**Topic: Sentiment Analysis in Social Network Data**

1. **Abstract:**

Sentiment Analysis in Social Networks begins with an overview of the latest research trends in the field. It then discusses the sociological and psychological processes underling social network interactions. It analyses if the person’s comment was actually happy, angry or neutral also whether it is a positive or a negative comment. Sentiment analysis is widely used by email services to keep spam out of your inbox and by review websites to recommend new content like films or TV shows.

1. **Introduction:**

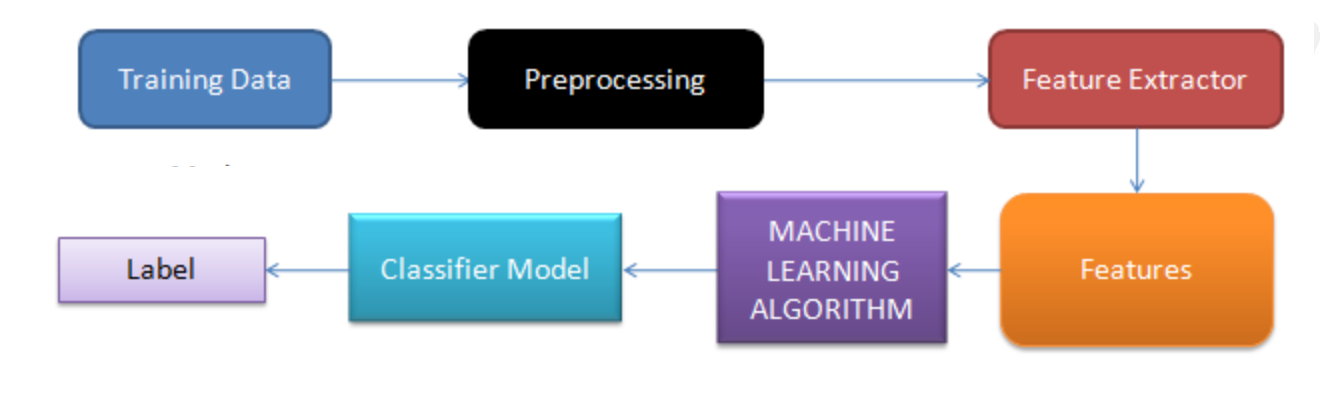
Sentiment Analysis in Social Networks begins with an overview of the latest research trends in the field. It then discusses the sociological and psychological processes underling social network interactions.

Sentiment Analysis of a chain of comments can allows us the analyze how people think about a particular topic or a comment. It can also allow us the understand a trend in a field. With technology’s increasing capabilities, sentiment analysis is becoming a more utilized tool for businesses. Social media monitoring tools use it to give their users insights about how the public feels in regard to their business, products, or topics of interest.

Suppose, From the comments on a particular post of a movie actor about their upcoming movie, we can actually be able to understand how excited people are about this project and how will it perform once released. Similarly, it can also be used to analyse the response of people towards any presidential candidate and understand what are their chances of winning. These are not the only application. It can be utilized in understanding people’s opinion in a lot of other ways.

1. **Approach:**

We use different feature sets and machine learning classifiers to determine the best combination for sentiment analysis of twitter. We also experiment with various pre-processing steps like - punctuations, emoticons, twitter specific terms and stemming. We investigated the following features - unigrams, bigrams, trigrams and negation detection. We finally train our classifier using various machine-learning algorithms - Naive Bayes and SVM ( Support Vector Machine) and compared their efficiency.

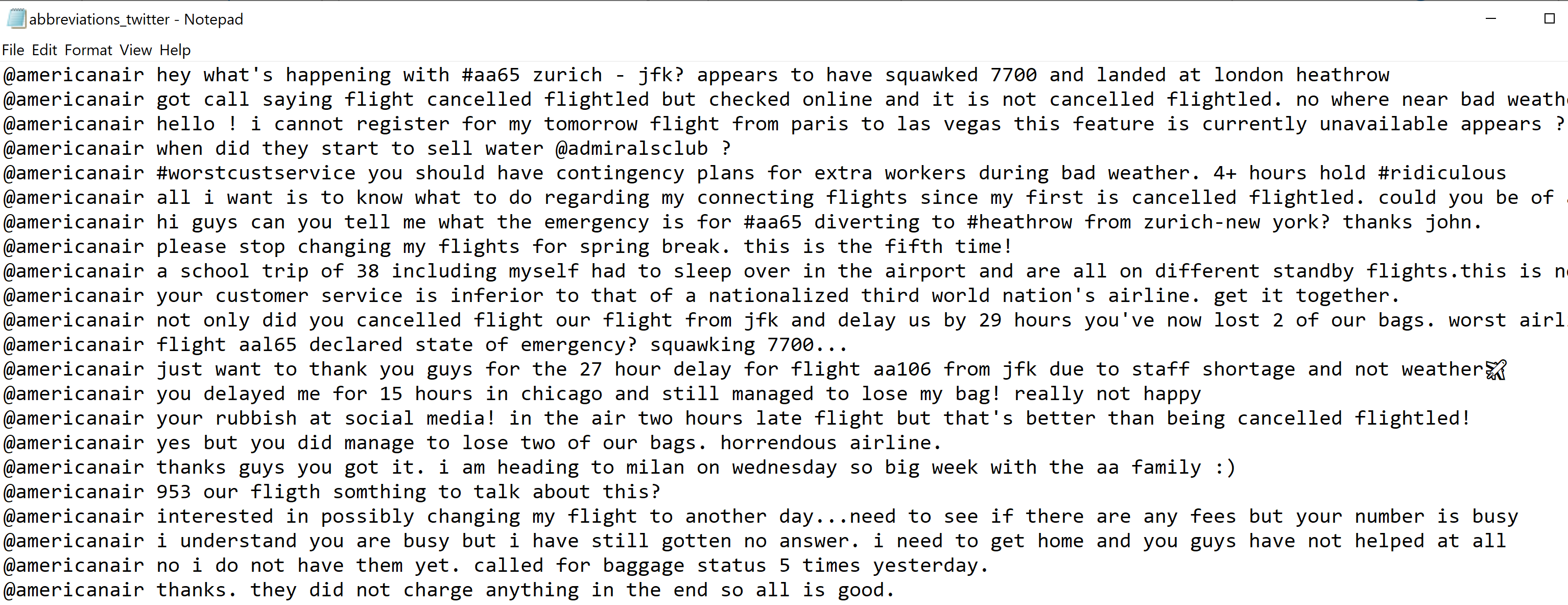


1. **Datasets used:**

The dataset used in the project was a tweet dataset related to airline companies ( i.e. tweets made by the airline companies or tweets in which an airline company was tagged). The dataset had nearly 15000 tweets and consisted of various airline companies in the dataset.

We approached the project by dividing this dataset into two parts namely training and testing data. 80% of the data was used as the training data and the remaining 20% was used as testing data.

**Snapshot of the dataset file:**



1. **Preprocessing:**

This is a very important step in training a machine learning model. User-generated content on the web is seldom present in a form usable for learning. It becomes important to normalize the text by applying a series of pre-processing steps. We have applied an extensive set of pre-processing steps to decrease the size of the feature set to make it suitable for learning algorithms.

1. **Hashtags**

A hashtag is a word or an un-spaced phrase prefixed with the hash symbol (#). These are used to both naming subjects and phrases that are currently in trending topics. For example, #iPad, #news

Regular Expression: #(\w+)

Replace Expression: HASH\_\1

Handles

Every Twitter user has a unique username. Any thing directed towards that user can be indicated be writing their username preceded by ‘@’. Thus, these are like proper nouns. For example, @Apple

Regular Expression: @(\w+)

Replace Expression: HNDL\_\1

1. **URLs**

Users often share hyperlinks in their tweets. Twitter shortens them using its in-house URL shortening service, like <http://t.co/FCWXoUd8> - such links also enables Twitter to alert users if the link leads out of its domain. From the point of view of text classification, a particular URL is not important. However, presence of a URL can be an important feature. Regular expression for detecting a URL is fairly complex because of different types of URLs that can be there, but because of Twitter’s shortening service, we can use a relatively simple regular expression.

Regular Expression: (http|https|ftp)://[a-zA-Z0-9\\./]+

Replace Expression: URL

1. **Emoticons**

Use of emoticons is very prevalent throughout the web, more so on micro- blogging sites. We identify the following emoticons and replace them with a single word. Table 4 lists the emoticons we are currently detecting. All other emoticons would be ignored.

1. **Punctuations**Although not all Punctuations are important from the point of view of classification but some of these, like question mark, exclamation mark can also provide information about the sentiments of the text. We replace every word boundary by a list of relevant punctuations present at that point. Table 5 lists the punctuations currently identified. We also remove any single quotes that might exist in the text.
2. **Lemmatization**Lemmatization is the process of normalizing a word rather than just finding its stem. In the process, a suffix may not only be removed, but may also be substituted with a different one. It may also involve first determining the part-of-speech for a word and then applying normalization rules. It might also involve dictionary look-up. For example, verb ‘saw’ would be lemmatized to ‘see’ and the noun ‘saw’ will remain ‘saw’. For our purpose of classifying text, stemming should suffice.
3. **Convert text to lowercase** – Allows us to deal with uniform case text.
4. **Remove numbers** – Numbers usually do not carry any importance in sentiment analysis
5. **Remove stop words** – Stop words are common words found in a language and they are mainly neutral.
6. **Stemming** – Transforms to root word. Stemming uses an algorithm that removes common word endings.
7. **Sparse terms** – We are often not interested in infrequent terms in our documents. Such “sparse” terms should be removed from the document term matrix.
8. **Feature Extraction**:

We have extracted features from each tweet. We have extracted features by creating the word embeddings. This was done by finding the frequency of each word in a tweet and creating a feature vector. Such feature vectors were created for all the tweets which were then combined into a single feature score vector. This feature score vector was then normalized using the StandardScalar library of nltk. Now this feature vector along with the true scores, was fed to the classifier model.

1. **Machine Learning Model:**

We used two different approaches to implement this project.

Approach 1: Implementation using Naïve Bayes approach using TextBlob library

Approach 2: Implementation using Support Vector Machine (SVM) using NLTK Library

As the features are extracted in the previous step, this step is all about how to utilize those features to get the answer that we are looking for. The model Trains a Classifier with train dataset. And then it processes the test dataset to come up with the result for the sentiment of the tweets.

Both the approaches analyze the sentiment using the concept of polarity score.

A basic task in sentiment analysis is classifying the polarity of a given text at the document, sentence, or feature/aspect level; whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral.

Consider a statement for example: **A really bad, horrible book.**

Now our model will work over this and return us: neg: 0.791, neu: 0.209, pos: 0.0

These are the polarity score for the above statement. It shows how biased our statement is towards a particular sentiment. Now who the winner is?

Negative

With polarity 0.791

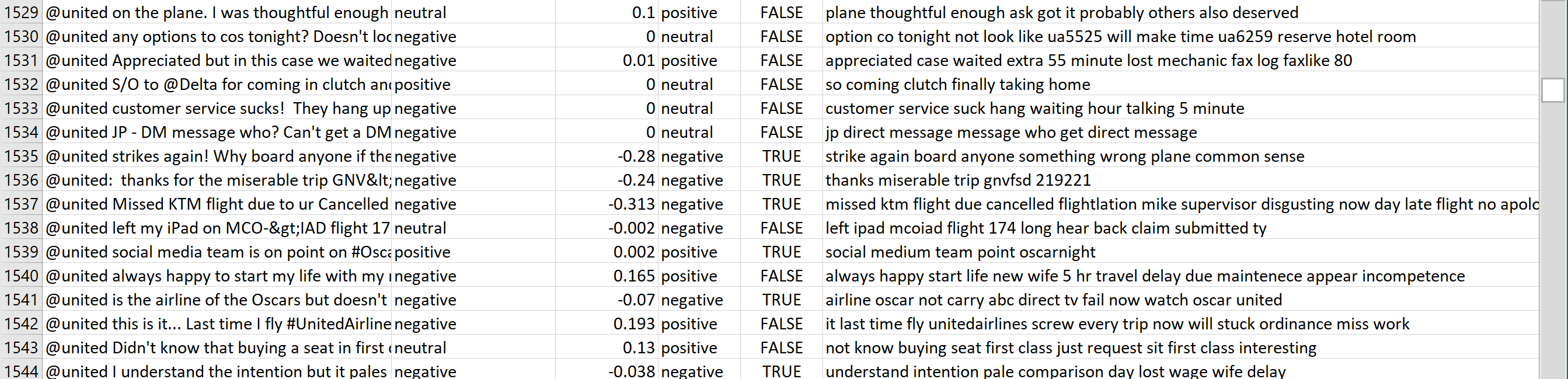
And hence resulting in “negative” result for that statement or tweet.

1. **Results:**

When trained the model efficiently, a result file is generated with a polarity value and the end result for the tweet.

For better understanding of the output, we have attached a snapshot of result of tweet with each answer namely, neutral, positive and negative.

**Result snapshot:**



Also, apart from this, another result for our project was to determine which of the two models worked better, the Naïve Bayes model using TextBlob library or the SVM model using NLTK library.

The result that we observed was that the SVM model performed better in terms of giving more accurate answers and improved efficiency.