**Introduction**

[Natural Language Processing (NLP)](https://monkeylearn.com/natural-language-processing/) is a branch of Artificial Intelligence (AI) that enables machines to understand human language. Its goal is to build systems that can make sense of the text and automatically perform tasks like translation, spell check, or [topic classification.](https://monkeylearn.com/topic-analysis/) Probably, the most popular examples of NLP in action are virtual assistants, like Google Assist, Siri, and Alexa. NLP understands the written text and transforms it into numbers, making it easy for machines to understand. Another well-known [application of NLP](https://monkeylearn.com/blog/natural-language-processing-applications/) is chatbots. They help support teams solve issues by understanding common language requests and responding automatically. Sentiment analysis in the financial domain is quickly becoming a prominent research topic as it provides a powerful method to predict market dynamics. In recent years, sentiment analysis has drawn a lot of research interest and has been applied in different domains. The financial domain is particularly relevant, as it has been shown that news and media can deeply affect market fluctuations. Indeed, positive news usually has a good impact on markets and generally tends to increase optimism. Thus, textual information processing has become a powerful tool to predict market dynamics. Sentiment analysis in the financial domain has been applied to a wide range of economic and financial fields, such as market prediction, analyzing consumers’ attitudes towards certain brands, or determining the financial blogger’s sentiment towards companies and their stock. Much of the financial sentiment analysis work has focused on data gathered from financial news. In this project, we have classified financial microblogs and news headlines as positive, neutral, and negative. A word embedding is a learned representation for text where words that have the same meaning have a similar representation. It is this approach to representing words and documents that may be considered one of the key breakthroughs of deep learning on challenging natural language processing problems. Various word embedding technique such as TF-IDF, Word2Vec, Fast text, and Embedding Layer was applied to the dataset. Transformed data were classified using machine learning models like Logistic Regression, Support vector classifier (SVC), Random forest classifier and K-Nearest Neighbour (KNN), and Deep Learning models like Long Short-Term Memory (LSTM) and Bi-directional LSTM. The report is organized into the following sections. Section 2 discusses the previous work done in this area. Section 3 is the methodology followed by a data description. The last section shows the result and discussion of our project.

**Literature Survey**

C. Zirn et al.[7] applied Markov logic to integrate polarity scores from different sentiment lexicons with information about relations between neighboring segments, and evaluate the approach on product reviews. The reviews were taken from 3 sections on ‘www.amazon.com’ namely “Cell Phones & Service”, “Gourmet Food” and “Kitchen & Housewares”. The reviews were classified based on the stars given by the user. Since lexicon-based methods do not rely on large amounts of training data and lexicons can easily be exchanged or added which makes the approach more flexible, polarity classification was used which looks up terms in a pre-compiled sentiment lexicon that lists terms and their polarities. SentiWordNet, Taboada and Grieve’s Turney Adjective List, and Unigram Lexicon were used to extract the polarity features. Three Markov Layer Networks (MLNs) and SVM were used to label discourse segments of a review as positive and negative. The MLN neighborhood formulation which incorporated structural information about neighboring segments was able to provide the best results. M. Atzeni et al.[1] used lexical, semantic as well as a combination of both. ‘n-grams’ were used for the lexical analysis. ‘BabelNet synsets’, which are a set of synonyms in different languages grouped by BabelNet, and ‘Semantic frames’ were used for the semantic analysis. A correlation metric between the word w and the sentiment scores of a set of messages M is computed and this metric is generalized and applied separately to unigrams, bigrams, 3-grams, BN synsets, and Semantic Frames, to select only the features that are the most significant in determining the polarity of the message. Finally, the system succeeded in achieving an accuracy level of more than 72% when the training model was boosted by semantics. C-C Chen et al.[2] used a total of 15 publicly available sentiment dictionaries and one sentiment dictionary constructed from the training set, containing sentiment scores in binary or real numbers, are used to compute the sentiment scores of text spans. They are used to capture the relationships between the interesting target and other stocks mentioned in a tweet and got the best accuracy of 55.43%. T Gaillat et al.[4] used data from microblog platforms such as StockTwits and Twitter to build a Sentiment Analysis (SA) system dedicated to financial microblogs in English. They extracted financial entities with relevant contexts and assigned scores on a continuous scale by adopting a deep learning method for the classification. They were able to get a 0.85 F1-Score on a two-class basis and a 0.62 cosine similarity score. F Tang et al.[6] proposes a joint aspect-based sentiment topic (JABST) model that jointly extracts multi-grained aspects and opinions through modeling aspects, opinions, sentiment polarities, and granularities simultaneously. Then, by means of supervised learning, a maximum entropy-based JABST model(MaxEnt–JABST) to improve accuracy and performance in extracting opinions and aspects is done. C.R Fink et al.[3] in their paper describing work they have done to annotate sentiment in blogs at the levels of sentences and sub sentences (clauses); to classify subjectivity at the level of sentences; and to identify the targets, or topics, of sentiment at the level of clauses. From the blog posts the sentence is chunked, and then checked whether it is subjective and contains sentiment then the fine-grained sentiment components are extracted after which it is sent for other processing and analysis. Their LS-SVM model was able to get a considerable accuracy with good recall and accuracy. S Taj et al.[5] presents a lexicon-based approach for sentiment analysis of news articles. The experiments were performed on a BBC news dataset, which expresses the applicability and validation of the adopted approach. The data collected was pre-processed and using WordNet the polarity of the sentiment words was calculated. Then the total sentiment score of the text is calculated and the result is output.

## Methodology

### Word Embedding

It is an approach for representing words and documents. Word Embedding or Word Vector is a numeric vector input that represents a word in a lower-dimensional space. It allows words with similar meanings to have a similar representation. They can also approximate meaning. A word vector with 50 values can represent 50 unique features. Word embedding can reduce dimensionality, to predict related words and we can also capture the semantic meaning of subwords.

**TF-IDF**

TF-IDF - stands for term frequency-inverse document frequency and it is a measure, used in the fields of information retrieval (IR) and machine learning, that can quantify the importance or relevance of string representations (words, phrases, lemmas, etc) in a document amongst a collection of documents (also known as a corpus).

TF-IDF can be broken down into two parts TF (term frequency) and IDF (inverse document frequency).

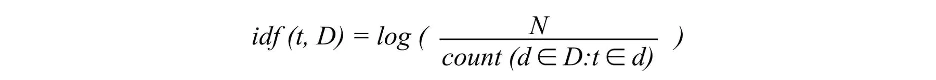
TF (term frequency)

Term frequency works by looking at the frequency of a particular term you are concerned with relative to the document. There are multiple measures, or ways, of defining frequency:

* A number of times the word appears in a document (raw count).
* Term frequency is adjusted for the length of the document (raw count of occurrences divided by the number of words in the document).
* Logarithmically scaled frequency (e.g. log(1 + raw count)).
* Boolean frequency (e.g. 1 if the term occurs, or 0 if the term does not occur, in the document).

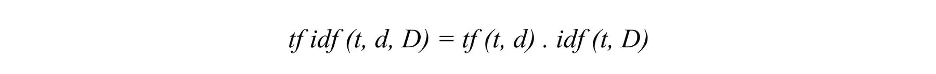
### IDF (inverse document frequency)

Inverse document frequency looks at how common (or uncommon) a word is amongst the corpus. IDF is calculated as follows where t is the term (word) we are looking to measure the commonness of and N is the number of documents (d) in the corpus (D). The denominator is simply the number of documents in which the term, t, appears in.



The reason we need IDF is to help correct words like “of”, “as”, “the”, etc. since they appear frequently in an English corpus. Thus by taking inverse document frequency, we can minimize the weighting of frequent terms while making infrequent terms have a higher impact.

To summarize the key intuition motivating TF-IDF is the importance of a term is inversely related to its frequency across documents.TF gives us information on how often a term appears in a document and IDF gives us information about the relative rarity of a term in the collection of documents. By multiplying these values together we can get our final TF-IDF value.

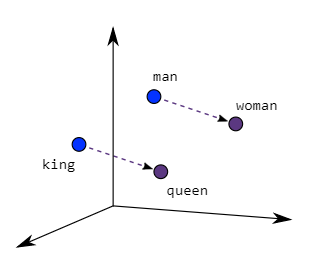


The higher the TF-IDF score the more important or relevant the term is; as a term gets less relevant, its TF-IDF score will approach 0.

**WORD2VEC**

Word2Vec creates vectors of the words that are distributed numerical representations of word features – these word features could comprise words that represent the context of the individual words present in our vocabulary. Word embeddings eventually help in establishing the association of a word with another similar meaning word through the created vectors.

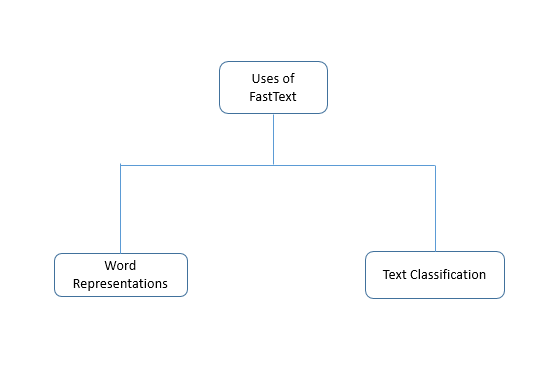
As seen in the image below where word embeddings are plotted, similar meaning words are closer in space, indicating their semantic similarity.



Two different model architectures that can be used by Word2Vec to create the word embeddings are the Continuous Bag of Words (CBOW) model & the Skip-Gram model.

**FAST TEXT**

FastText is a library created by the Facebook Research Team for efficient learning of word representations and sentence classification.



FastText differs in the sense that word vectors a.k.a word2vec treat every single word as the smallest unit whose vector representation is to be found but FastText assumes a word to be formed by an n-gram of character, for example, sunny is composed of [sun, sunn,sunny],[sunny,unny,nny], etc, where n could range from 1 to the length of the word. This new representation of word by fastText provides the following benefits over word2vec or glove.

It is helpful to find the vector representation for rare words. Since rare words could still be broken into character n-grams, they could share these n-grams with the common words. For example, for a model trained on a news dataset, the medical term eg: diseases can be the rare word.

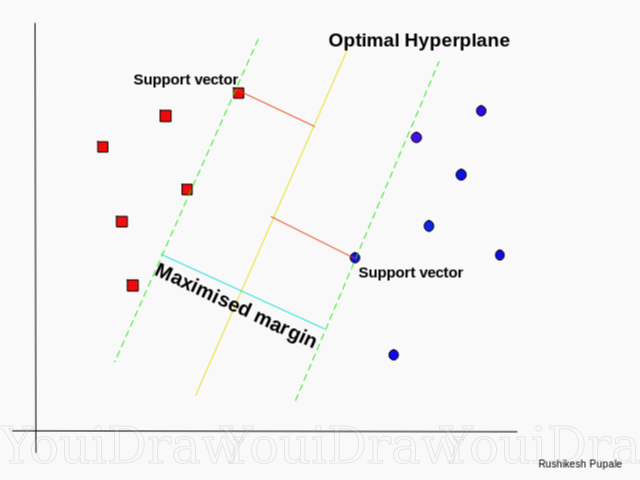
1. It can give the vector representations for the words not present in the dictionary (OOV words) since these can also be broken down into character n-grams. word2vec and glove both fail to provide any vector representations for words not in the dictionary.  
   For example, for a word like stupedofantabulouslyfantastic, which might never have been in any corpus, gensim might return any two of the following solutions – a) a zero vector or b) a random vector with low magnitude. But FastText can produce vectors better than random by breaking the above word into chunks and using the vectors for those chunks to create a final vector for the word. In this particular case, the final vector might be closer to the vectors of fantastic and fantabulous.
2. character n-grams embeddings tend to perform superior to word2vec and glove on smaller datasets.

**EMBEDDING LAYER**

**Machine Learning Models**

**Logistic Regression** - is a process of modeling the probability of a discrete outcome given an input variable. In the most common logistic regression model +a binary outcome; is something that can take two values such as true/false, yes/no, and so on. Multinomial logistic regression can model scenarios where there are more than two possible discrete outcomes. Logistic regression is a useful analysis method for classification problems, where you are trying to determine if a new sample fits best into a category.

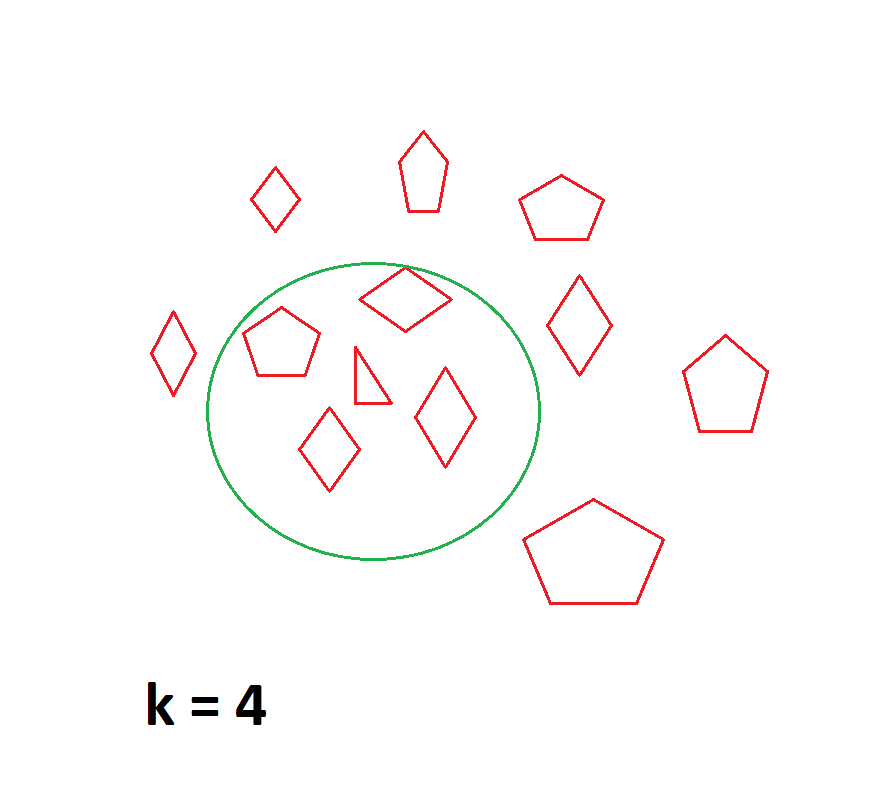
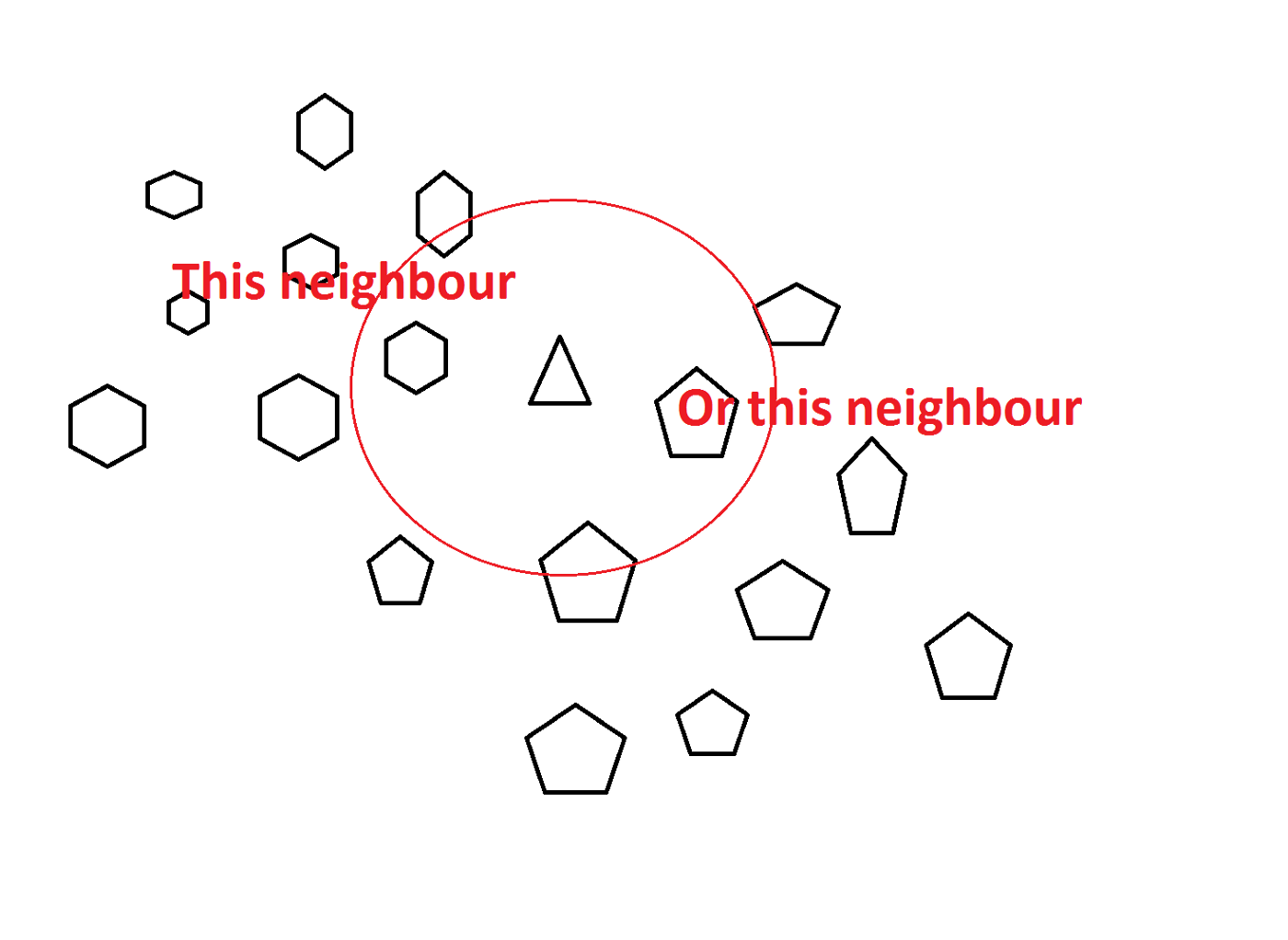
Support Vector Machines - or SVM is a linear model for classification and regression problems. It can solve linear and non-linear problems and work well for many practical problems. The idea of SVM is simple: The algorithm creates a line or a hyperplane which separates the data into classes. According to the SVM algorithm, we find the points closest to the line from both classes. These points are called support vectors. Now, we compute the distance between the line and the support vectors. This distance is called the margin. Our goal is to maximize the margin. The hyperplane for which the margin is maximum is the optimal hyperplane.



**Random forest** - is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables as in the case of regression and categorical variables as in the case of classification. It performs better results for classification problems.

**K Nearest Neighbors** - is one of the simplest forms of machine learning algorithms mostly used for classification. It classifies the data point on how its neighbor is classified.KNN classifies the new data points based on the similarity measure of the earlier stored data points. For example, if we have a dataset of tomatoes and bananas. KNN will store similar measures like shape and color. When a new object comes it will check its similarity with the color (red or yellow) and shape. K in KNN represents the number of the nearest neighbors we used to classify new data points.



**Deep Learning Model**

**LSTM**

The Long Short Term Memory architecture was motivated by an analysis of error flow in existing RNNs, which found that long time lags were inaccessible to existing architectures because backpropagated error either blows up or decays exponentially. An LSTM layer consists of a set of recurrently connected blocks, known as memory blocks. These blocks can be thought of as a differentiable version of the memory chips in a digital computer. Each one contains one or more recurrently connected memory cells and three multiplicative units - the input, output and forget gates - that provide continuous analogs of write, read and reset operations for the cells. More precisely, the input to the cells is multiplied by the activation of the input gate, the output to the net is multiplied by that of the output gate, and the previous cell values are multiplied by the forget gate. The net can only interact with the cells via the gates.

Recently, we have concentrated on applying LSTM to real-world sequence processing problems. In particular, we have studied isolated word recognition and continuous speech recognition with promising results.

## Dataset

Two datasets used for the project was taken from SemEval 2017 Task 5. The first dataset is a collection of 1694 financially relevant microblog messages focusing on stock market events. This messages are either exchanged via the StockTwits microblogging platform or via Twitter. This messages are either exchanged via the StockTwits microblogging platform or via Twitter. Usually, StockTwits messages contain references to company stock symbols, which are called cashtags. Each message is labeled with a real value denoting the sentiment towards the cashtag. The sentient value varies from -1 (very negative) to +1 (very positive) and zero shows the neutral sentiments. The other datasets is the collection of news headlines which consist of 1142 sentences from different sources, such as Yahoo Finance. This dataset is also labeled from -1 to +1 as did for the microblogs.

Dataset was further preprocessed by adding the positive, neutral and negative tags as the labels. For values ranging from -1 to -0.25 negative labels have been given, from -0.25 to 0.25 the text is considered as neutral and from 0.25 to +1 sentiment is considered as positive. Dataset was preprocessed by applying simple techniques like tokenization, removing stopwords, making the sentences lowercase, removing garbage and normalization.

Results and Discussions