/~treebank/home.html.

ALGORITHM

The algorithm consists of the following four steps:

<u>Step 1</u>

We use 4 opinion lexicons in our algorithm, namely:-

- List of Positive opinion words
- List of Negative opinion words
- List of Context Sensitive opinion words
- List of Useless words

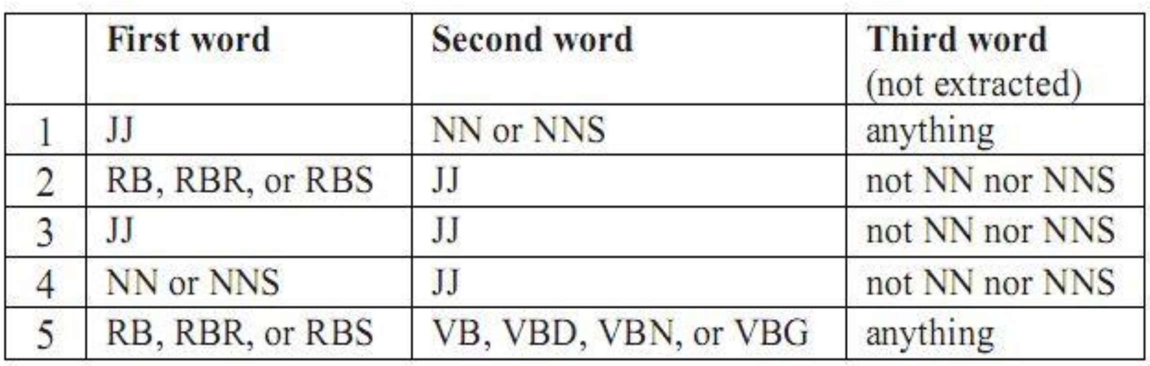
We downloaded the first two lists (containing about 6000 words) from Bing Liu's website while we compiled the remaining two lists ourselves.

<u>Step 2</u>

Perform part of speech (POS) tagging and extract phrases containing adjectives and adverbs based on manually specified patterns as shown in Table 1.2 .This table is an improvement upon the one used in the original algorithm by Turney, the author of [2], which is reproduced here as Table 1.3.

OPINION TARGET PATTERN	**PART-OF-SPEECH PATTERN**
ADJECTIVE+ NOUN+	(JJ(R|S)?)+ (CD* (NN(S|PS|P)?)+)+
ADVERB* ADJECTIVE+	(RB(R|S)?)* (JJ(R|S)?)+
ADVERB+ VERB+	(RB(R|S)?)+ (VB(D|N|G)?)+
ADVERB+ ADJECTIVE+ NOUN+	(RB(R|S)?)+ (JJ(R|S)?)+ (CD* (NN(S|PS|P)?)+))+
NOUN+ VERB ADVERB* ADJECTIVE+	(CD* (NN(S|PS|P)?)+))+ (VB(Z|D)) (RB(R|S)?)* (JJ(R|S)?)+

Table 1.2 Patterns of tags for extracting phrases (Our improved version)



	First word	Second word	Third word (not extracted)
1	JJ	NN or NNS	anything
2	RB, RBR, or RBS	JJ	not NN nor NNS
3	JJ	JJ	not NN nor NNS
4	NN or NNS	JJ	not NN nor NNS
5	RB, RBR, or RBS	VB, VBD, VBN, or VBG	anything

Table 1.3 Patterns of tags for extracting two-word phrases (Original version)

For example, in the sentence “The screen display is gorgeous”, the POS tagging of this sentence will produce: - “The/DT screen/NN display/NN is/VBZ gorgeous/JJ”. According to the last rule in Table 1.2, the consecutive nouns forming ”screen display” will be extracted along with its attribute “gorgeous”.

Step 3

- For each extracted phrase, determine its orientation using the four opinion lexicons as shown.

  If the phrase contains words which are in Useless Words List, ignore the phrase.

  Else if the phrase contains words in Context Sensitive Words List, determine its orientation using PMI as shown in the next point.

  Else if the phrase contains words in Positive Or Negative Words List, mark its orientation as positive or negative respectively.

  Else determine its orientation using PMI

  If the phrase contains a noun, extract its attribute from the remaining phrase and add to that noun’s positive or negative attributes as determined from the orientation of the phrase previously.

- Estimate the orientation of the phrase using the PMI measure given in Equation (1). (PMI is the amount of information that we acquire about the presence of one of the words when we observe the other.)

$$PMI(term_1, term_2) = \log_2\left(\frac{\Pr(term_1 \wedge term_2)}{\Pr(term_1)\Pr(term_2)}\right). \qquad (1)$$

- Here, $\Pr(\text{term}_1 \wedge \text{term}_2)$ is the co-occurrence probability of term_1 and term_2, and $\Pr(\text{term}_1)\Pr(\text{term}_2)$ gives the probability that the two terms co-occur if they are statistically independent.

- The ratio between $\Pr(\text{term}_1 \wedge \text{term}_2)$ and $\Pr(\text{term}_1)\Pr(\text{term}_2)$ is thus a measure of the degree of statistical dependence between them. The log of this ratio is the amount of information that we acquire about the presence of one of the words when we observe the other.

- The semantic/opinion orientation (SO) of a phrase is computed based on its association with the positive reference word “excellent” and its association with the negative reference word “poor”:

$$SO(phrase) = PMI(phrase, \text{“excellent”}) - PMI(phrase, \text{“poor”}). \qquad (2)$$

- The probabilities are calculated by issuing queries to a search engine and collecting the number of hits. For each search query, search engine returns the number of relevant documents to the query, which is the number of hits .
- Thus, by searching the two terms together and separately, we can estimate the probabilities in Equation (1). We can use Google's "AROUND" operator, which lets you specify the maximum number of words that separate the two terms, for this purpose. Let *hits(query)* be the number of hits returned. Equation (2) can be rewritten as:

$$SO(phrase) = \log_2\left(\frac{hits(\text{phrase } NEAR \text{ "excellent"})hits(\text{"poor"})}{hits(\text{phrase } NEAR \text{ "poor"})hits(\text{" excellent"})}\right). \qquad (3)$$

<u>Step 4</u>

- Given a review, the algorithm computes the average semantic/opinion orientation (SO) of all phrases in the review.

  Examples:

– "low fees", "JJ NNS", 0.333	: +ve
– "unethical practices", "JJ NNS", -8.484	: - ve
– "low funds", "JJ NNS", -6.843	: - ve

  The algorithm normalizes the SO to -1.0 to 1.0.

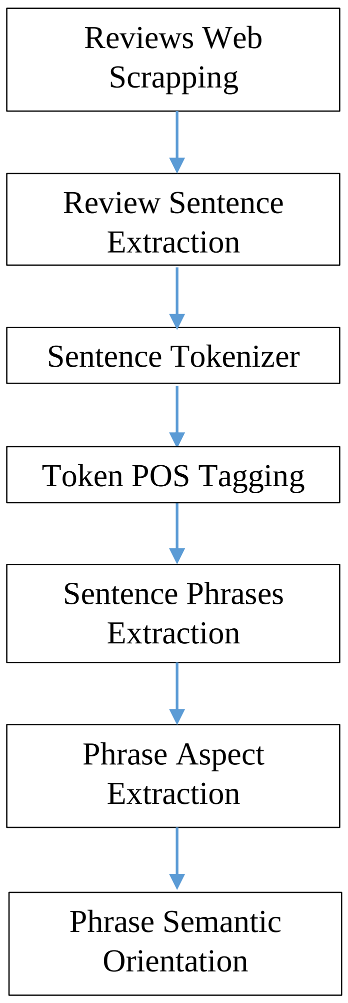
- Finally , the algorithm calculates the Odds that the review is Positive as:-

$$\frac{Total\ Positive\ Score\ of\ all\ positive\ opinion\ phrases}{Total\ Negative\ Score\ of\ all\ negative\ opinion\ phrases}$$

The reciprocal is used to calculate the Negative Odds.
The rating of the review is predicted based on the difference between the Positive Odds and Negative Odds of the review.

IMPLEMENTATION METHODOLOGY

4.1 Flow Chart



Reviews Web Scrapping
Review Sentence Extraction
Sentence Tokenizer
Token POS Tagging
Sentence Phrases Extraction
Phrase Aspect Extraction
Phrase Semantic Orientation

4.2 Reviews Web Scrapping

HTML Agilily Pack is used to parse the webpage. This is an agile HTML parser that builds a read/write DOM and supports plain XPATH or XSLT. It is a .NET code library that allows you to parse "out of the web" HTML files. The parser is very tolerant with "real world" malformed HTML. The object model is very similar to what proposes System.Xml, but for HTML documents (or streams).

HAP is available on http://htmlagilitypack.codeplex.com/

Sample XPATH for extracting review details from https://www.amazon.in/ is as show below:

```
reviewer = doc.DocumentNode.SelectNodes("//span[@class='txtsmall']");
review = doc.DocumentNode.SelectNodes("//div[@class='drkgry']");
title = doc.DocumentNode.SelectNodes("//a[@class='txtlarge gl3 gr4 reviewTitle
valignMiddle']/strong");
rating = doc.DocumentNode.SelectNodes("//div[@class='mt4
ttl']/span[contains(@class,'swSprite s_star')]");
```

Refer to Appendix A.1 for sample code snippet to extract reviews from https://www.amazon.com/, https://www.amazon.in/, https://www.flipkart.in/

4.3 Review Sentence Extraction

After extracting the review from the supplied web page, it is given as input to the Sentence Extractor phase, which detects and outputs the individual sentences in the review. This phase takes the help of the open source OpenNLP (Open Natural Language Processing) Library for this purpose. An interested reader may refer to Appendix A.2 for the actual source code implementing this phase.

4.4 Sentence Tokenizer

Each sentence extracted from the Sentence Extractor is given as input to the Sentence Tokenizer phase, which tokenizes the sentence into separate words. These tokens/words are put in an array of strings to be subsequently used by the next phase (POS Tagger).This phase also uses the OpenNLP Library to fulfil its purpose. The source code for this phase is reproduced in Appendix A.3 for the interested reader.

4.5 Token POS Tagging

The POS (Part-Of-Speech) tagger takes as input the array of tokens of a sentence produced by the previous Sentence Tokenizer phase. With the help of the OpenNLP Library, the most probable POS tag of each token/word of the sentence is determined. The output is in the form of an array of POS tags corresponding to each token in the token array. Appendix A.4 contains the source code implementing this phase.

4.6 Sentence Phrases Extraction

The Sentence Phrase Extraction is the most important phase in the implementation of this project. It takes as input the POS array (produced by Token POS Tagger) and the token array (produced by the Sentence Tokenizer), and extracts phrases from the sentence if and only if they confirm to any of the rules specified in Table 1.2 (shown previously).The phase uses an O(n) algorithm ,which is our original production, to efficiently check if the sentence contains any phrase following the rules in Table 1.2, and if so, add it to the list of phrases. Appendix A.5 contains a code snippet showing a part of this algorithm to give an idea of its implementation to the interested reader.

4.7 Phrase Aspects Extraction

For each phrase in the list of phrases produced by the previous phase, the Aspect Extraction phrase identifies the aspects and their attributes, if present, in the phrase. Each attribute is then determined to be either positive or negative, taking the context in due consideration. Even if the phrase doesn't contain any aspect, its Semantic Orientation is calculated by the next phase. A detailed discussion regarding this phase's implementation is done in Step 3 of our Algorithm in Chapter 3. A code snippet showing a part of this phase's implementation is given in Appendix A.6.

4.8 Phrase Semantic Orientation

The semantic orientation of a phrase is calculated using the formula given in Equation 3 of the Algorithm given in Chapter 3.The equation is reproduced here for the sake of completeness.

$$SO(phrase) = \log_2\left(\frac{hits(\text{phrase } NEAR \text{ "excellent"})hits(\text{"poor"})}{hits(\text{phrase } NEAR \text{ "poor"})hits(\text{"excellent"})}\right). \qquad (3)$$

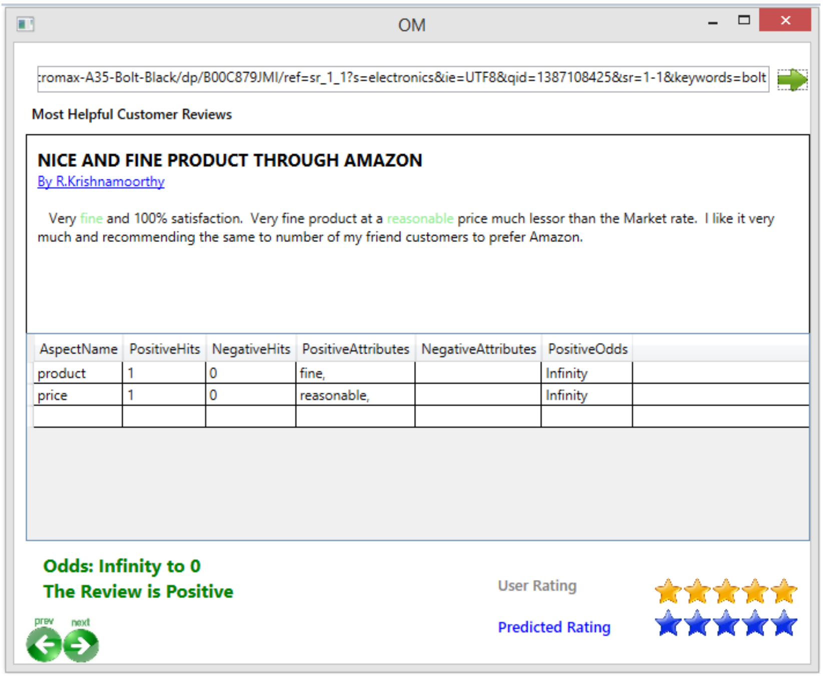
This formula is attributed to [2]. Instead of the NEAR Operator, we used Google's AROUND Operator. Laplacian correction is performed by adding 0.01 to the result of the above formula. The actual code implementing this formula is given in Appendix A.7.

RESULT

Individual Review Rating

Review 1

http://www.amazon.in/Micromax-A35-Bolt-Black/dp/B00C879JMI/ref=sr_1_1?s=electronics&ie=UTF8&qid=1387110005&sr=1-1&keywords=bolt+a35


OM

:romax-A35-Bolt-Black/dp/B00C879JMI/ref=sr_1_1?s=electronics&ie=UTF8&qid=1387108425&sr=1-1&keywords=bolt

Most Helpful Customer Reviews

NICE AND FINE PRODUCT THROUGH AMAZON

By R.Krishnamoorthy

Very fine and 100% satisfaction. Very fine product at a reasonable price much lessor than the Market rate. I like it very much and recommending the same to number of my friend customers to prefer Amazon.

AspectName	PositiveHits	NegativeHits	PositiveAttributes	NegativeAttributes	PositiveOdds
product	1	0	fine,		Infinity
price	1	0	reasonable,		Infinity

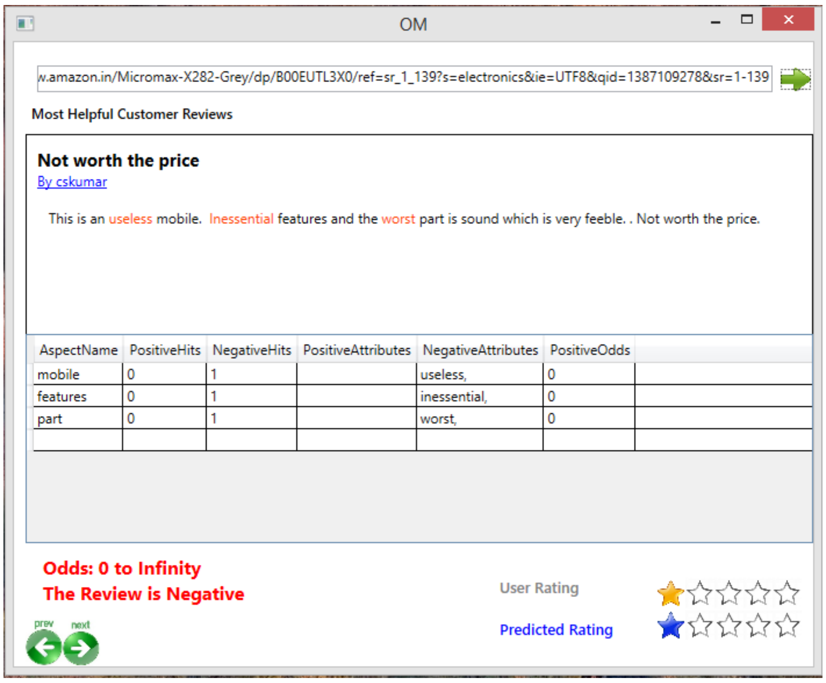
Odds: Infinity to 0
The Review is Positive

User Rating ★★★★★

Predicted Rating ★★★★★

Review 2.

http://www.amazon.in/Micromax-X282-Grey/dp/B00EUTL3X0/ref=sr_1_139?s=electronics&ie=UTF8&qid=1387109278&sr=1-139


OM

w.amazon.in/Micromax-X282-Grey/dp/B00EUTL3X0/ref=sr_1_139?s=electronics&ie=UTF8&qid=1387109278&sr=1-139

Most Helpful Customer Reviews

Not worth the price

By cskumar

This is an useless mobile. Inessential features and the worst part is sound which is very feeble. . Not worth the price.

AspectName	PositiveHits	NegativeHits	PositiveAttributes	NegativeAttributes	PositiveOdds
mobile	0	1		useless,	0
features	0	1		inessential,	0
part	0	1		worst,	0

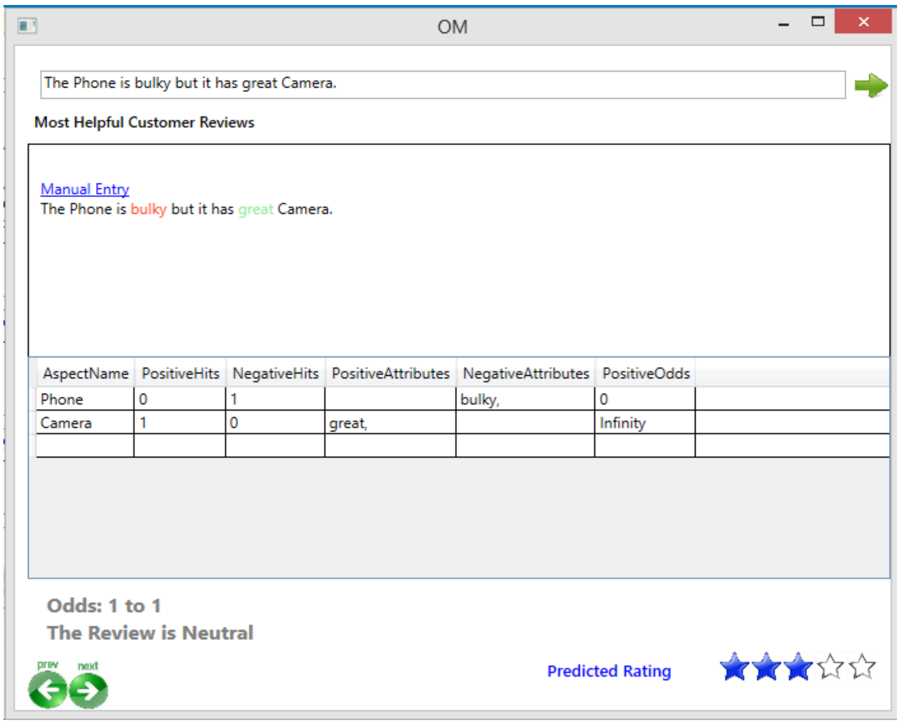
Odds: 0 to Infinity
The Review is Negative

User Rating ★☆☆☆☆

Predicted Rating ★☆☆☆☆

Overall Product Rating

Review 1.



OM

The Phone is bulky but it has great Camera.

Most Helpful Customer Reviews

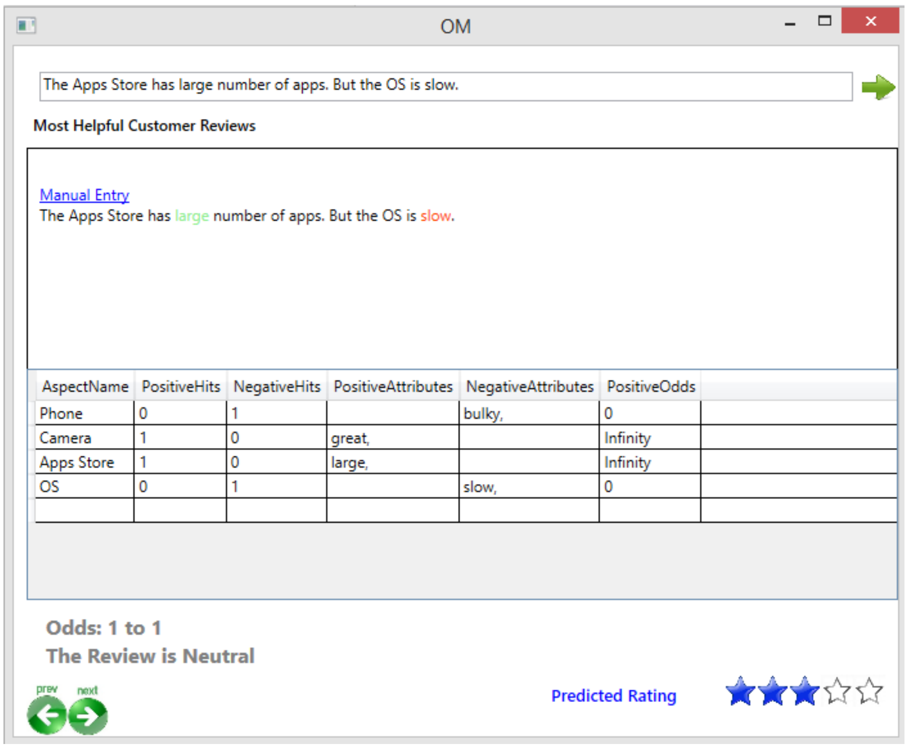
Manual Entry
The Phone is bulky but it has great Camera.

AspectName	PositiveHits	NegativeHits	PositiveAttributes	NegativeAttributes	PositiveOdds
Phone	0	1		bulky,	0
Camera	1	0	great,		Infinity

Odds: 1 to 1
The Review is Neutral

Predicted Rating ★★★☆☆

Review 2.



OM

The Apps Store has large number of apps. But the OS is slow.

Most Helpful Customer Reviews

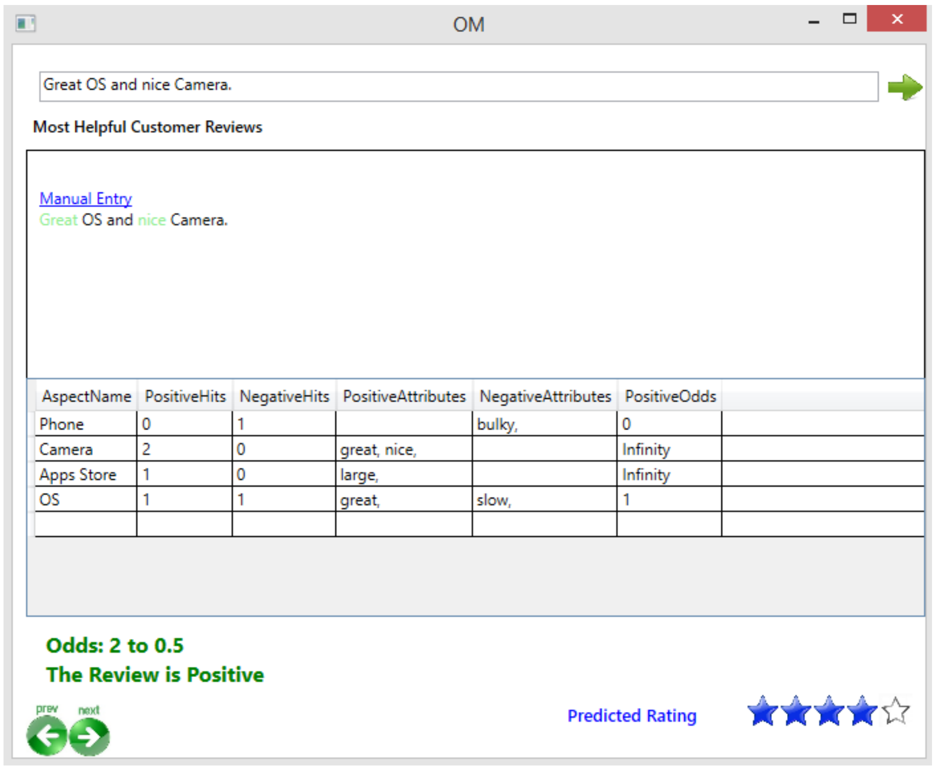
Manual Entry
The Apps Store has large number of apps. But the OS is slow.

AspectName	PositiveHits	NegativeHits	PositiveAttributes	NegativeAttributes	PositiveOdds
Phone	0	1		bulky,	0
Camera	1	0	great,		Infinity
Apps Store	1	0	large,		Infinity
OS	0	1		slow,	0

Odds: 1 to 1
The Review is Neutral

Predicted Rating ★★★☆☆

Review 3.


OM

Great OS and nice Camera.

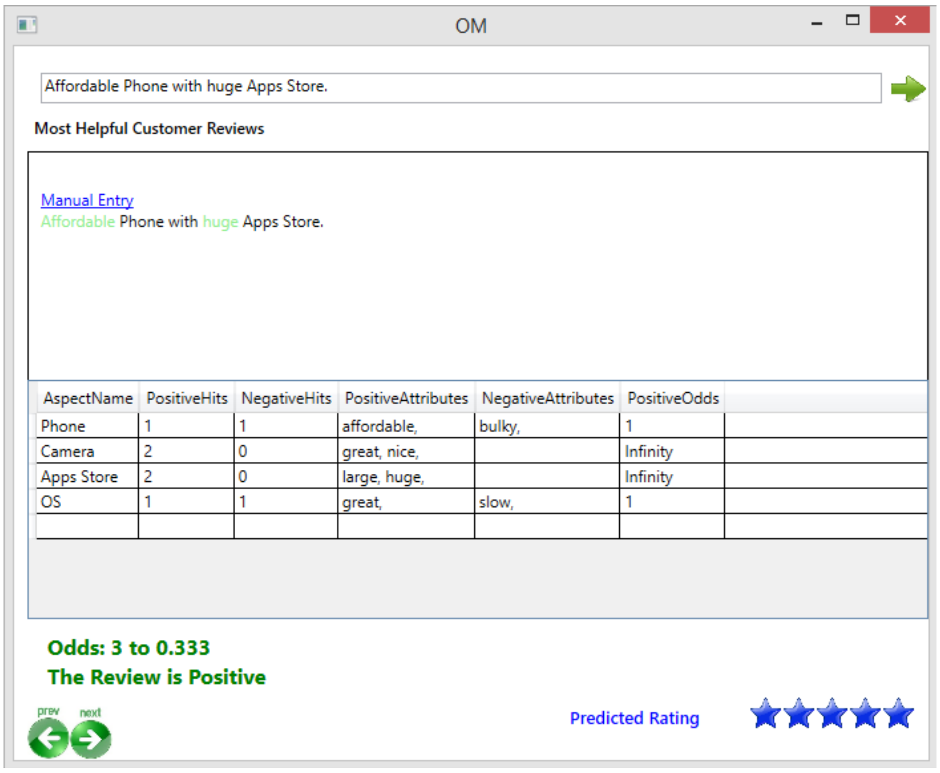
Most Helpful Customer Reviews

Manual Entry
Great OS and nice Camera.

AspectName	PositiveHits	NegativeHits	PositiveAttributes	NegativeAttributes	PositiveOdds
Phone	0	1		bulky,	0
Camera	2	0	great, nice,		Infinity
Apps Store	1	0	large,		Infinity
OS	1	1	great,	slow,	1

Odds: 2 to 0.5
The Review is Positive

Review 4.


OM

Affordable Phone with huge Apps Store.

Most Helpful Customer Reviews

Manual Entry
Affordable Phone with huge Apps Store.

AspectName	PositiveHits	NegativeHits	PositiveAttributes	NegativeAttributes	PositiveOdds
Phone	1	1	affordable,	bulky,	1
Camera	2	0	great, nice,		Infinity
Apps Store	2	0	large, huge,		Infinity
OS	1	1	great,	slow,	1

Odds: 3 to 0.333
The Review is Positive

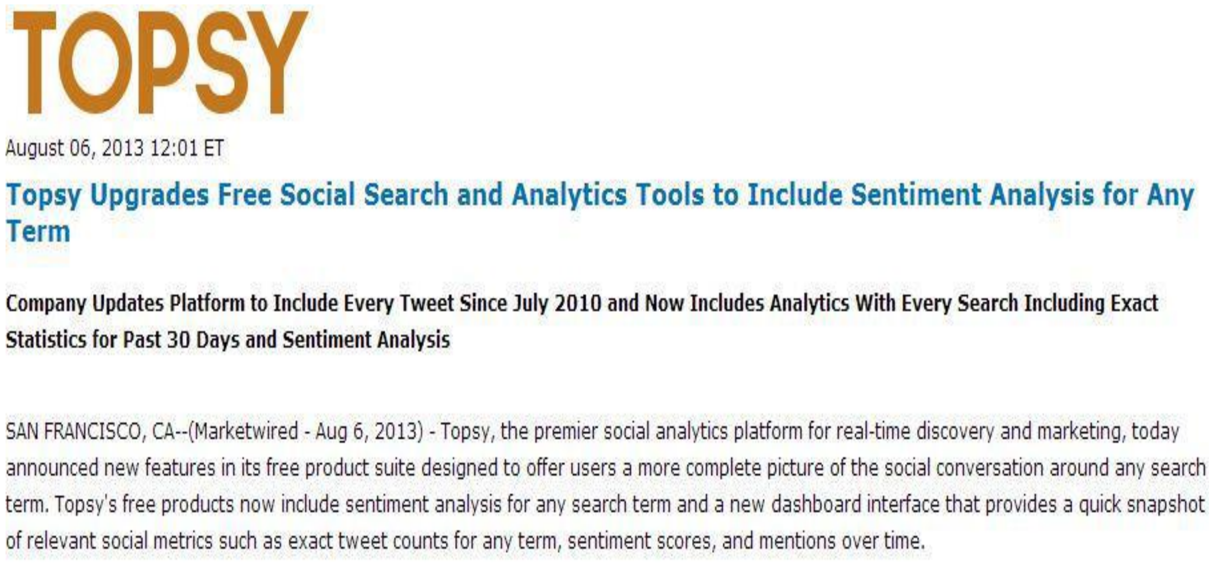
AREAS OF APPLICATION

Opinion Mining/Sentiment Analysis has become an indispensable tool to provide users, be it consumers searching for product reviews to corporate media houses trying to gauge public opinion about their products and services to entire governments monitoring potential terrorist threats, with structured and accurate information from the vast amount of unstructured data available on the Internet.

More formally, Sentiment Analysis is being actively used in the following areas:

- ✓ Online Reviews: Mining public opinion from product reviews into a structured and visual form that is more comprehensible to potential buyers and allows them to make fast and accurate decisions about their future purchases.

- ✓ Current Trends: Automatic financial and market sentiment analysis .For ex: Opfine.com is a real time and fully automatic statistical data mining engine of financial sentiment analysis, market sentiment analysis and companies sentiment analysis news from internet.



TOPSY

August 06, 2013 12:01 ET

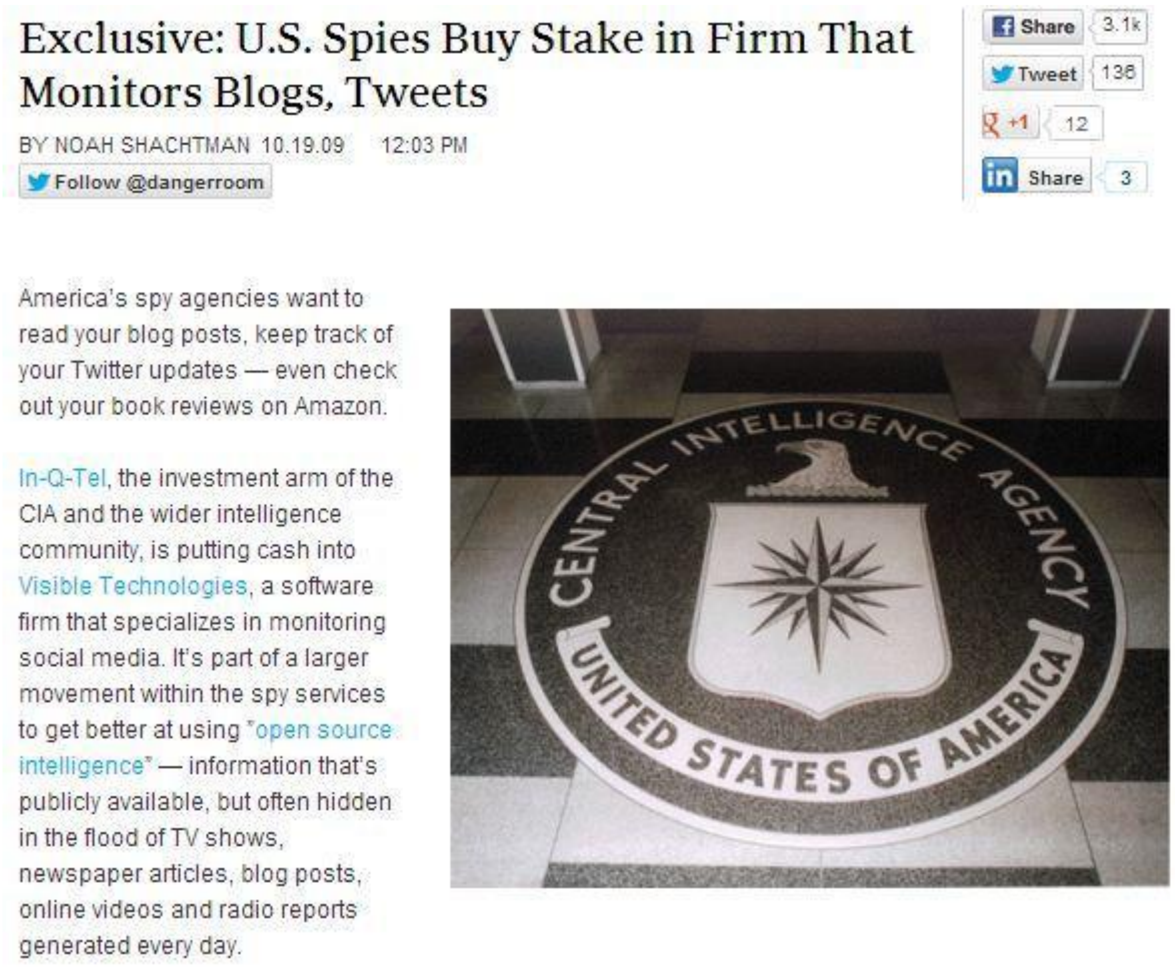
Topsy Upgrades Free Social Search and Analytics Tools to Include Sentiment Analysis for Any Term

Company Updates Platform to Include Every Tweet Since July 2010 and Now Includes Analytics With Every Search Including Exact Statistics for Past 30 Days and Sentiment Analysis

SAN FRANCISCO, CA--(Marketwired - Aug 6, 2013) - Topsy, the premier social analytics platform for real-time discovery and marketing, today announced new features in its free product suite designed to offer users a more complete picture of the social conversation around any search term. Topsy's free products now include sentiment analysis for any search term and a new dashboard interface that provides a quick snapshot of relevant social metrics such as exact tweet counts for any term, sentiment scores, and mentions over time.

- ✓ Surveillance Programs by National Governments: Data and Web mining are used to make sense of the vast amount of public data collected by these mass surveillance programs.

  For ex: PRISM is a clandestine mass electronic surveillance data mining program operated by the United States National Security Agency (NSA) since 2007.PRISM is a government code name for a data-collection effort known officially by the SIGAD US-984XN.



Exclusive: U.S. Spies Buy Stake in Firm That Monitors Blogs, Tweets

BY NOAH SHACHTMAN 10.19.09 12:03 PM

America's spy agencies want to read your blog posts, keep track of your Twitter updates — even check out your book reviews on Amazon.

In-Q-Tel, the investment arm of the CIA and the wider intelligence community, is putting cash into Visible Technologies, a software firm that specializes in monitoring social media. It's part of a larger movement within the spy services to get better at using "open source intelligence" — information that's publicly available, but often hidden in the flood of TV shows, newspaper articles, blog posts, online videos and radio reports generated every day.

FUTURE RESEARCH

In the future, we would like to incorporate the following features into our current project.

• Identify implicit opinions, e.g.
1) “I will never go back to any other camera”
2) “Every person on Earth should own one of these”

• Determine whether the subject and the object are relevant to the “true object”, e.g. in a phone review.
1) “My mom was mad at me because I didn’t tell her I bought a new phone”

• Identify irony/sarcasm, e.g.
1) “Do you know there is a search engine called Baidu” (in a poor search context)
2) “This product is apparently designed by high school students”

• It is not easily applicable to non-reviews, e.g., forum and blog postings, because many such postings evaluate multiple entities and compare them. Also, some of them may not be intended to be evaluations of products but may still contain a few opinion sentences. In such cases, these opinion sentences need to be identified and analysed. Hence further research is needed in this area.

• Social aspect: mine opinions from similar people only

REFERENCES

[1]. Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data by Bing Liu

[2]. Turney, P. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL-2002), 2002.

[3]. Sentiment Analysis in Practice by Yongzheng Zhang, Dan Shen and Catherine Baudin

[4]. Aspect-based Opinion Mining from Online Reviews by Samaneh Moghaddam & Martin Ester from Simon Fraser University

[5]. Brown Corpus Manual by W. N. Francis and H. Kucera, Brown University

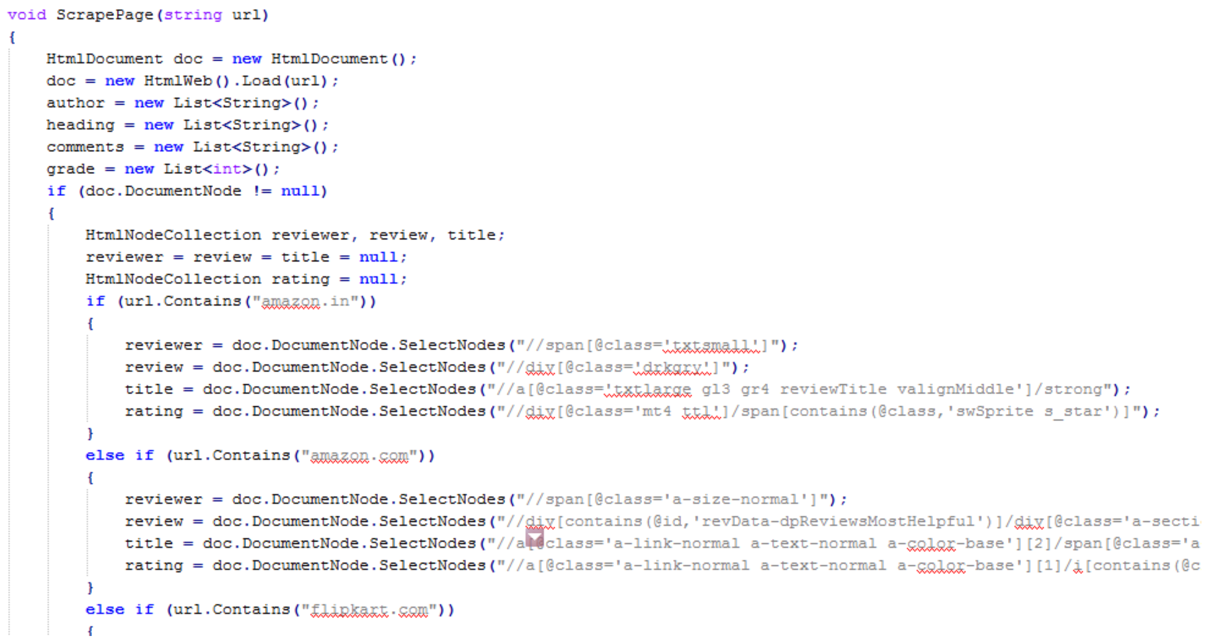
[6]. Hu, M. and B. Liu. Mining and summarizing customer reviews. In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004), 2004.

[7]. The Stanford Natural Language Processing Group

[8]. Brody, S. and S. Elhadad. An Unsupervised Aspect-Sentiment Model for Online Reviews. In Proceedings of the 2010 Annual Conference of the North American Chapter of the ACL, 2010.

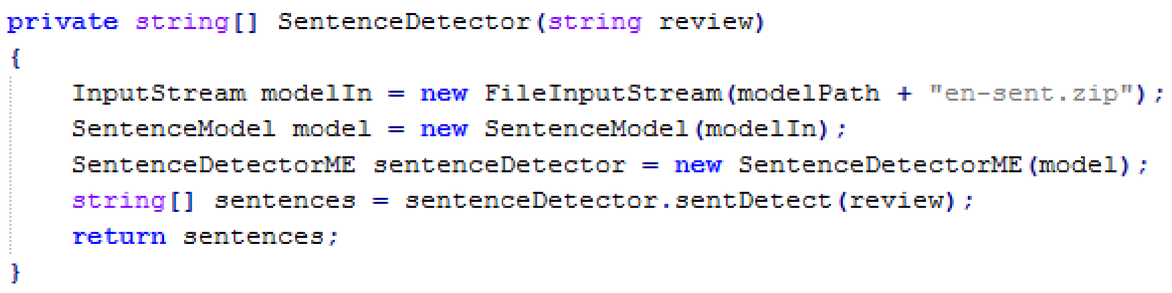
APPENDIX

A.1 ScrapePage



```
void ScrapePage(string url)
{
    HtmlDocument doc = new HtmlDocument();
    doc = new HtmlWeb().Load(url);
    author = new List<String>();
    heading = new List<String>();
    comments = new List<String>();
    grade = new List<int>();
    if (doc.DocumentNode != null)
    {
        HtmlNodeCollection reviewer, review, title;
        reviewer = review = title = null;
        HtmlNodeCollection rating = null;
        if (url.Contains("amazon.in"))
        {
            reviewer = doc.DocumentNode.SelectNodes("//span[@class='txtsmall']");
            review = doc.DocumentNode.SelectNodes("//div[@class='drkgry']");
            title = doc.DocumentNode.SelectNodes("//a[@class='txtlarge gl3 gr4 reviewTitle valignMiddle']/strong");
            rating = doc.DocumentNode.SelectNodes("//div[@class='mt4 ttl']/span[contains(@class,'swSprite s_star')]");
        }
        else if (url.Contains("amazon.com"))
        {
            reviewer = doc.DocumentNode.SelectNodes("//span[@class='a-size-normal']");
            review = doc.DocumentNode.SelectNodes("//div[contains(@id,'revData-dpReviewsMostHelpful')]/div[@class='a-secti
            title = doc.DocumentNode.SelectNodes("//a[@class='a-link-normal a-text-normal a-color-base'][2]/span[@class='a
            rating = doc.DocumentNode.SelectNodes("//a[@class='a-link-normal a-text-normal a-color-base'][1]/i[contains(@c
        }
        else if (url.Contains("flipkart.com"))
        {
```

A.2 SentenceDetector



```
private string[] SentenceDetector(string review)
{
    InputStream modelIn = new FileInputStream(modelPath + "en-sent.zip");
    SentenceModel model = new SentenceModel(modelIn);
    SentenceDetectorME sentenceDetector = new SentenceDetectorME(model);
    string[] sentences = sentenceDetector.sentDetect(review);
    return sentences;
}
```

A.3 Tokenizer

```
private string[] Tokenizer(string review)
{
    InputStream modelIn = new FileInputStream(modelPath + "en-token.zip");
    TokenizerModel model = new TokenizerModel(modelIn);
    TokenizerME tokenizer = new TokenizerME(model);
    string[] tokens = tokenizer.tokenize(review.Replace("-", " ").Trim());
    return tokens;
}
```

A.4 POSTagger

```
private string[] POSTagger(string[] tokens)
{
    InputStream modelIn = new FileInputStream(modelPath + "en-pos-maxent.zip");
    POSModel model = new POSModel(modelIn);
    POSTaggerME tagger = new POSTaggerME(model);
    string[] tags = tagger.tag(tokens);
```

A.5 ExtractPhrases

```
private void ExtractPhrases(string[] tags, string[] tokens)
{
    for (int i = 0; i < tags.Length; i++)
    {
        bool adv_found = false;
        bool adj_found = false;
        bool vrb_found = false;
        bool noun_found = false;
        string adv = "";
        string verb = "";
        string adj = "";
        string opn_word = "";
        string noun = "";
        string possible_not = "";
        while (i < tags.Length && (tags[i].Equals("RB") || tags[i].Equals("RBR") || tags[i].Equals("RBS")))
        {
            adv_found = true;
            adv += tokens[i] + " ";
            i++;
        }
```

A.6 AspectExtraction

```
private void AddAspectToDict(string opn_word, string opn_phrase, string aspect_name, string possible_not)
{
    if (opn_word.Length == 0 || aspect_name.Length == 0) return;
    bool invert_op = false;
    if (possible_not.Length > 0 && (possible_not.ToLower().Contains("not") || possible_not.ToLower().Contains("n't")))
    {
        invert_op = true;
    }
    opn_word = opn_word.ToLower().Trim();
    opn_phrase = opn_phrase.ToLower().Trim();
    aspect_name = aspect_name.Trim();
    if (aspects.ContainsKey(aspect_name))
    {
        AspectClass aspect_class = aspects[aspect_name];

        if (!(aspect_class.pos_opn_words.Contains(opn_word) || aspect_class.neg_opn_words.Contains(opn_word)))
        {
            if (useless_words.Contains(opn_word))
            {
                return;
            }
            else if (cxt_sen_words.Contains(opn_word))
            {
                if (SemanticOrientation(opn_phrase) > 0.0)
                {
                    if (invert_op)
                    {
                        aspect_class.neg_review++;
                        aspect_class.neg_opn_words.Add(opn_word);
                    }
```

A.7 SemanticOrientataion

```
private double SemanticOrientation(string phrase)
{
    double posHits = NumOfHits(Uri.EscapeUriString("\"" + phrase + "\"" + "AROUND(10)" + "\"excellent\"")) + 0.01;
    double negHits = NumOfHits(Uri.EscapeUriString("\"" + phrase + "\"" + "AROUND(10)" + "\"poor\"")) + 0.01;
    if ((posHits - negHits) == 0.0)
    {
        System.Console.WriteLine(phrase + " :0.01");
        return 0.01;
    }
    double so = (posHits / negHits) * (451000000.0 / 473000000.0);
    so = (Math.Log(so) / Math.Log(2));
    System.Console.WriteLine(phrase + ":  " + posHits + " , " + negHits + " : " + so);
    if (so > 2.0) return 1.0;
    else if (so < -2.0) return -1.0;
    else return so;
}
```