Coventry University

School of Computing

Engineering and Mathematics

Investigation of Sentiment Analysis Models for Natural Language Processing: learning from consumer reviews to classify polarity

Author:
Torin Pitchers
SID: 4955012
Degree Title:
Computer Science

1st Supervisor:
Phillip Smith
2nd Supervisor:
Simon Billings

June 15, 2016

Contents

1 Statement of Originality

Department of Computing
COVENTRY UNIVERSITY

300COM / 303COM Declaration of originality

I Declare that This project is all my own work and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet etc.) has been acknowledged by citation within the main report to an item in the References or Bibliography lists. I also agree that an electronic copy of this project may be stored and used for the purposes of plagiarism prevention and detection.

Statement of copyright

I acknowledge that the copyright of this project report, and any product developed as part of the project, belong to Coventry University. Support, including funding, is available to commercialise products and services developed by staff and students. Any revenue that is generated is split with the inventor/s of the product or service. For further information please see www.coventry.ac.uk/ipr or contact ipr@coventry.ac.uk.

Statement of ethical engagement

I declare that a proposal for this project has been submitted to the Coventry University ethics monitoring website (https://ethics.coventry.ac.uk/) and that the application number is listed below (Note: Projects without an ethical application number will be rejected for marking)

Signed: *Torin Pitchers* Date: **15/04/2016**

Please complete all fields.

First Name:	**Torin**
Last Name:	**Pitchers**
Student ID number	**4955012**
Ethics Application Number	**P41281**
1st Supervisor Name	**Phillip Smith**
2nd Supervisor Name	**Simon Billings**

This form must be completed, scanned and included with your project submission to Turnitin. Failure to append these declarations may result in your project being rejected for marking.

2 Certificate of Ethical Approval

Certificate of Ethical Approval

Applicant:

Torin Pitchers

Project Title:

computer application to analyse yelp reviews for data mining

This is to certify that the above named applicant has completed the Coventry University Ethical Approval process and their project has been confirmed and approved as Low Risk

Date of approval:

12 February 2016

Project Reference Number:

P41281

3 Definitions

For clarification purposes I will list the technical terms used throughout this report and define the meaning as I understand it to provide the context of the report.

Word	**Meaning**
Bigram	*a bigram is of size 2 and consists of two sequential tokens. For example [not like].*
Big Data	*Data that is large in volume, complex in variety and consistent in velocity.* (Berman 2013)
Classifier	*An algorithm that implements classification of input data to a single category*
Corpus	*A collection of written texts, especially the entire works of a particular author or a body of writing on a particular subject.* (Oxford Dictionary 2016)
Features	*A set of attributes of the text that holds value in terms of a given context. For example an individual word can be a feature or the amount of positive words in the whole document could be a feature also. This term is quite broad.*
Feature Extraction	*Extracting attributes of text that holds value in terms of a given context. For example extracting features of text to determine polarity.*
Feature Vector	*Essentially a list of features ready to be passed to the classifier as input.*

Table 1: A table of definitions.

Word	Meaning
Natural Language Processing	*The process of understanding natural language that is written and spoken by humans.*
Negation	*A token is subject to negation when the token that precedes it is equal to 'not'. where the token consequently implies the opposite of its value*
Opinion Words	*Words that hold some form of opinion value, such as words that express a positive or negative attitude.*
Polarity	*The positive or negative value expressed in a document, sentence or entity.*
Sentiment Analysis	*The extraction of the opinion of a document, sentence or entity which is classified in regards to polarity.*
SentiWordNet	*A corpus of English words each paired with corresponding sentimental information on that word.*
Stop words	*The most common words in the English language.*
Tokenizer	*An algorithm that splits a string of text into sub strings defined by the tokenizer. (NLTK Project 2016)*
Tokens	*The technical name for a sequential group of characters such as (hairy, his, not).*(Bird, Klein and Loper 2009)
Trigram	*a trigram is of size 3 and consists of three sequential tokens. For example [not like food].*
Unigram	*a unigram is of size 1 and consists of one token. For example [like].*
Valuable Word	*A word that is not a stop word and also present in SentiWordNet.*

Table 2: A table of definitions.

4 Abstract

In this project the aim is to investigate a sentiment analysis model that learns from consumer reviews from the yelp data set and then classifies these reviews as either positive or negative. This lies in the general area of artificial intelligence and uses natural language processing to understand the text while using machine intelligence allows the model to make the classification.

Specifically I investigate feature extraction models and the effect of including spell checking to the model. The accuracy, accuracy difference, most informative features and runtime was recorded for each feature extraction method that was used in the model, these were: Bag of words, Bag of valuable words, Attributed bag of valuable words, N-gram bigrams and N-gram trigrams. These feature extraction models were used to extract features of the reviews to create a feature vector that could be used by the naive bayes classification model.

It was found that the most successful model was the bag of valuable words model for feature extraction in combination with the naive bayes model for classification, learning from yelp reviews to classify as positive or negative also with the inclusion of a spell checker. This generated an accuracy of 90.2284915906%. The addition of only including features in the feature vector that held an opinion by SentiWordNet generated an increase in accuracy by 28.5897578882% when compared to the original bag of words model. Bigrams and trigrams were found to be successful for identifying common phrases that are repeated in the texts. The addition of a limited spell checker to the bag of valuable words model also provided a trivial improvement of +0.0014403756% for the bag of valuable words model. lastly, across all models negative features were found to be the most informative.

5 Introduction

As the consumer marketplace continues to evolve and becomes more and more integrated with the internet, the number of online consumer written reviews continues to increase. We are at a point today were consumer written reviews for businesses and products have become so abundant that they have entered the domain of big data. For example Amazon alone provide a data set that contains 34.6 million reviews from June 1995 to March 2013. A review fundamentally provides an assessment of something. It does this by describing experiences as positive or negative. Most reviews tend to also be paired with a numerical rating. From these two key points we can understand that this vast amount of data provides insight into how humans classify text as positive or negative. Therefore if we can create a system that can be taught to learn from this data we can teach a machine how to determine the polarity of text. This brings us one step closer to the goal creating a machine that has human like intelligence. This could also be used by businesses to apply the system to their specific data allowing them to further understand how the consumer feels about their content. For example the most positive and negative aspects of their content.

Other research has been carried out tackling similar problems. When reviewing the literature to implement my own model I had difficulty researching which feature extraction model was most accurate and/or efficient. This was mainly due to the variance in methods across the models, which had mixed accuracy results. So investigating the accuracy and efficiency for different sentiment analysis models is one aim of the project. The writer of a similar project to mine discussed investigating spell checking however never discussed it in the results (Rains 2016). I agree with this opinion and decided to investigate this. This is the second aim to investigate if spell checking will provide some improvement to the model accuracy.

The purpose of this project is therefore to investigate the modelling of a sentiment analysis system that can accurately predict the polarity of text written reviews as positive or negative, learning from existing consumer reviews paired with a star rating. This is performed by investigating feature models and then determining how the particular feature extraction model affected the accuracy of the classifier. Using this information to perform an analysis across the different models. My best model with the highest possible accuracy is then implement with spell checking and the difference in accuracy will be investigated.

5.1 Natural Language Processing

The first task of this problem requires the system to understand and process the review text, this lies in the area of natural language processing(NLP). NLP can be described as the term normally used to describe the function of software or hardware components in a system that analyses or synthesis natural language.

Natural language refers to a human language and is a language spoken and understood by people. A machine only understands machine language which causes a barrier, however Natural language processing allows human-computer interaction in the form of natural language.(Chopra, Prashar and Sain 2013)

To elaborate on this problem I will explain how a human and machine both view the same piece of text written in natural language. The example text I will use is *"I love Food"*.

What a human sees.

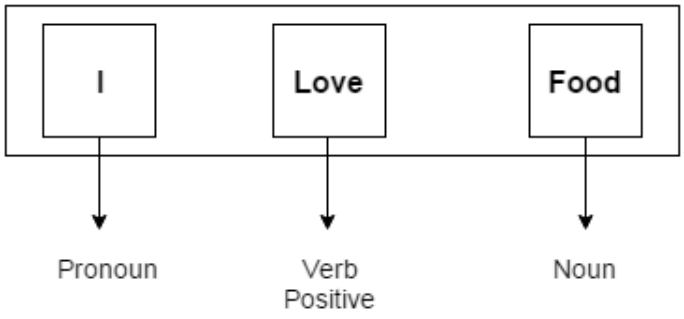


Figure 1: Human Interpretation of text

What a machine sees.

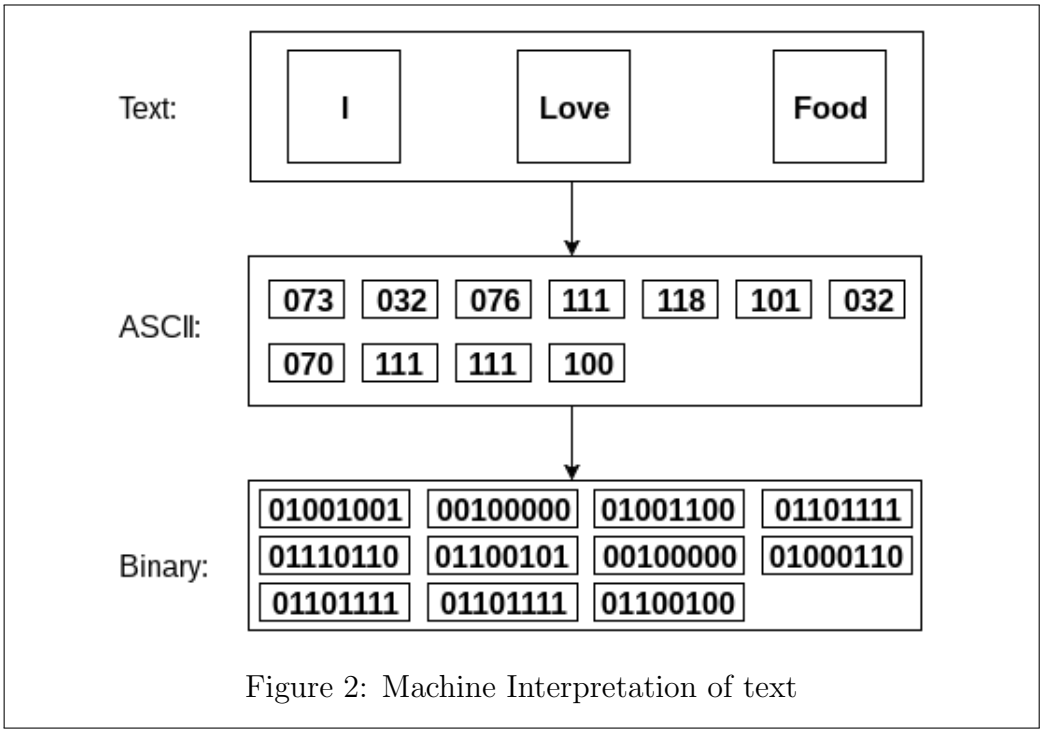


Figure 2: Machine Interpretation of text

From Figure 1 we can see that humans see words as structured objects with properties such as positioning, and meaning. We can make connections between the words based on this information and this allows us to understand the overall meaning of text. Where as from figure 2 we can see that a machine has a much different view. In reality all a computer understands is the numerical value that represents the individual characters of the text. The machine also knows the sequential structure however does not understand the connection between the structure. In summary we want to teach the machine more information about the text to further understand it like a human does.

A similar literature to mine described NLP as both an interesting and difficult task with lots of baggage, and as such is a problem for another project (Chopra, Prashar and Sain 2013). From this, rather than developing my own natural language processing framework for my system implementation, I make use of the NLTK Python module. Bird, Klein and Loper demonstrate the NKLT module as a natural language processing tool kit that defines an infrastructure to build NLP applications (Bird, Klein & Loper 2009).

5.2 Sentiment Analysis

The second and most important task is classifying a text review as positive or negative. This is known as sentiment analysis or otherwise known as opinion mining.

Sentiment analysis or opinion mining is the computational study of opinions, sentiments and emotions expressed in text. Sentiment analysis is therefore an attempt to synthesize text in terms of opinion, this is a specific area of Natural language Processing contributing to the wider problem of understanding natural language text (Bing, L 2010). Narrowing down even further the task of modelling a sentiment analysis system that determines polarity is a specific area of the sentiment analysis problem as polarity is only one aspect of an opinion.

Sharma's study and Bing 's study agree with one another and describe three levels on which sentiment analysis can be performed (Sharma, Nigam and Jain 2014) & (Bing, L 2010). These three levels were:

1. Document level (determine polarity of the whole document) (Sharma, Nigam and Jain 2014).

2. Sentence level (determine polarity of sentences) (Sharma, Nigam and Jain 2014).

3. Aspect & Feature level (determine polarity of documents/sentences based on the aspects of those documents/sentences) (Sharma, Nigam and Jain 2014).

In this project I will perform aspect and feature level sentiment analysis on the reviews at the document level. This means extracting features from the reviews that can be used to determine the polarity of the review at the document level. My reasoning for determining the polarity at the document level is because I am not concerned with how positive or negative the review is, I only need to know weather its positive or negative as whole. If i was concerned with the level of polarity then I would choose to determine the polarity at the sentence level, however I am not.

5.3 Software Development

To test the system and perform my investigation I will be required to implement the system. I have chosen to develop the software in Python because the NLTK module is written in python. Also due to python being an interpreted language new code can be executed at runtime which is useful for accessing the data for analysis.

6 Literature Review

6.1 Natural Language Processing

For my analysis the software for the model implementation needs to be able to understand the review text which is written in natural language which means I need to look at natural language processing.

A natural language refers to a human language and is a language spoken and understood by people. Natural language processing allows human-computer interaction by the form of natural language. A computer understands machine language and a human understand natural language therefore natural language processing enables the computer to understand and generate natural language.(Chopra, Prashar and Sain 2013)

The steps of natural language processing are:

1. Morphological and Lexical Analysis (analysing the structure of words and dividing the text into paragraphs, words and sentences) (Chopra, Prashar and Sain 2013).

2. Syntactic Analysis (analysing sentences to interpret the grammatical structure) (Chopra, Prashar and Sain 2013).

3. Semantic Analysis (Theses structures are given meaning from dictionary or exact meaning from context) (Chopra, Prashar and Sain 2013).

4. Discourse Integration (the meaning of a sentence is dependent upon sentences that precede it. Such as, "Tom had an apple. Adam wants it.". To understand what "it" is we need to read the sentence that precedes it.) (Chopra, Prashar and Sain 2013).

5. Pragmatic Analysis (what is intended by the natural language text and how should it be interpreted? does the speaker mean what they literally say or do they imply another meaning? such as sarcasm) (Chopra, Prashar and Sain 2013).

Chopra, Prashar and Sain describe NLP as both an interesting and difficult task with lots of baggage. (Chopra, Prashar and Sain 2013) We can understand that it is not practical to develop my own NLP software as this is not the aim of my project furthermore it is a tool enabling me to complete

my research plan. However Steven Bird, Ewan Klein, and Edward Loper demonstrate the open-source python NLTK module which would allow me to perform natural language processing on the review text, manipulating and analysing the language data (Bird,Klein & Loper 2009). The NLTK module Defines an infrastructure to build Natural language applications and provides methods for operations that are common to natural language processing applications.

They also recommend the following python libraries for building NLP applications in python:

- NumPy (a scientific computing library, required for certain probability, tagging, clustering, and classification tasks.).
- Matplotlib (a 2D plotting library, providing data visualisation).

The NLTK module and the recommended libraries will allow me to perform NLP as listed by Chopra, Prashar and Sain 2013. Open-source software is great for large problems such as this and is provided free. Allowing this large problem to be continually developed by many people and the solutions are provided for free and can be implemented to perform NLP.

6.2 Opinion Mining, Sentiment Analysis and Machine Learning

Opinion mining is an NLP and information extraction task that retrieves the feelings of the user expressed in comments by analysis. Sentiment analysis concerns classifying the documents to determine polarity, which is expressed in neutral, positive or negative. (Sharma, Nigam and Jain 2014) This is useful for my project to analyse the review text for rating and determine the positivity.

There are three levels that sentiment analysis can be performed:

1. Document level (determine polarity of the whole document) (Sharma, Nigam and Jain 2014).
2. Sentence level (determine polarity of sentences) (Sharma, Nigam and Jain 2014).
3. Aspect & Feature level (determine polarity of documents/sentences based on the aspects of those documents/sentences) (Sharma, Nigam and Jain 2014).

After this a machine learning classification model can be used to classify the document as positive, negative, or more specifically classify it as 1,2,3,4 or 5. A probabilistic classifier is an instance of machine learning where an algorithm is used on a given set of sample data (the feature vector) that then predicts a probability distribution over the set of classes (positive and negative).This probability distribution is then used to predict the class that the text belongs to (Hastie, Tibshirani & Friedman 2009).

Opinion mining is the core topic of this project, by which i mean the way that i extract the opinion of the review text is what the success of this project relies on. I found journals that concerned very similar projects to mine and I found that this can be done in different ways, and is a current area that is still being researched this literature will be discussed in the next section of this literature review.

6.3 Defining a basic model

If I wish to investigate sentiment analysis models then I need to understand the fundamentals of sentiment analysis models. To do this I cross referenced all the literature from studies that were similar to mine and that concerned sentiment analysis and polarity classification, the literatures were:

- *'opinion mining of movie reviews at document level'* (Sharma, Nigam & Jain 2014).
- *'Sentiment Analysis in Amazon Reviews Using Probabilistic Machine Learning'* (Rain 2013).
- *'Prediction of rating based on review text of Yelp reviews'* (Channapragada & Shivaswamy 2015).
- *'Predicting a Business' Star in Yelp from Its Reviews'* (Fan & Khademi 2014).
- *'Prediction of Yelp Review Star Rating using Sentiment Analysis'* (Li & Zhang 2014).

After cross referencing these literatures, three key sections seemed to be consistent. These were Data Collection, Feature Extraction and Classification.
Data Collection - This section included for example: Downloading the data, extracting the data. formatting the data, labeling the data, dividing the data.
Feature Extraction – This section involves extracting features of the text and formatting them so that the classifier can read them as input.
Classification – This section involves using a statistical probabilistic algorithm to learn from the data by calculating probability distributions to then use this to classify new input data.

The methods for each section did however have a large variance. For example one study implemented the bag of words model for feature extraction (Rains 2013). However another study used a slightly different method called seed list (Li & Zhang 2014). This variance occurred for every section.

Below is a series of tables documenting all the different methods, models or algorithms for feature extraction models and classification models that appeared in the literature:

Feature Extraction
Bag of words (Rains 2013)
Collocations/n-gram model (Rains 2013)
Seed list (Sharma, Nigam & Jain 2014)
Seed list from stopword removal (Li & Zhang 2014)
frequency distribution of the 1000 most common words (Channapragada & Shivaswamy 2015)
Positive vs negative (Li & Zhang 2014)

Table 3: A table showing the different feature extraction models across the literature.

Next is a table for classification models:

Classification
Naive Bayes (Rains 2013)
Decision List (Rains 2013)
Linear Regression (Channapragada & Shivaswamy 2015)
Support Vector Regression (Li & Zhang 2014)
Decision Tree Regression (Fan & Khademi 2014)
Logistic Regression (Li & Zhang 2014)
Support Vector Machine (Channapragada & Shivaswamy 2015)

Table 4: A table showing the different classification models across the literature.

6.4 Reason for the research & Hypothesis Generation

From reading the literature I found that there are lots of different feature extraction and classification models and most focused on the difference in error between them. For example Callen Rains found bag of words to be the most effective feature extraction model when used with the naive bayes classifier (Rains 2013). However Sasank and Ruchika found that using the top adjectives as a feature extraction model gave the best accuracy when used with the support vector machine classifier (Channapragada & Shivaswamy 2015). Whilst Mingming and Maryam concluded that using a list of the most frequent words and then checking if they existed in the review was the least error prone feature extraction model when used with the linear regression model for classification (Fan & Khademi 2015).

From the above discussion and comparison we can first conclude that the literature does not agree with each other, this means another investigation will be useful in contributing to the current research to decide the best and most appropriate model. I believe that a choice should be made to investigate one or the other rather than mixing different feature extraction models with different classification models. So by choosing one to investigate this ensures one thing is tested at a time so a clear cause and effect connection can be made by manipulating only one variable. I believe feature extraction to be the most important part of the model because if the classifier was given inaccurate features to learn from then it will obviously perform badly. So for this reason I will investigate different feature extraction model and use only one classification model, keeping this the same for each feature extraction model.

I have the hypothesis that spell checking will improve the accuracy of the model. This was first discussed by Callen Rains in her research where they talked about how important features can be missed due to a difference of spelling (Rains 2013). I feel this is a good hypothesis because people are not perfect and spelling mistakes do occur. For a corpus of 2.2 million reviews there is very high chance that there will be many instances of spelling mistakes. So this hypothesis seems relevant and will be investigated.

6.5 Choosing the dataset source

As explained in section 6.5 the data set we are using for the training and test data is going to be consumer reviews from Yelp. From the 5 literature I cross referenced earlier there seem to be two key sources. The first was very popular and is Yelp reviews which was present in 4 out of the 5 studies. However the one who differed used Amazon reviews (Rains 2013).

From looking at some other sources to decide on which one to use, woolf found that typically an amazon review is usually between 100 - 150 characters (Woolf 2014). Were as Hung and Qui found that a yelp review averages a word count of around 20-30 words for each star class(1-5), and the word lengths mainly range between 0 and 150 (Hung & Qiu 2014). Yelp reviews seem to hold more information than an Amazon review concluding from these two points. For this reason the yelp dataset should be used.

The yelp dataset is available as part of the yelp data set challenge and is retrieved by requesting the data set from yelp and can be used for academic purposes only (Yelp 2016). This means that I am able to use this data set in my project. The data set contains 2.2 million reviews from consumers who posted them to the yelp platform for businesses (Yelp 2016). This data set is also suitable for the project in that it has a very large amount of reviews that can be used to train the classifier. Finally the yelp data set does not provide any sensitive user data such as names or addresses, at most it contains simply a user ID (Yelp 2016). The anonymous properties of the data set is another reason to use the data set as I can avoid having to conform to the data protection act which would maybe cost more time.

6.6 Data Sample Size

Some of the literature decided to select a sample of the data to train and test from although their methods of choosing a sample varied. For example Li and Zhang decided to select the user with the most reviews and only use only them reviews for training and testing (Li & Zhang 2014). Fan and Khademi decided to randomly choose 1000 business's and analyse their reviews, consequently this was a sample of 35645 reviews (Fan & Khademi 2015). Another study which used the yelp data set similarly to me only chose a random sample of 162K reviews (Channapragada & Shivaswamy 2015). One studie decided to use all the reviews in the data set, for example Callen Rains used 50000 Amazon reviews obtained by downloading them manually using a script (Rains 2013). However this was not a large corpus size so it seems that sampling would not be required anyway. From this discussion it looks more common to scale back the dataset size. However after watching an interesting Ted talk by Fei-Fei Li. She discusses how she feels machine learning is being held back by the large quantity of data that is required to accurately learn from. Her argument for this stems from how a child learns in form of audio and images that are captured by there senses and by age 3 have been exposed to hundreds of millions and this is how they learn. She applied this to machine learning and started the ImageNet project and found success using this ideology when teaching a machine to see (Li 2015). I see the logic in Li's thought pattern and tend to agree with this. For this reason I have made the decision to use the entire data set sample of 2.2 million reviews.

6.7 What feature extraction models should we test

From table 3 & 4 we can understand that there are many different methods, models or algorithms to choose from and they can't all be investigated because of time restrictions, this means I need to choose which ones to investigate. One piece of literature stated that the bag of words model is the simplest method of feature extraction (Jackson 2010). Starting from the basics seems like a good reason to add this to the investigation list as my first feature extraction model. Bag of words is simply explained as a model where each word is stored as a feature and then translated into a pre labelled feature vector that is given to the classifier to train and then test.

I found in one study that they used extracted opinion words and used these in combination with a seed list to create a feature vector, this was their feature extraction method (Sharma, Nigam & Jain 2014). I feel strongly about this idea of opinion words because it would allow the classifier to focus on only words that contributed value in terms of polarity to the text. However the writer did not fully explain how they defined an opinion word or the method to obtain them. From my understanding of the NLTK module I discovered the package SentiWordNet. From their website SentiWordNet is described as *"A lexical resource for opinion mining. SentiWordNet assigns to each synset of WordNet three sentiment scores: positivity, negativity, objectivity.*" (SentiWordNet 2016) I assume that by using the positive and negative scores of the words in the text that I could then use this information to extract what I will define as opinion words. This should give the classifier more accurate features and hopefully an improved accuracy. This will be my second feature extraction model called bag of valuable words model as it is essentially an extension of bag of words model.

Following the thought pattern of more informative features will likely yield a higher accuracy it seems logical to also give the classifier more information about the bag of words as a whole. These could be attributes of the bag such as the size or maybe the total positive score of the bag by adding all the individual positive scores together. This will be my third feature extraction model I will investigate.

Callen Rains discusses collocations in her study saying that *"Since the bag of words feature model assumes the independence of each of the words, it ignores some of the relationships between words that add affect their meanings in the context of the article. For example, the phrase "low price", has a different meaning than 'low' and 'price' appearing independently."* (Rains

2013). Li and Zhang also mention the topic and make a suggestion for future work to try more feature extraction approaches such as bigram, trigram or word chunks. This is because this could effectively increase the information that can be obtained from one review text, and add more features to overcome the high bias problem. (Li & Zhang 2014)

This topic is more formally known as the n-gram model which includes any number of consecutive words such as a trigram which is three words consecutively. Using these as features allow the classifier to identify common recurring phrases and use this information to more accurately make a classification. In addition I could as previously discussed use SentiWordNet also to extract the opinion words with the n-gram model.

6.8 How feature extraction & classification is performed

So far I have discussed that I will investigate feature extraction models to understand how exactly this is performed I am reading from the Natural Language Processing With Python book (Bird, Klein & Loper 2009). This book is very well suited because all the examples are using the python NLTK module that I am also using. The book explains the process of tokenizing the text into single words called tokens and how these can be manipulates using the other NLTK packages. Good examples are provided in the book that teach you from the initial stage of gathering the data through to the final stage of implementing the classifier.

6.9 Which classification model to use

As discussed in section 7.4 I will use one classifier throughout all models. This creates the problem of what classification model I should use.

Callen Rains explains in her study that naive bayes is the most simple and robust classification method (Rains 2013). Agreeing with Callen Rains another author found that naive bayes classification model had slightly better accuracy than the support vector machine (Channapragada & Shivaswamy 2015). However Sasank & Ruchika disagree as they tested two different classification models (naive bayes classifier and the support vector machine(SVM) classifier) and used a prediction method called linear regression, SVM classification method was found to be the most successful (Sasank & Ruchika 2015). *"We see a common trend of SVM being better than naive bayes Classifier for all the features. This is because naive bayes falsely assumes that there is conditional independence"* (Sasank & Ruchika 2015). Naive bayes seems to stand out as the most appropriate classification model to use as it appears to have had a good response from other studies and has been described as simple to use although being quite robust.

Across all the studies When a classifier was implemented the data is divided into two categories training data and test data. The proportions of each category varied from study to study for example one study divided the data into 70% training data and 30% test data (Li & Zhang 2014). Another study used 90% training data and 10% for testing data (Fan & Khademi 2015). Furthermore another study used 80% training data and 20% for testing data (Channapragada & Shivaswamy 2015). In response to this it seems to be common to have upwards of 70% training data and the rest for testing data, so I will proceed to use 75% training data and 25% testing data.

6.10 Negation

Negation was mentioned in two studies where a negating word for example *"not"* inverts the polarity of the following word subject to the negation. This is an obvious yet important problem that should be handled because we want our model to be as accurate as possible. If this is not handled then words may be interpreted as the opposite meaning to what the writer intended.

Sharma, Nigam and Jain discuss in their study that negation was handled explaining their approach was that they used word net to retrieve the synonyms and antonyms and used these in their negation handler (Sharma, Nigam & Jain 2014) . Callen Rains also agrees with this problem and helpfully found that to best solve this problem you should store three words after negating words to determine when negation should occur (Rains 2013).

I have chosen to take the same approach with Callen Rains and check for a negating word within the previous 3 words and if found this will be stored as a bigram.

6.11 Defining success – What accuracy have others achieved

It makes sense to assess the current literature and see what accuracy other studies managed to achieve. This will allow me to make more realistic expectations and understand what is considered a good accuracy and what is considered a bad accuracy.

The highest accuracy was achieved by callen rain and was 0.84496124 which is about 84% this was however only concerned classifying book reviews from Amazon as positive or negative which makes these results a little limited (Rain 2013). Another study led by Sharma achieved a 63% accuracy classifying into the categories positive, negative and neutral (Sharma, Nigam and Jain 2014). The other studies all used root mean square error(RMSE) or mean square error(MSE). I could record the mean square error and compare, however I feel that the classic predictive accuracy measurement is a better metric because rather than focusing on the errors of the model I am more interested in what the model got correct. Channapragada and Shivaswamy somewhat agree when they explain, in section 3 of their study, that its more appropriate to use RMSE when the classification problem is for decimal ratings for example 1-5 (Channapragada & Shivaswamy 2015). Furthermore Callen Rains also mentions in her discussion that one of her models was deemed successful as it was more accurate than a randomized baseline of 50% accuracy when classifying between two categories (Rains 203). Summarizing this discussion i can deduce that I should aim for above 50% accuracy to deem my model to be useful, In comparison to the other research it would seem that upwards of 84% accuracy could be deemed successful in the sense that it would be more accurate than the most accurate literature I have reviewed.

While on the topic of accuracy it is important to note that there were differences between the labelling of the data and consequentially the classification categories of the data. For example one study was attempting to classify the text as a rating between 1 and 5 thus there were 5 different categories: 1,2,3,4 and 5 (Li & Zhang 2014). This rating prediction model seemed to the most common however another study had the classification categories positive, negative and neutral (Sharma, Nigam & Jain 2014). These differences mean that caution should be taken when directly comparing the results between studies and also with my own project.

7 Methodology

From cross referencing the literature in section 7.3 three general stages seem to be present, however each literature varied in how they approached each stage. A basic sentiment analysis model or structure can be be described by these 3 key stages. These are Data collection, Feature extraction, and classification.

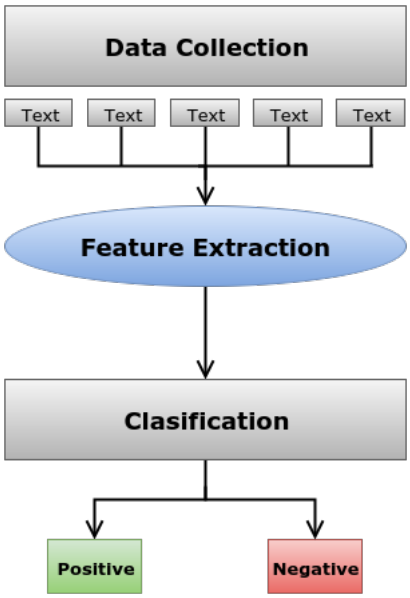


Figure 3: Basic Sentiment Analysis Structure for modelling

Figure 3 provides a high level overview of a sentiment analysis system however is generic and a better model with more detail could be described. For example how exactly should we collect and handle the data?, what features should we extract?, and how should we classify?. Figure 3 does however provide a solid structure to work from. Where a chosen stage can be investigated, expanded and linked together to build a model.

The literature review gave me the opinion that feature extraction is very important because the classifier only has access to the data it receives from the feature vector. If this data is inaccurate and uninformative then clearly

the classifier will have a difficult time classifying the text. However if we give the classifier accurate and informative feature vectors then this will be easier for the classifier as it wont get distracted and waste time on obvious uninformative features. The literature review also showed that naive bayes was the simplest and most robust classifier so i will use this as my classifier for my model.

To begin I implemented common aspects of other sentiment analysis systems and also investigated my own hypothesis for aspects and analysed how these aspects affected the accuracy of the system. For control I keep the classifier as naive bayes while investigating feature extraction models. This means that the classifier is the dependent variable and the feature extraction methods are the independent variable. I change the feature extraction methods to measure the effects on the accuracy of the classifier. For a more detailed analysis I also record the run time of the system and the most informative words.

Then once I have finished experimenting on the feature extraction models I will conclude my most accurate model from the results and implement this model to investigate the effect that spell checking the reviews first will have. To do this by implementing spell check and then running the model to determine the accuracy.

7.1 Data Collection

Data collection is the process of retrieving the data and then formatting the data ready to extract features. Lots of consumer product selling websites such as Amazon, Ebay and such provide large corpus's of reviews. However woolf found that typically an Amazon review is usually between 100 - 150 characters. (Woolf, M. 2014) Were as Hung & Qui found that a yelp review averages a word count of around 20-30 words for each star class(1-5), and the word lengths mainly range between 0 and 150. (Hung & Qiu 2014)

In response to this, as an average yelp review seemed to hold more information I chose to use the yelp dataset. This dataset provides a corpus of 2.2 million user written reviews for businesses. However I found that the dataset had 56467 errors which decreased the dataset size by 2.57%. This means the data set now contains 2143533 reviews.

7.2 Labelling the Data

As this project concerns a system that needs to classify text written reviews as positive or negative this means that the data needs to also be in the same format, so the reviews need to be labelled as positive or negative. However they are currently labelled 1-5. This is a separate classification issue of whether a rating of 3 is positive or negative, 1-2 is clearly negative and 4-5 is clearly positive, however for 3 it is not so clear. One argument is that a review of rating 3 is generally considered positive in respect to reviews. This is probably with 5 being absolute positive (the best rating) and 1 being absolute negative (the worst), then 3 is positive because its greater than half way which is 2.5. An opposing argument could be that 3 is exactly in the middle and is neutral or satisfactory which in terms of business could be viewed negatively as satisfactory is not good enough.

To solve this problem and to uphold validity of my results I decided to remove all reviews that was ranked 3 and did not use them as training for the system.

	Data set size	**%**
Original unedited data set(ODS)	2200000 Reviews	100
ODS, minus errors	2143533 Reviews	97.43
ODS, minus errors, minus ratings of 3	1881162 Reviews	85.51

Table 5: A table showing the change of the data set size.

Training Data & Test Data

The dataset was split into two categories. These were test data and training data. The training data was used to train the classifier and the test data was used to determine the accuracy. The proportions of this data are below:

Data Type	Percentage of original data
Training Data	75%
Test Data	25%

Table 6: A table showing the size of the test & training data.

7.3 Feature Extraction

Jackson explains that text feature extraction is the process of transforming what is essentially a list of words into a feature set that is usable by a classifier (Jackson 2010).This means that feature extraction involves extracting attributes of the text that holds value in terms of a given context, which can then be given to the classifier for classification. In the context of this project feature extraction involves the extraction of the attributes of the reviews that may hold value in terms of its polarity. Features are extracted and stored in a feature vector that is passed to the classifier for classification based on those features. Before I explain the feature extraction models I investigated I will discuss the common problems with the task for all feature extraction models and how these were solved.

Tokenization

Before I can extract any features of the text I need to be able to analyse the individual words used within the text. The text currently contains punctuation, I do not need this information as I am performing sentiment analysis at the document level, I have no use for sentences. To do this I use a tokenizer. One explanation of a tokenizer is that it segments a stream of characters into meaningful units called tokens. (Berman 2013) Another explanation is that a tokenizer is an algorithm used to divide a string into sub strings defined by the tokenizer. (NLTK Project 2016)

Many different tokenizers exist, and each one works slightly differently and produces different results. This depends on the algorithm by which the tokenizer divides the text into tokens. I will now discuss some tokenizers, how they work and what results they produce. Results were produced from textprocessing.com. On the text: *"The food tasted nice, however was salty."*(text-processing 2016).

Whitespace tokenizer - Splits the text into tokens by separating each token when a white space character is found. (NLTK Project 2016)

Figure 4: Example of the whitespace tokenizer (text-processing 2016)

WordPunct tokenizer - Tokenize a text into a sequence of alphabetic and non-alphabetic characters, using the regular expression `\w+|[^\w\s]+` (NLTK Project 2016).

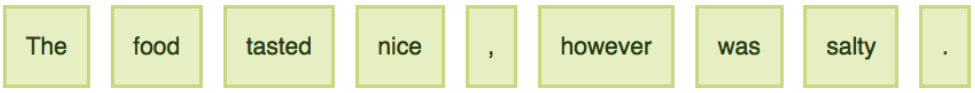


Figure 5: Example of the WordPunct tokenizer (text-processing 2016)

Regexp tokenizer - Allows you to define your own regular expression by which the tokenizer will separate the text into tokens. This is the tokenizer I chose to utilize because unlike most traditional tokenizers I needed to remove all punctuation. The regular expression used was: `((?<=[^\w\s])\w(?=[^\w\s])|(\W))+` where: `\w` = any alphanumerical character and underscore, `\s` = any non-whitespace character. This removes all punctuation.

Figure 6: Example of the RegExp tokenizer (text-processing 2016)

Unigrams, Bigrams & Trigrams

Each token is an example of a unigram. A unigram is of size 1 and consists of one token or word. However these tokens can be used to create a bigram or trigram. a bigram is of size 2 and consists of two sequential tokens, and as expected a trigram is of size 3 and consists of three sequential tokens. For example:

Unigram: [like]
Bigram: [not like]
Trigram: [not like food]

Note that these do not consist of any random tokens. sequential order is important. By concatenating tokens we can create bigrams and trigrams these will be used and further discussed in the n-gram model however bigrams are also important for handling negation.

Negation

Another common problem for a sentiment analysis system is negation. For example *"I like food"* implies a different meaning to *"I do not like food"*. This is because the word *"not"* is a negating word so any word that follows it becomes inverted in terms of polarity. If we store the tokens as individual words (unigrams) then we will be feeding the classifier false information, which means the classifier will probably be less accurate due to a negative being stored as a positive, causing confusion.

Before I can handle negation the system needs to act on the negating word: *"not"*. The tokenizer I am using does not handle negating words and distorts them when used with an apostrophe for example the word: *"don't"*. As it splits the word into the tokens [don], [t]. To solve this problem I changed any instance of a [t] token to be [not] and then removed the last character of the previous word. this then gave the tokens [do], [not].

(Rain 2013) performed a similar project and solved this problem by checking each word for a preceding negating word and if present then the word is stored as a bigram for example [not like]. The author also concluded that the best results were obtained when checking 3 words before for a not.

This negation handler i have discussed was implemented for each feature extraction model used and used the suggestion by (Rain 2013).

7.4 Feature Extraction - Bag of Words

One piece of literature stated that the bag of words model is the simplest method of feature extraction (Jackson 2010), and as such seems like a perfect starting point to expand upon. The bag of words model consists of extracting each word from the text and storing the word in a dictionary, this list then becomes the bag of words for that piece of text. All punctuation and ordering are removed after this process. As words are stored as single values this means the sequential ordering is lost with this model.

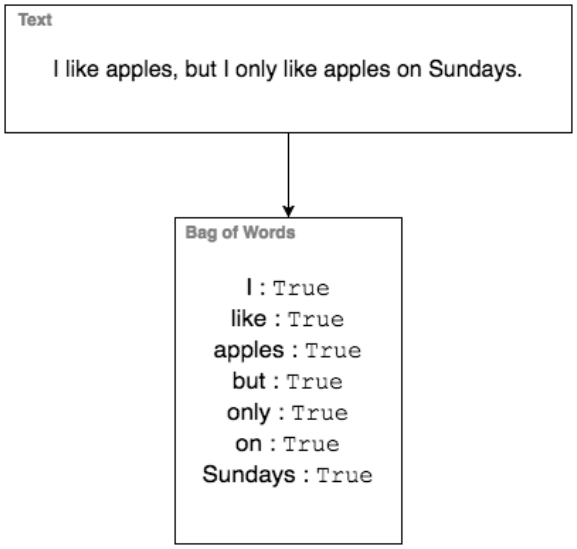


Figure 7: Example of the bag of words model

However multiplicity can be kept if you wish and you can store a count for each word.

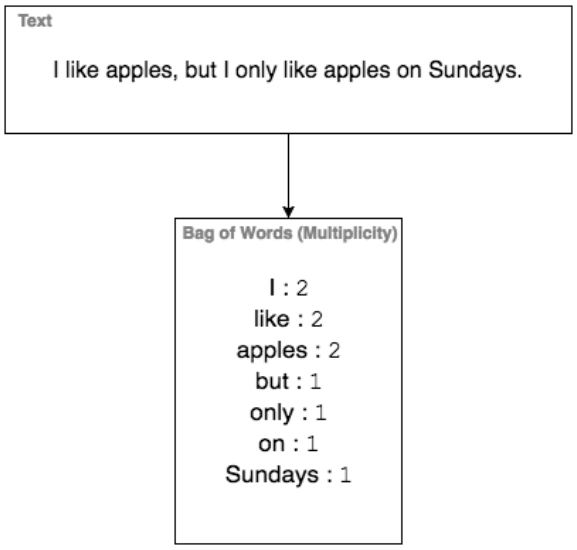


Figure 8: Example of the bag of words model with multiplicity

I used the bag of words model without multiplicity as this information would most likely only be useful if you were concerned with the polarity level where as previously mentioned we only wish to know if the text is positive or negative so we can leave multiplicity. The bag of words also contained the bigrams from the negation handler as explained previously in 7.2.

The bag of words was stored as a feature vector and was then given to the classifier for classification for each review.

7.5 Feature Extraction - Bag of valuable Words

After implementing the bag of words model, I felt the model was too basic and did not hold enough information for the classifier to adequately learn from. At the moment all the classifier knows is the words used in the text. However much of this information is useless. for example the words: *and, because, however, ...* are probably used in both negative and positive reviews and as such are not suitable to learn from.

To solve this issue I implemented a bag of valuable words. Which is to say the bag of words only contains words that hold value in terms of the context. In this case a the bag of valuable words would be the bag of words that hold some opinion value whether that be positive or negative. As before this bag of valuable words was then translated into a feature vector and given to the classifier as input.

Stopwords

Stop words are extremely common words that appear in documents that would appear to be of little value in helping selecting documents. (Manning, Raghavan & Schütze 2008) Or more simply put stopwords can be seen as the most used or common words in a language.

Before extracting the opinion words it seemed logical to first remove the obvious uninformative words that we know offer no value. To do this I used the stopwords corpus from the NLTK module this allowed me to remove all the stop words from the tokens by checking if the word was in the stopwords corpus. If the word existed in the stopwords corpus then this token was removed and not added to the bag.

SentiWordNet

To implement the bag of valuable words, I need to be able to extract the words that hold opinion value do this I used the SentiWordNet corpus from the nltk module. SentiWordNet is described on their website as "A lexical resource for opinion mining. SentiWordNet assigns to each synset of WordNet three sentiment scores: positivity, negativity, objectivity. " (SentiWordNet 2016). This is used by searching for a word in the SentiWordNet database and if present it will return sentimental information about that word such as its positive score and negative score. Using this information I extracted

words that held either a positive score or a negative score.

A problem i came across was some words not being found when searching SentiWordNet. To overcome this problem I used WordNet which SentiWordNet is dependant upon. I used WordNet to search for a word, if a word was found I then stored the offset value and used this value to find the word in the SentiWordNet database from the package. If a word i searched for in SentiWordNet was found It was added to the bag as long as two conditions were met:

1. The words holds a positive score or negative score that is greater than 0.

2. The positive score and negative score are not equal.

Also the negation handler was used on these words to transform them into a bigram if a *"not"* was present 3 or less words before where the word appeared in the review. This means negation was only used on a word that was present in SentiWordNet.

To search for the word in SentiWordNet the words part of speech value was required which brings me on to the next problem.

Part of Speech tagging

In the English language words can be considered the smallest elements that hold meaning. Based on their use and function, words are categorized into parts of speech. (part of speech 2015)

Part of speech (pos) tagging is software that reads text and assigns parts of speech to each word in the text for example a noun, verb or adjective. (Stanford NLP Group 2015)

For my implementation i used the default pos tagger from the NLTK module. After using this i was able to correctly search for words in SentiWordNet.

7.6 Feature Extraction - Attributed Bag of Valuable Words

Attributed bag of valuable words is similar to the bag of valuable words model however it has additional information which is attributes of the bag of valuable words. these attributes are:

1. Overall positive score of the bag
2. Overall negative score of the bag
3. Length of the bag

When negation occurred this meant that the scores needed to be inverted to respond to the negation. This is because the original scores are for the literal word however we need the scores for the opposite word of the literal word. This was achieved by swapping the scores round so the original positive score became the new negative score, and the old negative score became the new positive score.

7.7 Feature Extraction - n-gram model

The bag of words model removes most sequential ordering apart from when negation is handled and a bigram is generated. However as my first hypothesis is that more informative feature will in turn increase the accuracy of the classifier then it seems relevant to use this information. The n-gram model will help me extract this sequential information.

An n-gram is a continuous sequence of n items from a given sequence of text or speech. n-grams are particularly useful and commonly used to predict the next word in a sequential group of words. For example given the history of words what is the probability that a certain word will appear next.(Broder, Glassman, Manasse & Zweig. 1997). As discussed earlier an n-gram where n=1 is a unigram, an n-gram where n=2 is a bigram and an n-gram where n=3 is a t rigram

The n-gram model is useful however a drawback that can be seen from a first glance is the overflow of information. To elaborate typically when using the n-gram model in computer linguistics you calculate all the possible combinations of n-grams and this information is used to calculate a probability. This may be useful in statistics however in this case if i was to calculate all these combinations and use them as features then the classifier would likely be overwhelmed with information and will get distracted. To solve this problem I only created a bigram or trigram for opinion words, which is to say a word that is is not a stop word and present in SentiWordNet, as explained in 7.5.

I investigated both bigrams and trigrams separately and recorded the accuracy and runtime for both. For the bigram model I stored a bigram for each opinion word found in the review. I stored two bigrams for each opinion word found, one contained the word that preceded the opinion word followed by the opinion word its self, the other contained the opinion word its self followed by the word that post ceded the opinion word.For the trigram model I simply stored each opinion word found as a trigram that contained the word that preceded the valuable word followed by the opinion word its self then followed by the pre ceding word. Since bigrams and trigrams would include any preceding *"not"* words I did not need to include the negation handler.

7.8 Feature Extraction - Spell Check

As discussed by Callen Rains, they talked about how important features can be missed due to a difference of spelling (Rains 2013). After implementing the bag of words model this thought became more apparent as you can see in Table 11, one spelling mistake was found multiple times and consequently was one of the most informative features. This means many people miss spelled the same word.

I used the pyenchant module from python to perform spell checking. This is based on the aspell module. To use this module I had to download a dictionary file from the OpenOffice project. I found that Yelps website traffic is primarily from the US, specifically 91.6% of the traffic is from the US (Quantcast 2016). For this reason I chose the American dictionary rather than the English dictionary. Pyenchant works by first checking if a word is misspelled if the result is False which means a spelling mistake is present then that word can be used to search for suggestions of a correction.

Spell check was added as a phase to the bag of valuable words model it was added to this model because firstly it was found to have the highest accuracy as shown in Section 8. In trying to achieve the best accuracy we can it seems logical to add it to that particular model. Secondly it was because when implementing spell check efficiency had to be considered. The runtime is already very large due to the use of SentiWordNet, adding a spell check on top of this could take days to run for every word in every review for 2.2 million reviews. For this reason I decided to spell check the word after the opinion words had been extracted. This had the benefit of greatly reducing the amount of times words were spell checked because words that were not going to be used were not spell checked. However due to the spell check occurring after the check for an opinion word this meant that a word would still contain the error when the opinion word is searched for in SentiWordNet, meaning it may not be found when it could have if spelled correctly.

7.9 Classification - Naive Bayes

Document classification is the formal name given to the problem of classifying documents. Once we have a feature vector for the text we can utilize a probabilistic classifier.

A probabilistic classifier is an instance of machine learning where an algorithm is used on a given set of sample data (the feature vector) that then predicts a probability distribution over the set of classes (postive and negative). This probability distribution is then used to predict the class that the text belongs to. (Hastie, Tibshirani and Friedman 2009)

Naive bayes is an example of a probabilistic classifier that uses bayes theorem to predict the probability that a given feature set belongs to a particular label. (Jackson 2010)

The formal definition is:

$$P(label|features) = \frac{P(label) * P(features|label)}{P(features)}$$

Figure 9: Naive Bayes Formal Definition (Jackson 2010: 171)

P(label) "is the prior probability of the label occurring, which is the same as the likelihood that a random feature set will have the label. This is based on the number of training instances with the label compared to the total number of training instances. For example, if 60/100 training instances have the label, the prior probability of the label is 60 percent." (Jackson 2010: 171)

P(features|label) "is the prior probability of a given feature set being classified as that label. This is based on which features have occurred with each label in the training data." (Jackson 2010: 171)

P(features) "is the prior probability of a given feature set occurring. This is the likelihood of a random feature set being the same as the given feature set, and is based on the observed feature sets in the training data. For example, if the given feature set occurs twice in 100 training instances, the prior probability is 2 percent." (Jackson 2010: 171)

P(label|features) "tells us the probability that the given features should have that label. If this value is high, then we can be reasonably confident that the label is correct for the given features." (Jackson 2010: 171)

For my classification method I used the NLTK module package called classify. I implemented the existing naive bayes classifier from the package. Once the feature vector was generated from the feature extraction model then I passed this vector (which is labelled as positive or negative) as a parameter to the classifier and the classifier trained as described above. Once this training was finished I used the accuracy method from the classify package. I gave the pre trained classifier and a pre classified feature vector as parameters to the method and it returned the accuracy as specified in section 8.

During the implementation I encountered a crippling problem: Because the data set was so large there lots of reviews and hence lots of feature vectors that needed to be stored in random access memory (RAM). This was due to the classifier only accepting data from variables, which are stored in RAM. To solve this problem I edited the source code file for the naive bayes classifier from the NLTK module. I wrote the feature vectors to a file using the pickle python module that serialized the objects. I then changed the variable parameter to a file name or more specifically a string parameter. Next I added a loop that looped through the file that contained the feature vectors. This loop incorporated the use of a buffer to ensure the variable did not get too large.

Finally I used a method called: show most informative features, which ordered the features in rank from highest to lowest dependant on their respective probability distributions.

7.10 Python Modules

Below is a table showing which specific python modules were used and for what aspect of the model implementation. For the NLTK module i will also discuss each specific package of the module i used.The usage will be explained in the appropriate sections of the report for all these packages.

Module.Package	**Usage**
json	This package was used in every model. The yelp data set was in the format of json data (Java Script Object Notation) and I needed to used the json package to access the review text from the data.
NLTK.classify	The classify package was used in every model and was used for classification. This package contained an implementation of the naive bayes classifier that was ready to use on a feature vector, this was used for my classification method for all models. Also the package contained a built in accuracy checker that allowed me to easily record the accuracy of the system by passing it the pre trained classifier and a feature vector of correctly classified test data.
NLTK.corpus.stopwords	Used in the bag of valuable words model for feature extraction. Was used to extract words of less opinion value (most used words in a language). Contained a corpus of all the stopwords in the English language.

Table 7: A table documenting the usage of python modules and packages

Module.Package	Usage
NLTK.pos_tag	Used in all models that used SentiWordNet: bag of valuable words, attributed bag of valuable words and both n-gram models. this package was used to return the positioning of words in the text. This information was useful when extracting the opinion words as this gave SentiWordNet more information to generate a pos or neg score.
NLTK.sentiwordnet-corpus-reader	This package was used in all models that extracted opinion words in the feature extraction method. These were bag of valuable words, attributed bag of valuable words and both n-gram models. It read the SentiWordNet file that contains assigns to each synset of WordNet three sentiment scores: positivity, negativity, objectivity. I specifically needed SentiWordNet as I needed a positive and negative score for individual words in the text to implement the bag of valuable words model. This package depends on WordNet.
NLTK.tokenize	Used in all models to tokenize the review text. gives access to lots of different tokenizers as discussed I used the regexp tokenizer.
NLTK.wordnet	Used for all models that used SentiWordNet. Word net is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets). This is used specifically for extracting the opinion words from the text, because SentiWordNet depends on WordNet.

Table 8: A table documenting the usage of python modules and packages

Module.Package	Usage
numpy	This module was a dependency for the NLTK module to make complex scientific calculations. For example probability distributions that were specifically calculated in the NLTK.classify package.
os	This module was used in every model. It allowed python to interact with the OS (Operating System) which enabled me to delete files at the start of each instance of an analysis to remove large redundant files.
pickle	Used in every model, this was used for object serialization which was necessary because lots of the data had to be stored in files.
pyenchant	Used in the spell check model to check if a word was misspelled and if so to provide correction suggestions for that word.
time	The time module was used in every model. This allowed me to take timestamps which were used when calculating the model implementations runtime.

Table 9: A table documenting the usage of python modules and packages

8 Experimental Results & Discussion

8.1 Measurements

Before discussing and analysing the results it makes sense to first explain the measurements that were taken, why they were taken and finally how they were taken.

Accuracy

The first and most important measurement recorded is the accuracy of the model. This is required to analyse the difference in accuracy between the models to investigate which feature extraction method provides the best results. I record the accuracy by implementing the model and waiting for it to complete, I then used the training data as specified in section 7.2 of this document. The test data is already correctly labelled as neg or pos so I used the NLTK classify package which classifies each test data and then compares the models classification result with the actual classification and determines if it is correct(True) or incorrect(False). Using this information the formula below is then used to calculate the accuracy.

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn}$$

Figure 10: Accuracy Formal Definition

tp is a true positive. This means the prediction from the classifier was positive and was correct.

tn is a true negative. This means the prediction from the classifier was negative and was correct.

fp is a false positive. This means the prediction from the classifier was positive and was incorrect.

fn is a false negative. This means the prediction from the classifier was negative and was incorrect.

tp + tn is all the correct predictions from the classifier.

fp + fn is all the incorrect predictions from the classifier.

tp + tn + fp + fn is all the predictions made from the classifier.

Accuracy difference

The next measurement recorded is the difference in accuracy between the different models. This measurement is taken because to show which model was most accurate and how it compares to the other models. This was calculated by the formula below.

s

$$AccuracyDifference = CurrentModelAccuracy - OtherModelAccuracy$$

Figure 11: Accuracy Difference Formal Definition

Current Model Accuracy is the accuracy of the current model being compared.

Other Model Accuracy is the accuracy of the other model that the accuracy difference is being calculated in respect to.

Probability Distribution

The final measurement taken is the probability distribution of each feature. This is where each feature is given a probability to each of the possible outcomes (positive or negative). The NLTK classify package does this automatically and then has a method called show most informative features, this lists the features in the order of their probability distributions from highest to lowest. This information is used to understand in more detail how the model performed.

Runtime

The final measurement recorded is the runtime of the model. This measurement is taken to show the efficiency of the model and how it differs across the different models. This measurement is recorded by taking a time stamp before the model implementation starts to compute and then taking a following time stamp once the model has completed, and then calculating the difference between the time stamps.

$$Runtime = EndTime - StartTime$$

Figure 12: Runtime Formal Definition

8.2 Bag of words

Bag of words was the most simplest feature extraction model and was the first feature extraction model I investigated.

Measurement	Value
Accuracy(Percentage):	61.6372933268%
Accuracy compared to bag of valuable words:	-28.5897578882%
Accuracy compared to attributed bag of valuable words (Percentage):	-27.8959769501%
Accuracy compared to n-gram bigrams (Percentage):	-27.7959629898%
Accuracy compared to n-gram trigrams (Percentage):	-22.5109389145%
Accuracy compared to spell checked bag of valuable words (Percentage):	-28.5911982638%
Runtime(Seconds):	1516s

Table 10: A table of results for bag of words.

Feature	Probability Distribution
not returning	neg : pos = 102.6 : 1.0
not apology	neg : pos = 102.3 : 1.0
arrogantly	neg : pos = 81.1 : 1.0
ontrac	neg : pos = 78.7 : 1.0
not acknowledging	neg : pos = 78.2 : 1.0
fraudulently	neg : pos = 74.9 : 1.0
not refund	neg : pos = 73.9 : 1.0
not acted	neg : pos = 70.8 : 1.0
mismanagement	neg : pos = 70.8 : 1.0
not reimbursed	neg : pos = 68.8 : 1.0
discusting	neg : pos = 68.4 : 1.0

Table 11: A table of top 11 most informative features for bag of words.

Negative features are more informative than positive features.
From the top 11 most informative features we can identify that they are all negative. In fact out of the top 100 most informative features for bag of words just one was a positive feature which was *"foodgasm"*. Maybe people find it easier to describe a negative experience than a positive experience because your more likely to remember something that upset you and the results are maybe a reflection of this human behaviour. On the other hand maybe positive experiences are more complex in variety than negative experiences and as such its harder for the classifier to learn them as they can contradict one another. For example we are very clear on what we dislike for example (rudeness, lateness, no refunds, ect) and most people will likely agree on these. However positives such as (tasty food, cheap alcohol, fast car) are more subjective for example food that tastes good to me may not taste good to someone else, cheap alcohol for me may not be cheap for someone unemployed, a fast car may be positive in my eyes but not for a retired pensioner. These are more complex.

Negation is Informative
From the top 11 most informative features we can identify that 6 are negation bigrams, and the top 2 are both negation bigrams This is more than half which means negation was an important factor in identifying informative features. maybe this is because people tend to negate a positive word than write the negative equivalent for example people may write *"not like"* instead of *"dislike"*. people might be doing this because its easier or because its consistent because sometimes their is no obvious negative equivalent.

Spelling mistakes are present
From the top 11 most informative features we can identify that 1 had a spelling mistake, this was *"ontrac"*. Also Number 11 which is not in the table was *"discusting"*. Which means several people misspelled the same word. This will reduce the accuracy because *"discusting"* & *"disgusting"* are treated as separate features when they are technically the same as both writers meant the same thing.

Anomaly's are present
From the top 11 most informative features we can identify one anomaly. this is *"ontrac"*. Most likely this seems to be a spelling mistake of the word *"on-track"*, however this should be a positive feature if that was the case.

More accurate than random

The accuracy for bag of words was 62% when rounded up to nearest whole number. This is good in the sense that it is better than if you guessed. statistically you would expect roughly a 50% accuracy from guessing.

Efficient Runtime

The runtime was 1516s which is the equivalent of just over 25 minutes. Considering the system had to analyse 1881162 Reviews this is impressive.

Not accurate enough

Considering that the aim is to teach a machine to classify the text as much like a human as possible then 62% accuracy is not good enough. We also need to remove the anomaly's.

8.3 Bag of valuable words

Bag of valuable words was the second feature extraction method I investigated and is an extension of the bag of words model, the bag only contained words that were deemed valuable which is to say they were not a stopword and was present in SentiWordNet, with a pos score and neg score above zero and not equal to one another.

Measurement	**Value**
Accuracy(Percentage):	90.227051215%
Accuracy compared to bag of words:	+28.5897578882%
Accuracy compared to attributed bag of valuable words (Percentage):	+0.6937809381%
Accuracy compared to n-gram bigrams (Percentage):	+0.7937948984%
Accuracy compared to n-gram trigrams (Percentage):	+6.0788189737%
Accuracy compared to spell checked bag of valuable words (Percentage):	-0.0014403756%
Runtime(Seconds):	77763s

Table 12: A table of results for bag of valuable words.

Feature	Probability Distribution
arrogantly	neg : pos = 80.8 : 1.0
accusing	neg : pos = 78.8 : 1.0
fraudulently	neg : pos = 76.7 : 1.0
poorest	neg : pos = 72.6 : 1.0
not back	neg : pos = 64.1 : 1.0
blandest	neg : pos = 62.1 : 1.0
unethical	neg : pos = 61.4 : 1.0
unprofessional	neg : pos = 58.0 : 1.0
rudest	neg : pos = 55.2 : 1.0
not vacuumed	neg : pos = 55.1 : 1.0
unacceptably	neg : pos = 49.7 : 1.0

Table 13: A table of top 11 most informative features for bag of valuable words.

More Accurate than bag of words

Overall the percentage for bag of valuable words was about 90% and you can see that the model was about 46% more accurate than bag of words from table 6. This maybe supports the opinion that uninformative features distract the classifier from the informative ones creating noise. This is further backed up from looking at table 7: the top 11 most informative features. Because we can see that although the accuracy was higher for bag of valuable words the highest probability distribution for bag of valuable words was (80.8 : 1.0), is much lower than the highest for bag of words (102.6 : 1.0). This means that with lower weighted or quality informative features the model still achieved a higher accuracy leading to the conclusion that the quality of features is more important than the quantity.

Probability Distribution is on average lower

The most informative feature for the bag of valuable words model held a probability distribution of [neg : pos = 102.6 : 1.0] where as the bag of valuable words models highest was [neg : pos = 102.6 : 1.0]. However the model was still more accurate despite having what seems like less informative features than the original bag of words model.

Negation is still useful

From the top 11 most informative features we can identify that 2 are negation bigrams. This means that negation features were useful but not as useful when compared to the original bag of words model where 6 of the top ten most important features were negation bigrams.

Negative features are the most informative

All the top 11 most informative features where negative.

Accuracy costs efficiency

The runtime is much longer than the original bag of words model. The runtime was 77763s which is the equivalent of just over 21 hours. This is about 51 times as long as the original bag of words model. This points to the reasoning that to gain accuracy you may have to sacrifice efficiency. This was a big draw back to the models solution. This is most likely caused by SentiWordNet because that was the only change from the previous model. Constant reading of the SentiWordNet file to identify opinion words was most likely the cause of the slow runtime. Implementing a solid state hard

drive would increase the file read and write time which could provide some improvement. A cluster could also be used to provide more processing power or a sample of the data could be taken to reduce the data set size. These improvements could increase the efficiency.

Features are more expressive than bag of words model
From Table 15 we can see that the features were more expressive in terms of opinion than the previous model. The identification of opinion words using SentiWordNet seems to be fairly successful so far.

8.4 Attributed Bag of valuable words

Attributed bag of valuable words was the third feature extraction method I investigated and is a further extension of the bag of words model and the bag now also contained the number of positive and negative words and also a positive and negative score from SentiWordNet.

Measurement	Value
Accuracy(Percentage):	89.5332702769%
Accuracy compared to Bag of words (Percentage):	+27.8959769501%
Accuracy compared to bag of valuable words (Percentage):	-0.1000139603%
Accuracy compared to n-gram bigrams (Percentage):	+0.1000139603%
Accuracy compared to n-gram trigrams (Percentage):	+5.3850380356%
Accuracy compared to spell checked bag of valuable words (Percentage):	-0.6952213137%
Runtime(Seconds):	78029s

Table 14: A table of results for attributed bag of valuable words.

Feature	Probability Distribution
arrogantly	neg : pos = 80.8 : 1.0
accusing	neg : pos = 78.8 : 1.0
fraudulently	neg : pos = 76.7 : 1.0
poorest	neg : pos = 72.6 : 1.0
neg_score = 9.278	neg : pos = 64.4 : 1.0
not back	neg : pos = 64.1 : 1.0
blandest	neg : pos = 62.1 : 1.0
unethical	neg : pos = 61.4 : 1.0
unprofessional	neg : pos = 58.0 : 1.0
rudest	neg : pos = 55.2 : 1.0
not vacuumed	neg : pos = 55.1 : 1.0

Table 15: A table of top 11 most informative features for attributed bag of valuable words.

Less Accurate than previous bag of valuable words model
This model was again more accurate than the original bag of words model however was less accurate than the bag of valuable words model by 0.6937809381%. This means that the additional features extracted actually decreased the accuracy of the classifier maybe this was due to the variation between the scores for each review meaning the classifier got distracted by this information.

Probability Distribution is on average lower
The most informative feature for the bag of valuable words model held a probability distribution of [neg : pos = 102.6 : 1.0] where as the bag of valuable words models highest was [neg : pos = 102.6 : 1.0]. However the model was still more accurate despite having what seems like less informative features than the original bag of words model.

Negation is still useful
From the top 11 most informative features we can identify that 2 are negation bigrams. This means that negation features were useful but not as useful when compared to the original bag of words model where 6 of the top ten most important features were negation bigrams.

Negative features are the most informative
All the top 11 most informative features for this model where negative.

Runtime is longer than bag of valuable words
The runtime was 78029s which is slightly longer than the bag of valuable words model. This was what was expected due to generating additional features. However the trade of efficiency in this case did not improve accuracy.

8.5 N-gram Bigrams

The bigram model was the fourth feature extraction method I investigated and involved using SentiWordNet to extract opinion words and for each opinion word found a bigram was created on either side. The first bigram contained the word that preceded the opinion word and the opinion word itself, the second bigram contained the opinion word and the word that post ceded the opinion word.

Measurement	**Value**
Accuracy(Percentage):	89.4332563166%
Accuracy compared to Bag of words (Percentage):	+27.7959629898%
Accuracy compared to bag of valuable words (Percentage):	-0.7937948984%
Accuracy compared to attributed bag of valuable words (Percentage):	-0.7937948984%
Accuracy compared to n-gram trigram (Percentage):	+5.2850240753%
Accuracy compared to spell checked bag of valuable words (Percentage):	-0.7952352740%
Runtime(Seconds):	78579.34s

Table 16: A table of results for n-gram bigram.

Feature	Probability Distribution
zero stars	neg : pos = 615.4 : 1.0
give zero	neg : pos = 453.8 : 1.0
place awful	neg : pos = 275.6 : 1.0
manager rude	neg : pos = 234.1 : 1.0
worst chinese	neg : pos = 219.6 : 1.0
even deserve	neg : pos = 192.5 : 1.0
back horrible	neg : pos = 188.9 : 1.0
absolutely worst	neg : pos = 185.3: 1.0
zero star	neg : pos = 183.4 : 1.0
worst customer	neg : pos = 170.5 : 1.0
never business	neg : pos = 160.0 : 1.0

Table 17: A table of top 11 most informative features for n-gram bigram.

Much larger probability distributions compared to bag of words models

This model had a large difference in probability distributions when compared to the previous bag of words models. On average this models distributions where at least 2 times larger and many of the top features distributions were 3 times as large at least. When comparing these distributions to the corresponding feature we can see that they are more domain specific and maybe this meant the features appeared more often in the reviews and as such had a higher probability distribution. For example the most informative feature: *zero stars* is most likely specific to the domain of consumer reviews. The feature: *worst customer* is most likely specific to consumerism.

Sequential ordering did not improve the model design

This model was similar to the bag of valuable words model apart from each opinion word was stored as a bigram. This was a solution to give the classifier sequential information to learn from the text. This model however achieved a difference in accuracy of -0.7937948984% when compared to the bag of words model, which used unigrams only. Meaning that in this project sequential information did not improve the accuracy. However this could have been due to the use of SentiWordNet meaning only opinion words were used as features.

Negating features were no longer in the top 11 most informative features

From Table 17 we can see that there were no negating features in the table. This could mean that the sequential features are more useful than negation features.

Similar runtime to bag of valuable words

From Table 21 we can see that all the runtime was very similar to the bag of valuable words models runtime from Table 28. This leads to the conclusion that SentiWordNet and the identification of opinion words is what is causing the large increase in runtime.

Domain specific features

From Table 21 we can see that all the runtime was very similar to the bag of valuable words models runtime from Table 28. This leads to the conclusion that SentiWordNet and the identification of opinion words is what is causing the large increase in runtime.

Negative features are most informative

From Table 17 we can see that all the features and their probability distributions were negative.

8.6 N-gram Trigrams

The trigram model was the fifth and final feature extraction method I investigated and involves using SentiWordNet to extract opinion words and for each opinion word found one trigram was created by concatenating the tokens either side of the opinion word.

Measurement	Value
Accuracy(Percentage):	84.1482322413%
Accuracy compared to bag of words (Percentage):	+22.5109389145%
Accuracy compared to bag of valuable words (Percentage):	-6.0788189737%
Accuracy compared to attributed bag of valuable words (Percentage):	-5.3850380356%
Accuracy compared to n-gram Bigrams (Percentage):	-5.2850240753%
Accuracy compared to spell checked bag of valuable words (Percentage):	-6.0802593493%
Runtime(Seconds):	78125s

Table 18: A table of results for n-gram trigram.

Feature	Probability Distribution
give negative stars	neg : pos = 222.9 : 1.0
not even deserve	neg : pos = 174.2 : 1.0
not even apologize	neg : pos = 138.1 : 1.0
one worst experiences	neg : pos = 121.8 : 1.0
really wanted like	neg : pos = 115.2 : 1.0
not recommend anyone	neg : pos = 110.5 : 1.0
would never recommend	neg : pos = 110.0 : 1.0
food horrible service	neg : pos = 107.4 : 1.0
never recommend anyone	neg : pos = 105.6 : 1.0
one star would	neg : pos = 102.0 : 1.0
place still business	neg : pos = 102.0 : 1.0

Table 19: A table of top 11 most informative features for n-gram trigram.

Smaller probability distributions than n-gram bigram
This model had a lower accuracy and probability distributions than the n-gram bigram model. By looking at the most informative features table and their respective probability distributions we could make the conclusion that the lower probability distribution is caused by the terms being more specific and less likely to keep reappearing in the reviews, causing a lower probability distribution, maybe 3 consecutive tokens is too specific.

Better understood domain specific features
From Table 19 we can understand that the features contained much more domain specific terminology than the previous models. The bigram model did provide domain specific features but were limiting and hard to make sense of. However with this model the trigrams created features that are more human readable and easier to understand how customers feel about the business or product. For example the features *not even apologize, food horrible service, not recommend anyone*. If this was applied on one business for all their reviews we would be able to understand the worst aspects of their business for example their food service is horrible, their customers are not recommending the business to others and also that they are rude and do not apologize for mistakes. This means this model could be useful for businesses to understand the worst aspects of their business.

8.7 Spell check

The final model was adding a spell checker on the data before extracting the bag of valuable words.

Measurement	Value
Accuracy(Percentage):	90.2284915906%
Accuracy compared to bag of words (Percentage):	+28.5911982638%
Accuracy compared to bag of valuable words (Percentage):	+0.0014403756%
Accuracy compared to attributed bag of valuable words (Percentage):	+0.6952213137%
Accuracy compared to n-gram bigrams (Percentage):	+0.7952352740%
Accuracy compared to n-gram tigrams (Percentage):	+6.0802593493%
Runtime(Seconds):	7876s

Table 20: A table of results for spell checked bag of valuable words.

Feature	Probability Distribution
arrogantly	neg : pos = 80.8 : 1.0
accusing	neg : pos = 78.8 : 1.0
fraudulently	neg : pos = 76.7 : 1.0
poorest	neg : pos = 72.6 : 1.0
not back	neg : pos = 64.1 : 1.0
blandest	neg : pos = 62.1 : 1.0
unethical	neg : pos = 61.4 : 1.0
unprofessional	neg : pos = 58.0 : 1.0
rudest	neg : pos = 55.2 : 1.0
not vacuumed	neg : pos = 55.1 : 1.0
unacceptably	neg : pos = 49.7 : 1.0

Table 21: A table of top 11 most informative features for spell checked bag of valuable words.

Spell check provided a trivial improvement
This model had the highest accuracy overall only beating the original bag of valuable words that did not have spell check by 0.0014403756. This means that the hypothesis of spell check providing some improvement to the model was true in the case of bag of valuable words combined with the naive bayes classifier. Spell check could have maybe provided a larger improvement if it was performed before the search for an opinion word as it would also increase the accuracy of the opinion word extraction using SentiWordNet. This was likely due to the reading time for the software to read the SentiWordNet file.

Probability Distribution was exactly the same
The probability distribution for this model was exactly the same as the original bag of valuable words that did not have spell check. This could mean that spell check was not too common among the reviews. Again maybe this was due to the spell check being performed after the opinion word had been searched for and found. Because if a spelling mistake occurred then SentiWordNet would not find the positive and negative scores and therefore the word was not an opinion word and discarded.

Runtime was not much longer than the original model
The runtime was not much longer than the original bag of valuable words that did not have spell check. Again this leads to the conclusion that spell check was not performed much.

9 Conclusion

Overall the results from the project were successful in that all models achieved above a 50% random baseline, which is better than guessing and the most accurate model is higher than the literature I reviewed. The most accurate sentiment analysis model I investigated was the bag of valuable words model for feature extraction in combination with the naive bayes model for classification, learning from yelp reviews to classify as positive or negative also with the inclusion of a spell checker. This generated an accuracy of 90.2284915906%. Although the most informative feature extraction model was found to be the n-gram Trigram model with more domain specific terminology contained in the individual features.

The success of this project relied mainly on the use of SentiWordNet to identify opinion words and use these only as feature vectors - implemented in the bag of valuable words model. This feature extraction model was found to improve the accuracy by 28.5897578882% when using the naive bayes classifier to classify as positive or negative. This was likely because the feature vectors contained more expressive words that held better value in the context of the classification in terms of polarity. Although this gain in accuracy was found to cost efficiency as the model had an increased runtime of 51 times as long as the original bag of words model.The slow runtime for the models that implemented SentiWordNet was most likely due to the constant reading of the SentiWordNet file to identify opinion words.

Keeping the sequential structure of these features by using n-grams such as bigrams and trigrams was found not to provide an increase in accuracy. However using bigrams and trigrams was successful for identifying common phrases that are repeated in the texts, showing to be quite domain specific terminology that could be utilized by businesses to understand their best and worst aspects of their business. Trigrams were most domain specific, however were found to be maybe too specific which created difficulties for the classifier and as result had a lower accuracy than bigrams.

It was also found that the addition of a spell checker to the bag of valuable words model provided a trivial improvement of +0.0014403756% when used with the naive bayes classifier classifying as positive or negative. However the spell check was only applied on specific words and at a late stage in the model due to efficiency problems which held back its true potential.

The last finding of this study was that negative features are more informative than positive features when using the naive bayes classifier. From

the results in section 8 we can see that all models top 11 most informative features were negative. This could be due to the fact that People find it easier to describe a negative experience rather than a positive experience in a review, or they are just more memorable.

10 Further Work

After investigating feature extraction models some successful models were found that are more accurate than the current research. SentiWordNet and identifying opinion words in the bag of valuable words model was successful with the naive bayes classifier and improved the accuracy. However this could be further investigated for different classifiers to discover if this is a reliable feature extraction model among all classification models.

The choice of using all the data could have also contributed to these high accuracy scores or the binary labelling of positive or negative rather than 1-5 could have contributed. I suggest for further work to investigate the effect of the data sample size that is used to teach and test the classifier and how this affects the accuracy. This could hopefully show if it is essential to utilise all the data, or if it could be reduced to improve efficiency. It could also be useful to investigate how these models are effected when the labelling is changed from a binary classification of positive or negative to a rating scale of 1 to 5.

11 Project Management & Reflection

11.1 Agile Development

For the management of my project I used agile methodologies and processes. I chose an agile development methodology because the project primarily involved me implementing the model by developing software. I used the user story approach from extreme programming. I started this by documenting individual tasks of the project on cards, I then implemented a storyboard and placed the cards in sections using the MoSCoW method. The MoSCoW method is where you order tasks into the groups:

1. Must Have
2. Could Have
3. Should Have
4. Won't Have

Since my project was not very broad I decided to only implement the groups Must Have and Could Have. By organizing the tasks into these groups I was able to add flexibility and creativity into my project as I could quickly add ideas to the storyboard and record them. To provide a higher level view of the project progress I also added the groups: Doing & Done. This enabled me to also record the progress of the project.

Project Storyboard

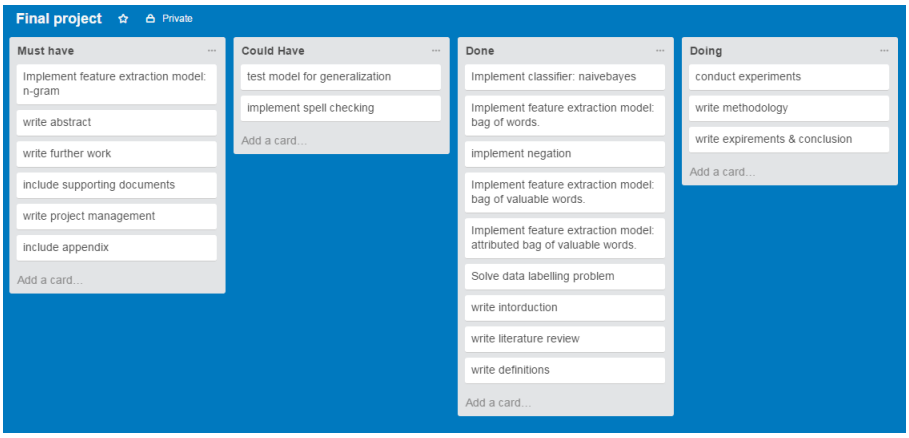


Figure 13: screen shot of the project storyboard from 01/04/2016

For time management of the project I used the SCRUM methodology. SCRUM is an agile methodology and as such fitted well with the existing framework for the project development. This methodology was implemented in the form of sprints which occurred every two weeks. During the Start of each sprint I looked at the storyboard and selected one or two user stories to complete. During the sprint I then worked on these user stories and completed them by the end of the sprint. When a sprint finished I would review the sprint and begin preparation for the next one. This enabled me to complete the work in a more structured and organized format where I worked on smaller chunks of work to solve the larger problem.

Overall I found my project development method was effective. The benefits of the method I chose was that firstly I was able to focus on the project because the method only required me to write short notes of functionality and rearrange them. Secondly it was well suited to the project and allowed me to incrementally complete the project in an organized work flow. The last and key benefit of an agile approach was its ability to handle changes in requirements which did happen throughout the project but were easy to manage. Although an improvement to this method could have been the addition of adding due dates on the storyboard cards, as this would have added a little more structure to the development time line. A trivial im-

provement that could have been the inclusion of a sprint backlog group for the user storyboard, I originally did not include one because I felt the software development was a small task with not many stories. I tended to leave backlogged stories in the Doing group until they were completed, however this meant that the storyboard was not truly representative of the current project progress. For this reason a sprint backlog could have been added to add a better overview of the project.

I learned from this experience that large projects are daunting however easily manageable with the right organization and mindset. I now understand that a good approach is it split the larger problem into a series of smaller problems and tackle them individually to work towards completing the larger problem. This makes the tasks seem much more obtainable and realistic.

11.2 Version Control

When developing any software product an important part of the development process for the product is version control.Version control allows you to store, track and organise the software code and changes made to documents or code over the development cycle.

For version control i decided on using Git. This organises your code by storing it as a remote repository than can be accessed from anywhere with an internet connections. This allowed me to work remotely from any location with an internet connection and a computer by cloning the git repository as a local copy and then making changes that can be committed back to the remote repository. Git is commonly used as a version control solution for agile development because its supports continuous integration and automated testing.

These features improved the code quality by tracking the changes to code. This improved code quality by allowing me to see which changes were potentially causing errors, by comparing the changes to an earlier working version. If needed I could then revert back to the working version and discard the changes that broke the code.

Below is a screen shot of the remote repository:

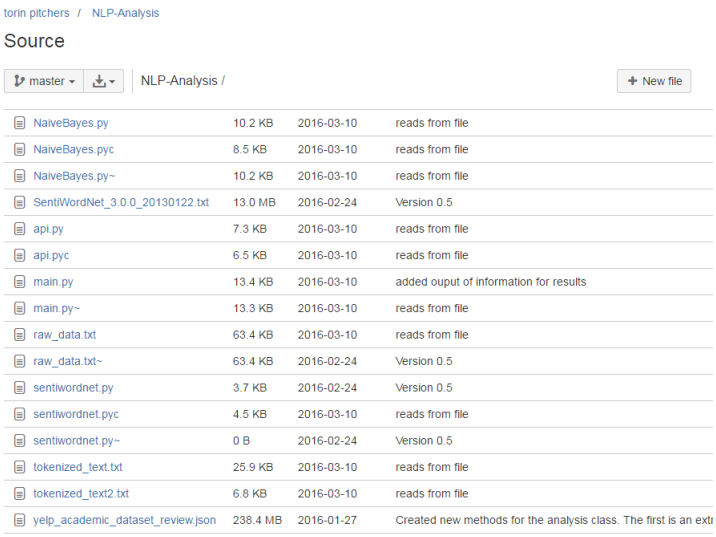


Figure 14: screen shot of the project code repository from 01/04/2016

11.3 Supervisor meetings

Throughout the duration of the project I had meetings with my supervisor. I organized these meetings when I had a problem I was finding difficult or if I had completed my current work and needed some help with the direction of the project. We had the meetings in this format because I felt confident with the project and felt that it would allow me to focus on the work whilst being able to get feedback every month to ensure I was on the right path. The meetings were very beneficial and Phil gave me lots of good feedback that was useful. For example He introduced me to unigrams and bigrams and suggested I read up on them, which I did and used them as a feature extraction method. Sometimes I listened to the feedback and investigated the ideas, however did not include them in the project as the reading did not support the use in this project. This did not happen often but I still captured the feedback given and learned why it was not suitable in that case. I documented a rough transcript after each project meeting and also included how i responded to the feedback for each meeting, these are included in the appendix and have been signed by my supervisor.

11.4 Consideration of Social, Legal and Professional Issues

Overall the project did not have many social, legal or professional issues. This was primarily due to the project not involving any participants or human interaction. The fact that the data for the reviews was anonymous and contained no sensitive user information meant that there were also no concerns of data security or privacy. Legally and professionally I had the responsibility of ensuring I conformed to the terms of the licenses of any software I included in my project. All the software used was open source and as such were all bound to an open source license that provides the rights to study, change, and distribute the software (Laurent 2004). This allowed me to use the NLYK module and pyenchant module for my project. Another professional issue I had was ensure all the results I have provided are accurate and truthful to the best of my knowledge.

11.5 Response to feedback

I responded to all feedback that my supervisor gave me throughout the project below is a table showing what feedback was received and how i responded to it. The second column contains the reference page of where the evidence of the feedback is contained in the report.

Feedback	Page	Response
Suggested a minor change to the title of the project.	97	Changed the title of the project in accordance to Phil's feedback (Included a colon).
Reduce the number of hypothesis you intend to investigate due to time constraints.	98	After reassessing the time line of the project I agreed with Phil's opinion and reduced the number of hypothesis to 2. The main one was the hypothesis that spell checking will improve the accuracy I focused on this one because as Phil stated, The results seem to agree with this so far. I then decided if I had time at the end I will attempt to generalize the model to classify other forms of opinionated text.

Table 22: A table of Feedback Response.

Feedback	Page	Response
Experimental Approach is suitable	99	After getting the green light from Phil that my experimental format for the project was correct, I continued with my experiments as normal.
The general model Is correct	100	After receiving feedback that the model diagram I created after cross referencing the literature in the area was correct I included the diagram in my final report.
Include references for yelp dataset vs Amazon dataset	101	I followed Phil's advice and included references to support the reason for using the yelp data set as training data.

Table 23: A table of Feedback Response.

Feedback	Page	Response
Be sure not to make a statement without a reference to support it, state that its your opinion if so	101	After this feedback I read my current draft and once or twice i stated my opinion without being clear. Phil's feedback made sense and I ensured I did not make any statements without a reference and if it was an opinion I made that clear by adding "In my opinion...".
Include data labelling problem in final report	103	I added this problem in the methodology section of the report and explained why I encountered the problem, how I solved the problem and why I used that particular solution.
Explain terms to the reader	104	As I agreed with Phil that the reader would likely not know what these terms meant and also combining the fact that they included some interesting problems that could be discussed I included these problems in my report and in response to Phil's specific concern I included them in a table with there definitions and a full explanation of the terms in their appropriate sections.

Table 24: A table of Feedback Response.

Feedback	Page	Response
Explain terms to the reader	104	As I agreed with Phil that the reader would likely not know what these terms meant and also combining the fact that they included some interesting problems that could be discussed I included these problems in my report and in response to phil's specific concern I included them in a table with there definitions and a full explanation of the terms in their appropriate sections.
Investigate only one classification method	106	I was happy that Phil gave this response to the classification methods because I was already concerned with the time available for the project and decided to investigate only one classification method which was the most popular from the literature: Naive bayes.
Include 11th most informative feature in the table of results	108	Initially I was not happy with changing the top ten most informative features to the top 11, however after much thought I decided to change the results to include the top 11 most informative features which allowed me to include an important feature that contained a spelling mistake that was 11th.

Table 25: A table of Feedback Response.

Feedback	Page	Response
Include Precision and Recall	109	I investigate precision and recall and agree with phil that they do provide some further measurements that can be used to investigate the models. However my main concern is the time it would take to perform the experiments again to gather these measurements. The main reason of this project is to investigate the hypothesis that spell checking will provide some improvement to the accuracy of the model. So I will continue to focus on this and the accuracy of the model and if there is time available then these measurements can be included.
Read literature and choose research method and purpose	114	After this meeting with Phil I read some literature on similar projects and found that the experimental approach we discussed is the best suited and most used. I found an interesting suggestion from (Rains 2016) that involved spell checking the data first, I agreed with this suggestion as people do sometimes make spelling mistakes so i chose this as the main purpose of the research.

Table 26: A table of Feedback Response.

Feedback	Page	Response
Use the NLTK module and learn the basics	114	I read an interesting book called Natural Language Processing with Python. (Bird, Klein & Loper). This book explained how to use the NLTK module for text classification from basic uses to advanced uses. This book taught me a lot of the basics such as tokenization, position tagging, classification and feature extraction
Start project using NLTK	116	After this meeting I started the project and achieved a 35% accuracy with a basic model that utilized bag of words learning from only 100 thousand reviews because of memory problems.
Further read literature	116	After this meeting I read more on the literature from similar projects. This enabled me to solve problems such as negation by understanding how other people solved the problem.

Table 27: A table of Feedback Response.

Feedback	Page	Response
Read into bigrams and trigrams	117	After this feedback from phil I read about bigrams and trigrams which led me to the n-gram model which is fundamental to bigrams and trigrams. I learned that these n-gram's can be used to keep the structure of the text in a limited sense by grouping tokens into groups of 2 or 3. This was eventually used as a feature extraction model
Corpus reader could improve efficiency	117	After looking into the corpus reader I concluded that although this would improve efficiency however because of the specific nature of a corpus reader I would have firstly been required to implement this myself based on the NLTK module which would take time. Secondly it would rearrange the data into a specific format that would make it harder to manipulate the data as I need to. In conclusion it was not appropriate to include a corpus reader this late in the project.

Table 28: A table of Feedback Response.

12 Appendix

12.1 Detailed Project proposal Form

300/303COM Detailed Project Proposal

First Name:	Torin
Last Name:	Pitchers
Student Number:	4955012
Supervisor:	Philip Smith

SECTION ONE: DEFINING YOUR RESEARCH PROJECT

1.1 Detailed research question

Help: Your detailed research question is the statement of a problem within the computing domain which you will address in your project. Refining the research question involves narrowing down an initial question until it is answerable using a primary research method(s) that you will conduct during the time of your project. The refined research question must not be so general that it is answerable with a yes or no answer. It must not be so broad that you would be unable to achieve a solution during your project. The key to this is BEING SPECIFIC: Narrow down the method or technology you will use, narrow down the group that the question refers to (localize a general question) If the project is still 'too big', can you think of a way to work on a part of the problem? Avoid using words that cannot be measured, by you, without a huge research budget e.g. 'effects on society', 'effects on business'. *Example:* The initial question "Does cloud computing effect business" needs narrowing down *(for a start the answer is yes) W*hat is meant by cloud computing? Or 'effect'? Or 'business', in this question? Refining this first question will involve narrowing it down to something you, personally, can measure. A refined version of this question might be: "Does implementing a cloud based voting system improve the speed of decision making in a small company in Coventry?" This refined question is implementable: You can now identify a small company to work with, document their current decision making processes, implement a cloud based voting system, compare decision making speeds over a limited time period (say 1 month) and evaluate your findings. *A small piece of genuinely new knowledge is produced.*

The Yelp open dataset consists of 1.6M reviews, containing 500K tips by 366K users for 61K businesses. There is a lot of useful information in these reviews that is very time consuming to extract manually. The aim of this project would therefore be to develop software to examine this data set.

Primarily, the research questions for this project are in the area of natural language processing:

1. What are the most common positive and negative words used in reviews?

2. How well can you guess a review's rating from its text alone? does this match the numerical rating given by the same reviewer

3. Are Yelpers a sarcastic bunch?

4. What kinds of correlations can be computed between tips and reviews: could you extract tips from reviews?

5. Can a program be developed to summarise the best and worst aspects of a business, based on user reviews alone?

1.2 Keywords

Help: Include up to 6 keywords separated by a semi-colon; what keywords are appropriate to describe your project in an online database like Google Scholar? Keywords should include the general research area and the specific technologies you will be working with. *Example.* A project that proposes a novel way of visualising large amounts of twitter feed data may have the keywords: Data visualisation; twitter; hashtags; database design; graphics libraries. For further help take a look at the ACM keywords list http://www.computer.org/portal/web/publications/acmtaxonomy

Natural Language Processing; review data set; yelp; software development; machine learning; opinion mining

1.3 Project title

Help: The project title is a statement based on your detailed research question. For example, the research question *'to what extent does a mobile application reduce the number of errors made in class registers at Coventry University in comparison to current paper based registers'* may be stated in the project title*: "A Wi-Fi driven mobile application for large group registers using iBeacons".*

computer application to analyse yelp reviews for data mining

1.4 Client, Audience and Motivation:

Help: Why is this project important? To whom is this project important? A research project must address a research question that generates a small piece of new knowledge. This new knowledge must be important to a named group or to a specific client (such as a company, an academic audience, policy makers, people with disabilities) to make it worthwhile carrying out. This is the ***motivation*** for your project. In this section you should address who will benefit from your findings and how they will benefit. Example: If you intend to demonstrate that a mobile application that automates class registers at Coventry University will be more efficient than paper based registers - the group who would be interested in knowing/applying these findings would be both academic and administrative staff at Coventry University and they would benefit by time saved and a reduction in their administrative workload. If you are making a business case for an organization explain how the organisation will benefit from your findings.

This project is important firstly because it can provide some useful information to yelp about their business's user group that leave reviews. Such as the most used positive and negative words left in yelps reviews. Yelp can also benefit from this analysis as they can check the consistency between the text review and the numerical review left by customers. More importantly the businesses on yelp can benefit from this analysis as they will potentially have more information on what their best and worst aspects are of their business are.

In terms of academia currently the area is still being researched and performing opinion mining on review text has a lack of research specific no conclusive efficient method has been found with very high accuracy the greatest I found was 75%. This project can provide some useful data in this area in terms of the best classification method to use. Other researchers can then use this information in similar projects.

1.5 Primary Research Plan

Help: This is the plan as to how you will go about answering your detailed research question - It must include a primary research method (an extended literature review is not an acceptable primary method). Think and plan logically. Primary methods may include experiments, applications or software demonstrators, process models, surveys, analysis of generated data ...

Example: In the class register example above "to what extent does a mobile application reduce the number of errors made in class registers at Coventry University in comparison to current paper based registers" - the research plan may involve: 1) Collecting and analysing paper based registers in a given class on five occasions. 2) Identifying the error rate average on these occasions 3) Designing and implementing a mobile application that automatically records attendance in class. 4) Deploying the application in the class on five occasions. 5) Identifying the error rate average of the mobile application on these occasions. 6) Comparison of data and summary of findings.

1. Design and develop software to train and then analyse the yelp review data set
2. test the developed software
3. run experiments using the software on the yelp data set
4. analyse results and determine accuracy
5. make improvements to increase accuracy
6. repeat as necessary
7. generate conclusions, comparison of classification methods and future suggestions

This is the end of section one.

SECTION TWO: ABSTRACT AND LITERATURE REVIEW (1500 WORDS SUGGESTED)

2.1 Abstract

Help: An abstract is a short summary of a research project that enables other researchers to know if your report or research paper is relevant to them without reading the whole report. It is usually written retrospectively so that it can include findings and results. It is fully expected that you will rewrite your abstract when you come to write your final paper. For now, you should write an abstract of about 250 words that define the project described in section one. Before writing your abstract you MUST read some abstracts from conference or journal papers on *Google Scholar* or from *portal.acm.org* (to understand their style) and then provide your own abstract that outlines what your question is and what you 'did' to answer it.

yelp reviews can be valuable in that they provide information about business's and if you have a vast number of these reviews you can potentially understand a business's faults, well doings and overall rating. The problem is that there are lots of these reviews, 1.6 million of them to be exact. People do not have the time to look through every review out of the hundreds or thousands for a specific business. In this project I will aim to predict the rating of a review from the text alone, potentially summarise a bussiness's best and worst aspects and also potentially make correlations between tips and reviews. The machine learning bag of words model will be used to provide top frequent words that will train the system. Different classification models such as Support Vector Machine Classifier will be used to investigate accuracy. recommendations from other authors who did a similar task such as spell checking, normalisation when using SVM and only using adjectives in the bag of words will be used in hope to provide a method with improved accuracy. The model will be developed in python using the python natural language toolkit as a framework and can then be used for experiments between classification methods and we can make a recommendation on which classifier to use generally across all categories as some research only focused on specific categories.

2.2 Initial/Mini Literature Review (500 words – 750 words)

Help: A literature review is a select analysis of current existing research which is relevant to your topic, showing how it relates to your investigation. It explains and justifies how your investigation may help answer some of the questions or gaps in this area of research. A literature review is not a straightforward summary of everything you have read on the topic and it is not a chronological description of what was discovered in your field. Use your literature review to:

- compare and contrast different authors' views on an issue
- criticise aspects of methodology, note areas in which authors are in disagreement
- highlight exemplary studies
- highlight gaps in research
- show how your study relates to previous studies

Natural Language Processing

Abhimanyu Chopra, Abhinav Prashar, Chandresh Sain (2013)

For my analysis my software needs to be able to understand the review text which is written in natural language which means I need to look at natural language processing.

A natural language refers to a human language and is a language spoken and understood by people. Natural language processing allows human-computer interaction by the form of natural language. A computer understands machine language and a human understand natural language therefore natural language processing enables the computer to understand and generate natural language.(Chopra, Prashar and Sain 2013)

The steps of natural language processing are:

1. Morphological and Lexical Analysis (analysing the structure of words and dividing the text into paragraphs, words and sentences)
2. Syntactic Analysis (analysing sentences to interpret the grammatical structure)
3. Semantic Analysis (Theses structures are given meaning from dictionary or exact meaning from context)
4. Discourse Integration (the meaning of a sentence is dependent upon sentences that precede it. Such as, "Tom had an apple. Adam wants it.". To understand what "it" is we need to read the sentence that precedes it.)
5. Pragmatic Analysis (what is intended by the natural language text and how should it be interpreted? does the speaker mean what they literally say or do they imply another meaning? such as sarcasm)

(Chopra, Prashar and Sain 2013)

Chopra, Prashar and Sain (2013) describe NLP as both an interesting and difficult task with lots of baggage. We can understand that it is not practical to develop my own NLP software as this is not the aim of my project furthermore it is a tool enabling me to complete my research plan. However Steven Bird, Ewan Klein, and Edward Loper (2009) demonstrate the open-source python NLP toolkit which would allow me to perform natural language processing on the review text, manipulating and analysing the language data. The NLP toolkit Defines an infrastructure to build NLP applications and provides methods for operations that are common to NLP applications. Steven Bird, Ewan Klein, and Edward Loper (2009) also recommend the following python libraries for building NLP applications in python:

- NumPy (a scientific computing library, required for certain probability, tagging, clustering, and classification tasks.)
- Matplotlib (a 2D plotting library, providing data visualisation)

(Bird, Klein and Loper 2009)

The NLP Toolkit and the recommended libraries will allows me to perform NLP as listed by Chopra, Prashar and Sain 2013. open-source software is great for large problems such as this and is provided free. NLP toolkits exist that allow this large problem to be continually developed by many people and the solutions are providing for free and can be implemented to perform NLP.

Opinion Mining, Sentiment Analysis and machine learning

Opinion mining is a NLP and information extraction task that retrieves the feelings of the user expressed in comments by analysis. Sentiment analysis concerns classifying the documents to determine polarity, which is expressed in neutral, positive or negative. (Sharma, Nigam and Jain 2014) This is useful for my project to analyse the review text for rating and determine the positivity.

There are three levels that sentiment analysis can be performed.

1. Document level (determine polarity of the whole document)
2. Sentence level (determine polarity of sentences)
3. Aspect & Feature level (determine polarity of documents/sentences based on the aspects of those documents/sentences)

(Sharma, Nigam and Jain 2014)

After this a machine learning classification model can be used to classify the document as positive, negative, or more specifically classify it as 1,2,3,4 or 5.

Opinion mining is the core topic of this project, by which i mean the way that i extract the opinion of the review text for yelp is what the success of this project relies on. Looking I could find any journals that relate exactly to my project however there are very similar projects to mine and I found that this can be done in different ways, and is a current area that is still being researched.

Sharma, Nigam and Jain (2014) performed a similar task proposing a document based opinion mining system that classify the documents as positive, negative, and neutral. Doing this by exploring, analysing and organising reviews.

The proposed system uses a machine learning technique called unsupervised dictionary technique, which uses an automatic seed word selection method. utilising information about commonly occurring negations and adverbials. Other techniques were investigated by Sharma, Nigam and Jain however this technique was found to have the greatest accuracy. reviews of movies were collected and applied as an input to the system. The system classifies each document as positive, negative and neutral then presenting the total number of positive, negative and neutral number of documents separately in the output. The polarity of the given documents is determined on the basis of the majority of opinion words. (Sharma, Nigam and Jain 2014)

The phases of the system are:

- Data collection (collecting the data)
- POS Tagging (tags all the words of the documents to their appropriate part of speech tag, to determine the opinion words)
- Extracting opinion words and seed list preparation (machine learning to train)
- polarity detection and classification (determine positivity of text and classify accordingly)

The system was compared to human classification of the same reviews and achieved an accuracy of 63%.

Different methods have been used and achieved different results. In more detail all the methods were similar and had a general outline of:

1. data collection
2. process and prepare text, removing punctuation.
3. use machine learning to train and generate list of opinion words
4. compare text opinion words to the list of opinion words
5. determine results using a classification method

The key difference between the two methods is the classification method. Sharma, Nigam and Jain (2014) used a simple method where the positive words and negative words were compared and more positive words is deemed a positive review. Where as Callen Rain 2013 used a classification techniques Naive Bayes and decision list classifiers to tag a given review as positive or negative gave an improved accuracy of 75% on average across five different categories. (Rain 2013)

This could also be due to the fact that Callen Rain used bag of words to train the classifier.

Furthermore Callen Rain performed an addition step also and included spell checking, which has the aim of ensuring words are not missed when miss-spelled, which could have contributed. Another common problem that both authors agree on is negation, the problem of words that appear after a negating word now means the opposite such as "not like". This problem was solved and the author found that you should store three words after negating words to determine when negation should occur. (Rain 2013)

Predicting star rating of yelp reviews, and classification method.

So far It has become clearer on how to approach developing an opinion mining system using natural language processing however More specifically I looked at two further journals that both were performed on yelp reviews, predicting the star rating. I need to answer two questions I now have:

1. which machine learning algorithm I use to train the data
2. the classification method (more important)

Sasank & Ruchika 2015 classified reviews as positive or negative, along with also rating each review as cool as funny, cool or useful. Then they predicted the rating from the text. Mingming & Maryam simply only predicted rating from the text. Both used different classification methods.

Sasank & Ruchika 2015 tested two different classification models (Naive Bayes' Classifier and the Support Vector Machine Classifier) and a prediction method that was used, which was linear regression SVM Classification method was found to be the most successful. We see a common trend of SVM being better than Naive Bayes Classifier for all the features. This is because Naive Bayes falsely assumes that there is conditional independence. (Sasank & Ruchika 2015)This method is an improvement on Callen Rains proposal of Naïve Bayes however looking more closely the SVM only actually achieved about 58% accuracy normally. So maybe Callen Rains high accuracy was due to the spell checking, handling of negation or another factor Interestingly Fan & Khademi recommend as an improvement to their model to use POS tagging. (Fan & Khademi 2015) leading me to believe this could be an important aspect in relation to a high accuracy. This poses the question would SVM with some extra steps performed by Callen Rains improve on Callen Rains 75% average?

Li & Zhang also attempted to predict yelp reviews ratings and state that Naïve Bayes does not predict very well for both polarity and 5-star classification. This disagrees also to Callen Rains and agrees with Sansank & Ruchika and I am discovering that the classification methods results vary between projects and has shown not be useful for the yelp data set however applicable to other data sets.

Sasank & Ruchika 2015 also provide the most informative features noting that most were adjectives, they suggest that the bag of words should only be composed of adjectives and state it should improve performance this will be interesting to verify. Fan & Khademi agree with this suggesting to apply our models with better features. Fan & Khademi 2015 also explain that normalisation is helpful when applying SVM, so should be performed.

Fan & Khademi also make a suggestion for future work to try more feature extraction approach such as bigram, trigram or word chunks. Because this could effectively increase the information that can be obtained from one review text, and add more features to overcome the high bias problem. (Li & Zhang 2014)

2.3 Bibliography (key texts for your literature review)

Help: Please provide references, in correct Harvard style, for at least three key texts that have informed your literature review. If you are implementing an application, select texts which demonstrate how other researchers have tackled similar implementations? The references should be recent and sufficiently technical or academic. Your markers will be looking for you to identify technical reports, conference papers, journal papers, and recent text books. Avoid *Wikipedia* entries, newspaper reports that do not cite sources, and general or introductory texts.

Chopra, A., Prashar, A., Sain, C. (2013) 'Natural Language Processing?'. INTERNATIONAL JOURNAL OF TECHNOLOGY ENHANCEMENTS AND EMERGING ENGINEERING RESEARCH, VOL 1, ISSUE 4 [online] available from <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.407.6907&rep=rep1&type=pdf> [01 January 2016]

Bird, S., Klein, E., Loper, E. (2009) Natural Language Processing with Python. 1st edn. California: O'Reilly Media, Inc.

Sharma, R., Nigam, S., Jain, R. (2014) 'opinion mining of movie reviews at document level'.

International Journal on Information Theory (IJIT), Vol.3, No.3. [online] available from <http://arxiv.org/pdf/1408.3829.pdf> [03 January 2016]

Rain, C. (2013) Sentiment Analysis in Amazon Reviews Using Probabilistic Machine Learning [online] Final Project. Swarthmore College. available from <http://www.sccs.swarthmore.edu/users/15/crain1/files/NLP_Final_Project.pdf> [03 January 2016]

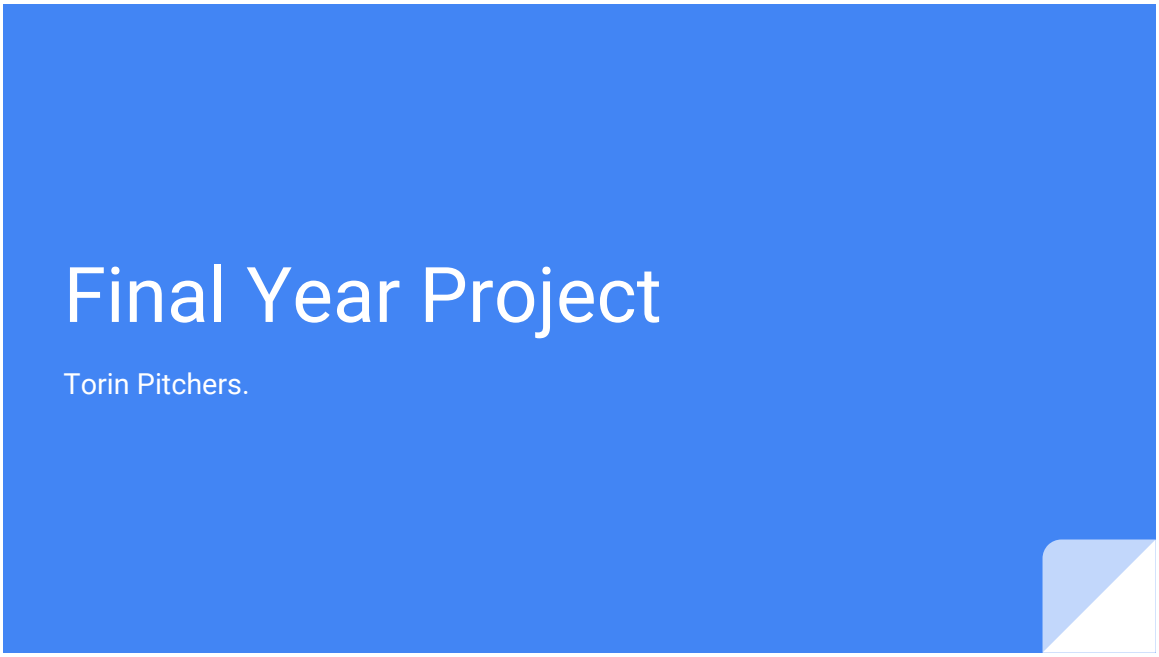
Channapragada, S. & Shivaswamy, R. (2015) Prediction of rating based on review text of Yelp reviews [online] Report. UC SanDiego. available from < http://cseweb.ucsd.edu/~jmcauley/cse255/reports/fa15/031.pdf > [15 January 2016]

Fan, M. & Khademi, M. (2014) Predicting a Business' Star in Yelp from Its Reviews' Text Alone [online] Report. Stanford University. available from < http://www.ics.uci.edu/~mkhademi/files/Publications/arXiv2013.pdf > [15 January 2016]

Li, C. & Zhang, J. (2014) Prediction of Yelp Review Star Rating using Sentiment Analysis. University of California. available from < http://cs229.stanford.edu/proj2014/Chen%20Li,%20Jin%20Zhang,%20Prediction%20of%20Yelp%20Review%20Star%20Rating%20using%20Sentiment%20Analysis.pdf > [22 January 2016]

This is the end of section two.

12.2 Presentation



Aim & Title of the project.

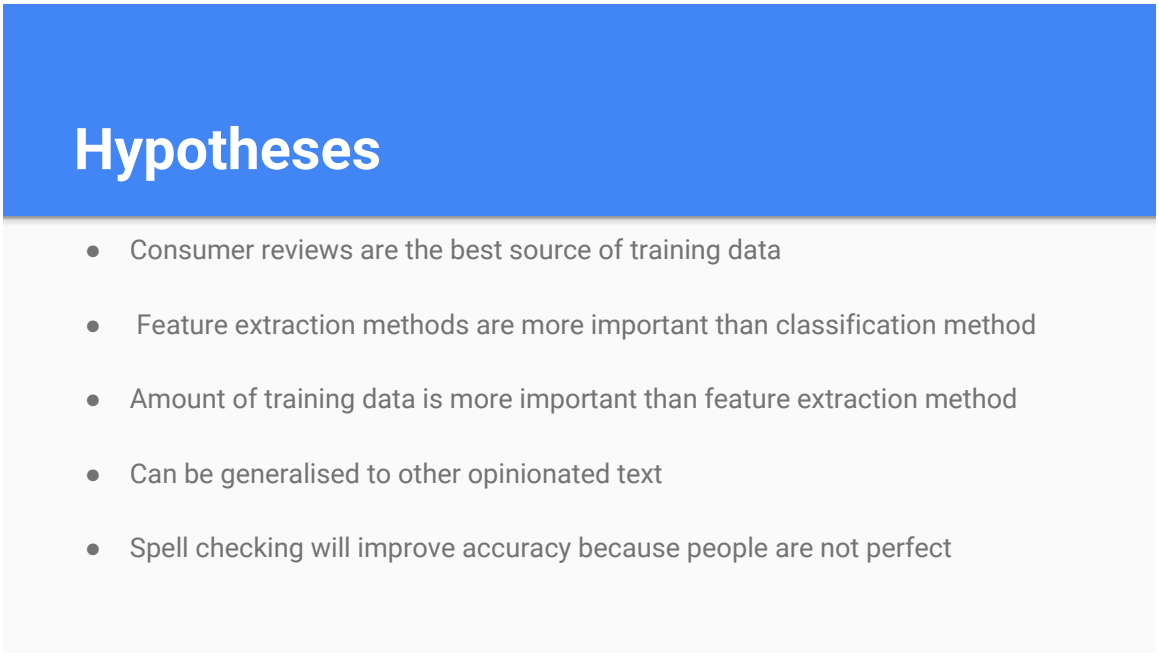
Title:
"Investigation of Sentiment Analysis Models for Natural Language Processing. Learning from consumer reviews, to classify polarity"

Aim:
To investigate how different aspects of a sentiment analysis model affects the accuracy and less importantly the runtime. For example the feature extraction or classification method. Also can this model be generalised?

Why:
From looking at the literature people seemed more concerned with the overall accuracy.

Feedback

Title is suitable for the project however could be better written as "Investigation of Sentiment Analysis Models for Natural Language processing: learning from consumer reviews to classify polarity".



Feedback

There are too many hypothesis that can be fully investigated in the time frame given. A more realistic approach would be to choose maybe two. Some can be ruled out because they are difficult to test for example the hypothesis "consumer reviews are the best source of training data" would require to test the model with other sources of data requiring all experiments to be repeated at least another two times: this will simply take too long. Focus on the main hypothesis that spell checking will improve accuracy as the experiments seem to agree with this. If you have time try to generalize to other opinionated text however this is not a priority.

Experimental Approach

Decided to tackle the problem in the form of experiments. This is because I need to determine the accuracy in relation to each aspect.

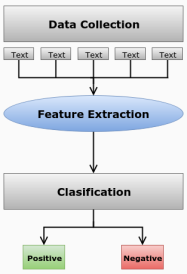
1. Develop base model
2. Change an attribute
3. Run the system and measure accuracy
4. **Repeat for all ideas or aspects**
5. Conclude 'perfect' model from results
6. Test model accuracy when classifying other opinionated texts after learning from consumer reviews.

Feedback

The approach is suitable and what would be expected for this type of project.

Methodology - Finding a basic model

From cross referencing the literature I found that the general model seemed to be:



Feedback

The general model is correct.

Methodology - Data collection & handling

- Consumer reviews are the best text to learn from because they are in the domain of big data (consistent velocity , complex variety, large volume).
- Used the yelp dataset of 2.2m reviews, because on average held more information than an amazon review.
- The normal big data problems: storage & runtime

Feedback

make sure the first bullet point is regarded as an opinion or for example "In my opinion ...", unless you have references to support this. Also be sure to add the references for bullet point 2.

Methodology - Data labelling

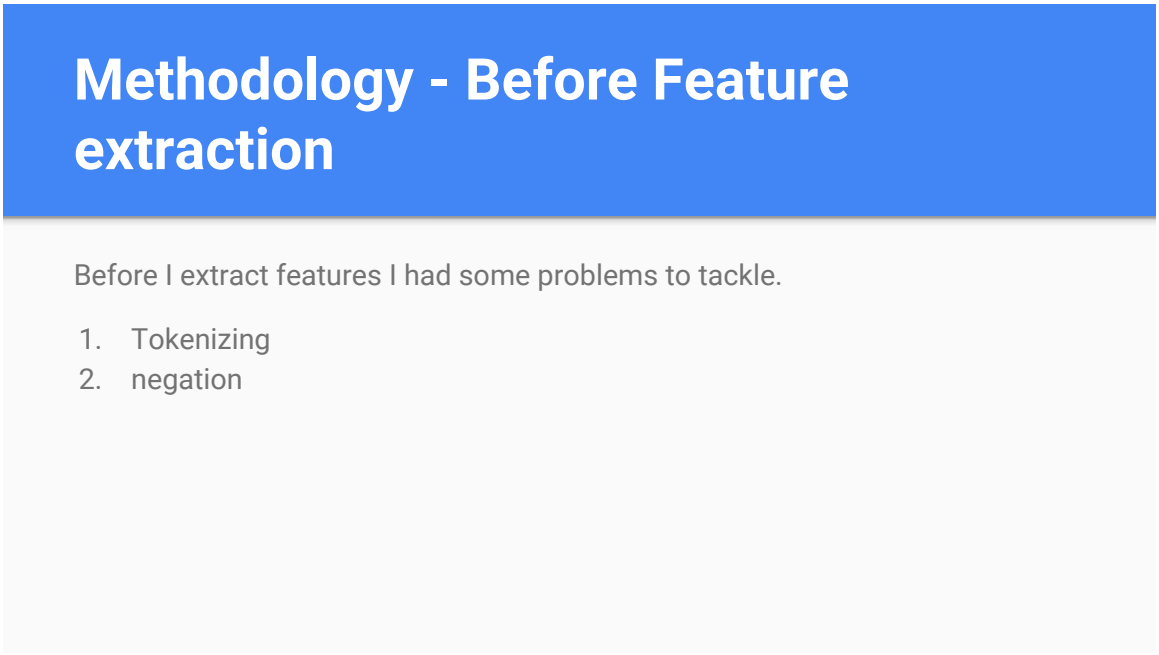
Problem:
As this project concerns a system that needs to classify text written reviews as positive or negative this means that the data needs to be in the same format. This means that **the reviews need to be labeled as positive or negative. However they are currently labeled 1-5.**

Methodology - Data labelling

Solution:
To solve this problem and to uphold validity of my results I decided to **remove all reviews that was ranked 3** and did not use them as training for the system. This meant that I would roughly lose 20% of my data assuming that the amounts are equally weighted across the ratings, however this meant I completely avoided the problem.

Feedback

Be sure to include this problem in your methodology and talk about how you solved it.



Feedback

Explain to the reader what these terms are and how you executed these, this could be added in the methodology section. Because the readers may not understand what these terms mean.

Methodology - Feature extraction

The first aspect i chose to investigate was feature extraction methods (commonly called feature generation). Which are methods used to extract ‘features’ of the text that are used to classify the text.

- Bag of words
- Bag of valuable words
- Attributed bag of valuable words
- N-gram model

Methodology - Classification method

The second aspect i chose to investigate was classification methods. A classifier is a machine learning algorithm that implements classification of input data to categories.

- Naive Bayes
- Support vector machine
- Decision tree

Feedback

In regards to your earlier opinion that feature extraction is more important than classification methods then it would be wiser to only use one classification method, most likely the most popular one. This is because to investigate classification methods as well you would need to implement each one for each feature extraction method which again will require too much time.

Methodology - Spell checking

The third aspect i chose to investigate was spell checking. Testing if spell checking all the text before hand improved the accuracy.

Results - Feature Extraction methods

Bag of Words

Measurement	Value
Accuracy(Percentage):	61.637293268%
Runtime(Seconds):	1516s

Table 3: A table of results for bag of words.

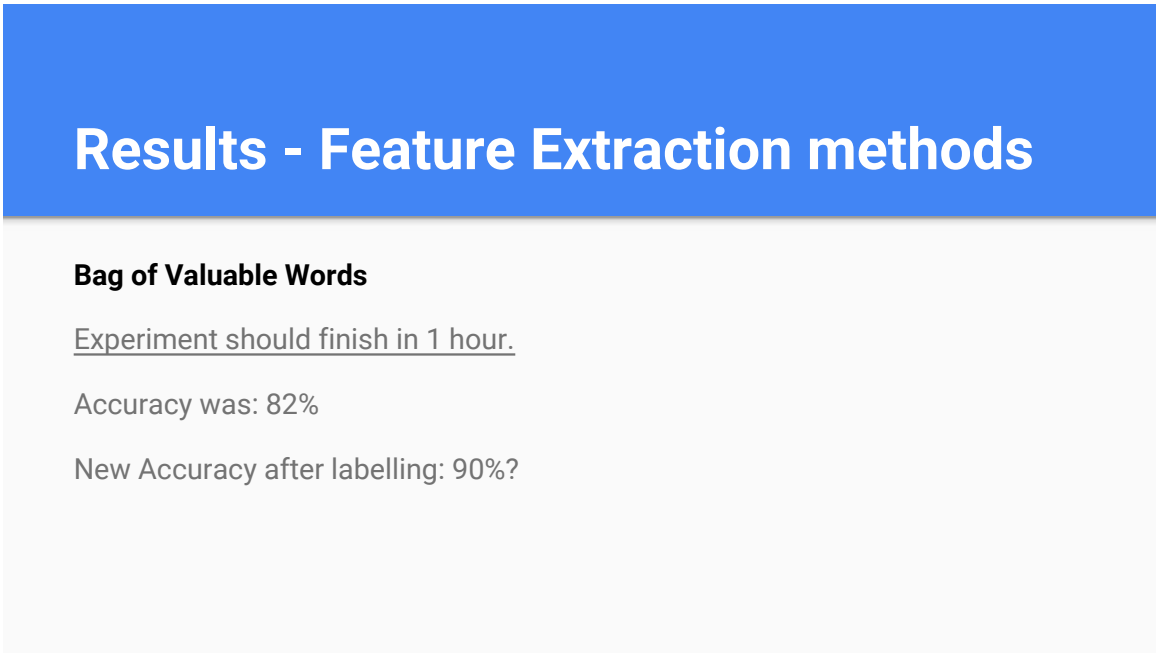
Feature	Probability Distribution
not returning	neg : pos = 102.6 : 1.0
not apology	neg : pos = 102.3 : 1.0
arrogantly	neg : pos = 81.1 : 1.0
ontrac	neg : pos = 78.7 : 1.0
not acknowledging	neg : pos = 78.2 : 1.0
fraudulently	neg : pos = 74.9 : 1.0
not refund	neg : pos = 73.9 : 1.0
not acted	neg : pos = 70.8 : 1.0
mismanagement	neg : pos = 70.8 : 1.0
not reimbursed	neg : pos = 68.8 : 1.0

Table 4: A table of top 10 most informative features for bag of words.

1. Negative features are more informative than positive features.
2. Negation is Important
3. Spelling mistakes are common
4. Anomaly's are present
5. better than guessing

Feedback

Interesting results seems to agree with the spell check hypothesis somewhat so continue with that. Be sure to add the 11th most informative feature which you said had a spelling mistake in it.



Feedback

Could look into precision and recall and include and discuss these measurements in the results.

Results - To be continued

Other Aspects are still to be investigated and experiments are running this very second.

- N-gram model
- Support vector machine
- Decision tree
- Spell check

Progress Report

Task	Completion
Write up (introduction, methodology, definitions, literature review, some of the results section)	60%
Experiments (bag of words, bag of valuable words, attributed bag of words)	50%
Generalization	0%

Project management method

Agile Methodology

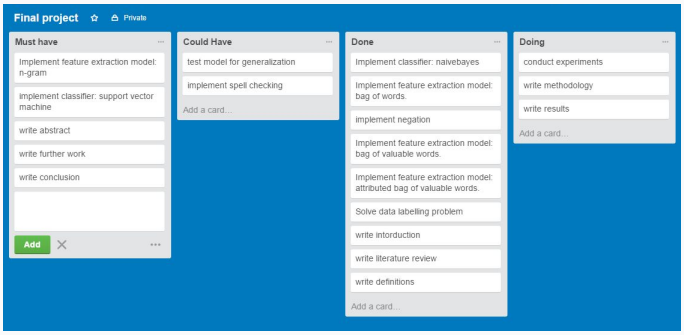
To manage,organise and plan my project I used an agile approach using SCRUM to define the workflow in the format of sprints. With agile processes such as user stories to plan the project and measure progress

Documented Meetings

After each meeting I document what we have discussed and what i did in response - will need to get you to sign and confirm these.

Project management method

Trello Board



12.3 Project Meeting Forms

304COM Project Meeting 1 – 06/11/2015

Discussion
We talked about the overall project such as the purpose and what the project entails. I Was also given information about topics of interest such as: • Natural Language Processing • Sentiment Analysis or otherwise known as opinion mining We also discussed how to approach the project and the research method to answer the questions proposed. I learned this concerns developing a natural language processing application that analyses the yelp data. I was informed of the NLP tool kit for python that would allow me to develop such an application. I was told to read on the information I was given and find some literature then decide on the research method and purpose of the project whilst getting some hands on experience with the nltk module.

Work
After some reading I started to look at journals and found that the question had been answered but in a limited sense. For example I found that it was possible to predict a reviews positivity however the best method to do so was unclear. The literature review created more specific questions such as which Machine Learning classifier produces the best accuracy and also how does error handling such as spell checking improve accuracy. All the models I saw from the literature has minor differences between one another and I discovered that there are lots of different aspects to the system such as: • Methods to clean the data(spell check, negation) • Method to extract features and which features do you extract?(tokenizer, pos taggers) • Which Machine learning classifier was used Wrote some basic code to open the yelp data file and extract the text and then tokenize the text.

Generated Conclusions after the work
• High accuracy does not define the successful of the project, the success is measured on identifying which aspects affect the accuracy and how they affect it. • Code needs to work with text files variables become to large and cause memory problems • Understand the aspects that are needed for a basic model

Supervisor Signature: P Smith

304COM Project Meeting 2 –04/01/2016

Discussion
We talked about my current understanding from reading some literature and Phil said he feels I understand enough to start the project and we agreed I should continue reading the literature in my spare time as there is lots to look at. Told me to start implementing the nltk module for the project and create a basic model.

Work
I was initially able to predict the reviews with a 35% accuracy, however by changing the format of my feature set to contain a dictionary of the words and also a rating of pos or neg where a rating of 3 – 5 is pos and 1- 3 is neg improved this slightly. Also From my reading of sentiment analysis I implemented the bag of words model which provided an improved accuracy of about 60%. Also found sentiword net that can help as it holds the positive or negative score for a word

Generated Conclusions after the work
• What feature extraction method should I use? And why? • Variable size becomes too large for training nltk needs to train from a file • Regex tokenizer gives user defined token structure • Sentiword net provides positive and negative scores for words

Supervisor Signature: P Smith

304COM Project Meeting 3 –20/02/2016

Discussion
We talked about my current implementation and the accuracy I have achieved which is 60% currently. Discussed the code and talked about efficiency problems. Phil introduced me to the idea of trigrams and bigrams as feature extraction methods and told me to read up on them. Phil also discussed the use of a corpus reader which can decrease the runtime.

Work
Investigated corpus reader, trigrams and bigrams and concluded that they would be suitable and interesting to implement into the project.

Generated Conclusions after the work
• Corpus reader would increase efficiency however would not allow me to manipulate the data as needed. • Trigrams and bigrams generate lots of features maybe should use sentiwordnet to identify important features only.

Supervisor Signature: P Smith

304COM Project Meeting 4 –10/03/2016

Discussion
Discussed the presentation ahead and told me important information such as the format of the presentation: informal 10 – 15 minute presentation to report the current progress. Told me to also include any current results and a small discussion or conclusion on them along with any problems I faced and how I solved them.

Work
Started the presentation and structured it in response to phils feedback by including some results for the bag of words feature extraction method that I had obtained. In response to the problems I included the key problems that were: 1. Labelling the data 2. Tokenizing 3. Negation

Generated Conclusions after the work
• Progress is a little bit behind schedule • Current results look promising

Supervisor Signature: P Smith

12.4 Project Diary

1. DATA COLLECTION & HANDELLING

COLLECTION:
Uses the yelp dataset, 2.1m reviews.

HANDELLING
- Due to large size was more logical to operate on one review at a time and then write to a file.

Then train the classifier from the formatted data. This allows the program to be stable and remove the risk of variable overload for machines with low ram.

- ERRORS, The JSON file contains errors at the moment I just log them and state how many occur and skip them.
Still investigating these... maybe format issues?

PROCESSING:
- RUN TIME, was directly affected by the data processing: cleaning the review and feature extraction.

LABELLING:
- need to re-label from 1-5 to pos & neg
1-2 = neg
4-5 = pos
3 = ignored

2. TOKENIZATION

splittings text into ~~tokens~~ words, these are called tokens and allow us to analyse the components of the text.

Problems:

- PUNCTUATION, tokens contain punctuation this creates errors and confusion as wordnet does not understand ~~there~~ "does'nt"

To resolve this problem I used RegexpTokenizer which tokenizes words from a regular expression. From this I used a regular expression that removed punctuation.

text-processing.com/demo/tokenize - shows comparison between tokenizers

sentence
~~word~~: "I don't like you.
tokens: [I], [don], [t], [like], [you]

Then search for character "t" if found ~~remove~~ skip it and/or not drop last letter of word that precedes it.

now have: [I], [do], [not], [like], [you]

- Then convert to lowercase

NLTK.tokenize

3. NEGATION

Need to identify negating words such as: "NOT". As discussed by CALLEN Rains 2016 & Sharma 2014.

CALLEN Rains approach seems most logical: searching for preceding "not" instances. She also actually found that searching 3 words before was most effective.

PROBLEMS TO CONSIDER:

- If a word is negated what should be done? - store as a bigram?
- Need to also ~~negate~~ invert Pos & neg scores from senti word net
- what if no words exist before the opinion words?

4. STOPWORDS

The tokens contain all the information or words of the text. some of these are not important and actualy create noise and distract the classifier from the useful ones.

stopwords are common words that hold no sentimental vale such as: to, join, go, because, ...

Using the stopwords module I removed these words.

NLTK. corpus. stopwords

5. SENTIWORDNET

NLTK.corpus.reader.sentiwordnet

REQUIRES:

1) NLTK.WORDNET
2) SentiWordNet ~~document~~ file containing all words
3) NLTK.POS_tag (need to POS tag for better results)

CAN BE USED FOR:

- provide POS & neg scores for words to identify opinion words.

PROBLEMS:

- some words not being found: solved by using wordnet manuals to find word then use its offset value to find the word in the SentiWordNet database.
- Takes a long time to run.

6. CLASSIFICATION

NLTK. Classify. NaiveBayes

REQUIRES:

- A pre labelled feature vector

PROBLEMS:

- Memory problems, due to the size of the feature vector which is having stored in a variable in RAM. This causes the program to crash.

Solved this by editing the Naive Bayes code to read and train from a file. This was simple and involved adding a simple loop through a file that contained the feature vector.

References

[1] Berman, J. (2013) *Principles of Big Data.* Massachusetts: Elsevier Inc, 20-21

[2] Bing, L. (2010) *Sentiment Analysis and Subjectivity* [online] available from <http://people.sabanciuniv.edu/berrin/proj102/1-BLiu-Sentiment%20Analysis%20and%20Subjectivity-NLPHandbook-2010.pdf >[28 February 2016]

[3] Bird, S., Klein, E., Loper, E. (2009) *Natural Language Processing with Python. 1st edn.* California: O'Reilly Media, Inc, 7

[4] Broder, A., Glassman, S., Manasse, M. & Zweig, G. (1997) *'Syntactic clustering of the web'* [online] available from <http://www.sciencedirect.com/science/article/pii/S0169755297000317>[16 March 2016]

[5] Channapragada, S. & Shivaswamy, R. (2015) *Prediction of rating based on review text of Yelp reviews* [online] Report. UC SanDiego. available from <http://cseweb.ucsd.edu/~jmcauley/cse255/reports/fa15/031.pdf>[15 January 2016]

[6] Chopra, A., Prashar, A., Sain, C. (2013) *Natural Language Processing?.* International Journal of Technology Enhancements and Emerging Engineering Research, Vol 1, Issue 4 [online] available from <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.407.6907&rep=rep1&type=pdf>[25 February 2016]

[7] Fan, M. & Khademi, M. (2014) *Predicting a Business' Star in Yelp from Its Reviews Text Alone* [online] Report. Stanford University. available from <http://www.ics.uci.edu/~mkhademi/files/Publications/arXiv2013.pdf>[15 January 2016]

[8] Hastie, T., Tibshirani, R. & Friedman, J. (2009) *'The Elements of Statistical Learning: Data Mining, Inference, and Prediction.'* [online] available from <http://statweb.stanford.edu/~tibs/ElemStatLearn/>[15 April 2016]

[9] Hung, K. & Qiu, H (2014) *Yelp Dataset Challenge 2014 Submission* [online] available from <http://kevin11h.github.io/

YelpDatasetChallengeDataScienceAndMachineLearningUCSD/>[28 February 2016]

[10] Jackson, P (2010). *Python Text Processing with NLTK 2.0 Cookbook.* Birmingham: Packt Publishing Ltd.

[11] Jackson, P.& Moulinier, I. (2007). *Natural Language Processing for Online Applications : Text Retrieval, Extraction and Categorization.* Amsterdam: John Benjamins Publishing Company.

[12] Laurent, A (2004). *Understanding Open Source and Free Software Licensing.* California: O'Reilly Media inc.

[13] Li, C. & Zhang, J. (2014) *Prediction of Yelp Review Star Rating using Sentiment Analysis* [online] Report. University of California. available from <http://cs229.stanford.edu/proj2014/Chen%20Li,%20Jin%20Zhang,%20Prediction%20of%20Yelp%20Review%20Star%20Rating%20using%20Sentiment%20Analysis.pdf>[15 January 2016]

[14] Li, F. (2015). *How we're teaching computers to understand pictures.* [online] available from <http://www.ted.com/talks/fei_fei_li_how_we_re_teaching_computers_to_understand_pictures>[22 April 2016]

[15] Manning, C., Raghavan, P. & Schütze, H. (2008) *Introduction to Information Retrieval* Cambridge: Cambridge University Press. [online] available from <http://www-nlp.stanford.edu/IR-book/>[15 March 2016]

[16] NLTK Project (2016) *nltk.tokenize package* [online] available from <http://www.nltk.org/api/nltk.tokenize.html>[08 March 2016]

[17] Oxford Dictionary (2016) *corpus - definition of corpus in English from the Oxford dictionary* [online] available from <http://www.oxforddictionaries.com/definition/english/corpus>[11 March 2016]

[18] Part of speech (2015) *Part of speech* [online] available from <http://partofspeech.org/>[15 March 2016]

[19] Princeton University (2015) *About WordNet* [online] available from <https://wordnet.princeton.edu/>[25 February 2016]

[20] Quantcast (2016) *Yelp.com Traffic and Demographic Statistics by Quantcast* [online] available from <https://www.quantcast.com/yelp.com#/trafficCard>[25 April 2016]

[21] Rain, C. (2013). *Sentiment Analysis in Amazon Reviews Using Probabilistic Machine Learning* [online] Final Project. Swarthmore College. available from <http://www.sccs.swarthmore.edu/users/15/crain1/files/NLP_Final_Project.pdf >[03 January 2016]

[22] SentiWordNet (2016) *SentiWordNet* [online] available from <http://sentiwordnet.isti.cnr.it/>[28 March 2016]

[23] Sharma, R., Nigam, S., Jain, R. (2014) *opinion mining of movie reviews at document level.*International Journal on Information Theory (IJIT), Vol.3, No.3. [online] available from <http://arxiv.org/pdf/1408.3829.pdf>[03 January 2016]

[24] Stanford NLP Group (2015) *The Stanford Natural Language Processing Group* [online] available from <http://nlp.stanford.edu/software/tagger.shtml>[11 March 2016]

[25] Text-Processing (2016) *Python NLTK Demos for Natural Language Text Processing and NLP* [online] available from <http://text-processing.com/demo/tokenize/>[20 March 2016]

[26] Woolf, M. (2014) *A Statistical Analysis of 1.2 Million Amazon Reviews* [online] available from <http://minimaxir.com/2014/06/reviewing-reviews/>[28 February 2016]

[27] Yelp (2016) *Yelp Dataset Challenge — Yelp* [online] available from <https://www.yelp.co.uk/dataset_challenge>[11 February 2016]