Sentimental Data Analysis

Project report submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology
in
Computer Science and Engineering

by

Namish Narayan: 15UCS080
Yash Nag: 15UCS173

Under Guidance of
Mr. Kshitiz Verma

Department of Computer Science and Engineering
The LNM Institute of Information Technology, Jaipur

March 2018

The LNM Institute of Information Technology
Jaipur, India

CERTIFICATE

This is to certify that the project entitled Sentiment Data Analysis , submitted by Yash Nag (15UCS173) and Namish Narayan (15UCS080) in partial fulfillment of the requirement of degree in Bachelor of Technology (B. Tech), is a bonafide record of work carried out by them at the Department of Computer Science and Engineering, The LNM Institute of Information Technology, Jaipur, (Rajasthan) India, during the academic session 2018-2019 under my supervision and guidance and the same has not been submitted elsewhere for award of any other degree. In my/our opinion, this thesis is of standard required for the award of the degree of Bachelor of Technology (B. Tech).

_____ _____

Date Kshitiz Verma

Dedicated to our amazing Faculty and supportive Seniors and Peers

Acknowledgments

We would like to express our special thanks of gratitude to our mentor faculty, Kshitiz Verma who gave us a chance to perpetrate this project under his tutelage on this growing field of Machine Learning, we thank him for his constant support, clear answers and helping us improve this project along the way. It would also be unjust to not thank Andrew Ng, Harrison Kinsley and numerous other creators and tutors on their amazing video or text based lectures and in-depth illustrations of Machine Learning, without which, it would've been impossible to accomplish this project. We would also like to thank our seniors and peers at LNMIIT, who helped us correct every silly bug and made us easily understand those notorious computations with the silliest analogies. This project has inspired us to commit to more such projects in the future in this field of Science and we're more than determined to continue along this path, thanks to all of our supporters and well wishers.

Abstract

Sentimental Data Analysis is the process of determining the emotional tone behind a series of words, used to gain an understanding of the the attitudes, opinions and emotions expressed within an online mention. Using Neural Networks and other Machine Learning techniques, we can extract features, words and expressions that directly correlate to the respective writer's emotions. To determine the same, we extract data from tweets and other articles, and use TensorFlow to implement and TensorBoard to visualize the Neural Network propagation and extract the true weight of a phrase and it's correlation to a positive or negative sentiment.

Contents

Chapter Page

1 Introduction . . . 1
1.1 The Area of Work . . . 1
1.2 Problem Overview . . . 1
1.3 Solution . . . 1
1.4 Process . . . 2
1.5 Existing System . . . 3
1.5.1 Machine Learning Approach . . . 3
1.5.1.1 Support Vector Machines . . . 3
1.5.1.2 Naive Bayes . . . 3
1.5.1.3 Neural Networks . . . 4
1.5.2 Lexicon-based Approach . . . 4
1.5.2.1 Dictionary-based Approach . . . 4
1.5.2.2 Statistical Analysis . . . 4
1.5.2.3 Semantic Analysis . . . 5
1.6 Our Approach . . . 5

2 Neural Network . . . 7
2.1 Introduction . . . 7
2.2 About the Data . . . 7
2.3 Data Preprocessing . . . 8
2.3.1 Creating Lexicon . . . 8
2.3.2 Creating Vectors . . . 8
2.3.3 Adding Labels . . . 9
2.3.4 Creating Featuresets . . . 9
2.3.5 Exporting Final Data . . . 9
2.4 Building The Neural Network . . . 9
2.4.1 Neural Network Model . . . 10
2.4.2 Feed-forward and Back-propagation . . . 10
2.5 Testing and Results . . . 11
2.5.1 Lexicon Dictionary and Input . . . 11
2.5.2 Cost Reduction . . . 12
2.5.3 Varying Epochs . . . 12
2.5.4 Varying Hidden Layer Nodes . . . 13
2.5.5 Varying Number of Hidden Layers . . . 14
2.6 Conclusion . . . 15

Bibliography 16

Chapter *1*

Introduction

1.1 The Area of Work

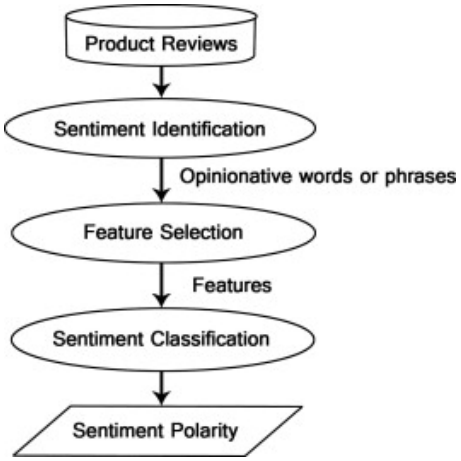
Our area of work will specifically be directed towards analyzing and extracting data. We'll be focusing on finding patterns are relation of phrases and words towards positive or negative sentiments of online comments, posts and tweets of the individual.

1.2 Problem Overview

The recent boom of online social media creates a giant void of making content be meaningful, to machines along with humans. Social Media giants like Facebook, Twitter, Instagram, Reddit etc. all need this data analyzed to filter out inappropriate content, categorization of posts along with numerous other applications. This gap between Hundreds of Terabytes of unrecognized data can be understood using Natural Language Processing, and for the specific problem we discussed, Sentimental Data Analysis hits the key goal.

1.3 Solution

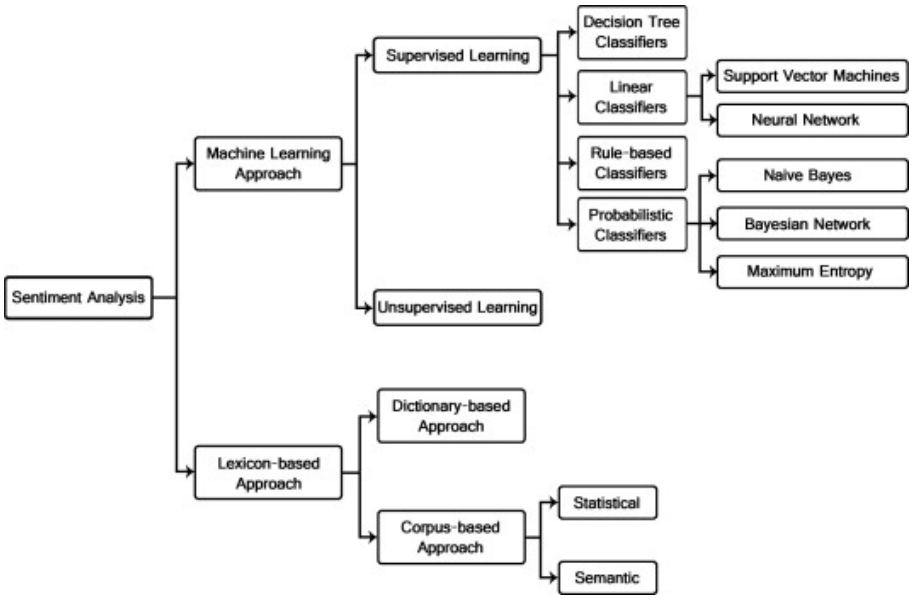
All of the data for the sentiment analysis that we will be using, will be compilation of open source data acquired from product reviews, tweets, blog posts or other social media outlets. Sentiment Analysis can be considered a classification process, There are three main classification levels in Sentiment Analysis: document-level, sentence-level, and aspect-level. We will focus our attention on Sentence-Level sentiment polarity classification.



Source: https://www.sciencedirect.com/science/article/pii/S2090447914000550

1.4 Process

The process for sentiment analysis varies as the data varies. Only Lexicon based Approach doesn't define a well fitted line for positive and negative sentiment, but helps significantly in associating words with sentiments. To get a well fitted line, we'll focus our approach towards Linear Classifiers. From the empirical comparison between SVMs and Artificial neural networks presented by Moraes and Valiati [13], indicated that ANN produced superior results to SVM except for some unbalanced data contexts. Therefore, we pursue our study in Neural Networks by using labelled data for our Supervised Learning. Neural Network's classification can always be improved by feeding more data.



Source: https://www.sciencedirect.com/science/article/pii/S2090447914000550

1.5 Existing System

Following sub-sections give brief overview of current systems to tackle Sentiment Analysis. Some of the following summaries are extracted from Sciencedirect's excellent overview of Sentiment Analysis and it's approaches.

1.5.1 Machine Learning Approach

Machine learning approach relies on Machine Learning algorithms to solve the Sentiment Analysis as a regular text classification problem that makes use of syntactic and/or linguistic features.

1.5.1.1 Support Vector Machines

Chen and Tseng [2] have used two multiclass SVM-based approaches: One-versus-All SVM and Single-Machine Multiclass SVM to categorize reviews. They proposed a method for evaluating the quality of information in product reviews considering it as a classification problem. They worked on digital cameras and MP3 reviews.

SVMs were used by Li and Li [9] as a sentiment polarity classifier. Unlike the binary classification problem, they argued that opinion subjectivity and expresser's credibility should also be taken into consideration. They proposed a framework that provides a compact numeric summarization of opinions on micro-blogs platforms. They identified and extracted the topics mentioned in the opinions associated with the queries of users, and then classified the opinions using SVM. They worked on twitter posts for their experiment.

1.5.1.2 Naive Bayes

The Nave Bayes classifier is the simplest and most commonly used classifier. Nave Bayes classification model computes the posterior probability of a class, based on the distribution of the words in the document. The model works with the BOWs feature extraction which ignores the position of the word in the document. It uses Bayes Theorem to predict the probability that a given feature set belongs to a particular label.

$$P(Label \mid Features) = \frac{P(Features \mid Label)\, P(Label)}{P(Features)}$$

A better solution was proposed by Kang and Yoo [7] to solve the problem of the tendency for the positive classification accuracy to appear up to approximately 10% higher than the negative classification accuracy. This creates a problem of decreasing the average accuracy when the accuracies of the two classes are expressed as an average value. They showed that using this algorithm with restaurant reviews narrowed the gap between the positive accuracy and the negative accuracy compared to Naive Bayes and SVM.

1.5.1.3 Neural Networks

There is an empirical comparison between Support Vector Machines and Artificial Neural Networks presented by Moraes and Valiati [13] regarding document-level sentiment analysis. Their experiments indicated that ANN produced superior results to SVM except for some unbalanced data contexts. They have tested three benchmark data sets on Movie, GPS, Camera and Books Reviews from amazon.com. They proved that the experiments on movie reviews ANN outperformed SVM by a statistically significant difference.

Van de Camp and Van den Bosch's [16] case study was based on historical biographical information describing people in a particular domain, region and time frame. They showed that their classifiers were able to label these relations above a majority class baseline score. They found that a training set containing relations, surrounding multiple persons, produces more desirable results than a set that focuses on one specific entity.

1.5.2 Lexicon-based Approach

Classification using existing positive and negatively defined opinion words.

1.5.2.1 Dictionary-based Approach

[5] and [8] presented the main strategy of the dictionary-based approach. A small set of opinion words is collected manually with known orientations. Then, this set is grown by searching in the well known corpora WordNet [11] or thesaurus [12] for their synonyms and antonyms. The newly found words are added to the seed list then the next iteration starts. The iterative process stops when no new words are found. After the process is completed, manual inspection can be carried out to remove or correct errors.

Qiu and He [14] used dictionary-based approach to identify sentiment sentences in contextual advertising. They proposed an advertising strategy to improve ad relevance and user experience. They used syntactic parsing and sentiment dictionary and proposed a rule based approach to tackle topic word extraction and consumers attitude identification in advertising keyword extraction. They worked on web forums from automotvieforums.com.

1.5.2.2 Statistical Analysis

Finding co-occurrence patterns or seed opinion words can be done using statistical techniques. This could be done by deriving posterior polarities using the co-occurrence of adjectives in a corpus, as proposed by Fahrni and Klenner [4]. The polarity of a word can be identified by studying the occurrence frequency of the word in a large annotated corpus of texts [15]. If the word occurs more frequently among positive texts, then its polarity is positive. If it occurs more frequently among negative texts, then its polarity is negative. If it has equal frequencies, then it is a neutral word.

Hu and Bose [6] expected that the writing style of the reviews would be random due to the various backgrounds of the customers, if the reviews were written actually by customers. They worked on Book reviews from amazon.com and discovered that around 10.3% of the products are subject to online reviews manipulation.

Latent Semantic Analysis (LSA) is a statistical approach which is used to analyze the relationships between a set of documents and the terms mentioned in these documents in order to produce a set of meaningful patterns related to the documents and terms [3]. Cao and Duan [1] have used LSA to find the semantic characteristics from review texts to examine the impact of the various features. The objective of their work is to understand why some reviews receive many helpfulness votes, while others receive few or no votes at all. They investigated the factors that determine the number of helpfulness votes which a particular review receives (include both yes and no votes). They worked on software programs users feedback from CNET Download.com. They showed that the semantic characteristics are more influential than other characteristics in affecting how many helpfulness vote reviews receive.

1.5.2.3 Semantic Analysis

The Semantic approach gives sentiment values directly and relies on different principles for computing the similarity between words. This principle gives similar sentiment values to semantically close words.

The Semantic approach is used in many applications to build a lexicon model for the description of verbs, nouns and adjectives to be used in SA as the work presented by Maks and Vossen [10]. Their model described the detailed subjectivity relations among the actors in a sentence expressing separate attitudes for each actor. These subjectivity relations are labeled with information concerning both the identity of the attitude holder and the orientation (positive vs. negative) of the attitude. It provided means for the identification of the attitude holder, the polarity of the attitude and also the description of the emotions and sentiments of the different actors involved in the text. They used Dutch WordNet in their work. Their results showed that the speakers subjectivity and sometimes the actors subjectivity can be reliably identified.

1.6 Our Approach

Support Vector Machines and Neural Networks can be used also for the classification of personal relationships in biographical texts as presented by van de Camp and van den Bosch [16]. They marked relations between two persons as positive, neutral, or unknown. They showed that their classifiers were able to label these relations above a majority class baseline score. They found that a training set containing relations, surrounding multiple persons, produces more desirable results than a set that focuses on one specific entity. They proved that SVM and one layer NN (1-NN) algorithm achieve the highest scores. In order to get more relevant intelligence and better efficiency we strive to achieve, we pursued towards more of a generalized solution than one that will only work on a specific dataset.

Considering all the available approaches, we chose to go with the most sustainable, expandable and efficient way. Hence, to find the best combination, we selected the Lexicon Based approach, along with Neural Networks, coherent with the dataset.

Chapter *2*

Neural Network

2.1 Introduction

After going through a bunch of different platforms to work on, we concluded that TensorFlow, using Python, was the most efficient and time-effective way to implement our ideas. This is due to their excellent documentation, numerous examples and a giant welcoming community of enthusiastic developers.

2.2 About the Data

To start off our attempt at implementation, we used a small openly available positive and negative sentiment dataset with 5331 lines of labelled data, for each positive and negative sentiment. We used a basic, Feed-forward Back-propagating Neural Network and focused on it being expandable, flexible and easy to understand and implement. Full Code Link here.

”too much power , not enough puff”
”narc is all menace and atmosphere”
”an almost unbearably morbid love story”
”feels strangely hollow at its emotional core”
”enigma is well-made , but it’s just too dry and too placid”
”it’s tough to be startled when you’re almost dozing”

Extracts (per line) for Negative Sentiment Dataset

”another best of the year selection”
”an entertaining , if ultimately minor , thriller”
”an almost unbearably morbid love story”
”a lovably old-school hollywood confection”
”simultaneously heartbreakingly beautiful and exquisitely sad”
”a solidly entertaining little film”

Extracts (per line) for Positive Sentiment Dataset

The entire data consists of 2 files, namely Positive.txt and Negative.txt, both of which can be extracted from Positive.txt and Negative.txt links.

2.3 Data Preprocessing

The data, as one can see, is in natural language and is completely unstructured. Therefore, to apply any implementation, first we need to process it in an organized format to make it structured. To restructure this data, we used various Library like NLTK, NumPy, Pandas, Pickle etc. The preprocessing follows the given structure

- Creating Lexicon
- Creating Vectors
- Adding Labels
- Creating Featuresets
- Exporting Final Data

2.3.1 Creating Lexicon

First we created Lexicon, which is basically a dictionary for all the words that appear in all the lines in a file. Tokenizer extracts words from each line of the file, removing unnecessary words like ”the”, ”and” etc. which do not contribute to sentiments or quality of the line. We also used Lemmatizer to eliminate words which infer the same (ex. ”cat” and ”cats” can be treated as one). Creating Lexicon gives us the full discovery of all the words occurring in both the files

2.3.2 Creating Vectors

After creating the Lexicon, we form a NumPy array (of Zeros initially) for each and every line of both the files. This array represents the words that are present in that particular line with respect to the lexicon. This means, that the array is a vector, which has 1 (or > 1) values due to words occurring in that line, at the same index as of the lexicon's.

```
Let the Lexicon be:
    lex = {"Cat", "Ball", "Mouse", "Pizza"}
Also, let a line be:
    line = "Cat ate the mouse"
Therefore, it's vector will be:
    vec = [1, 0, 1, 0]

Note: Actual Size of Lexicon = 423
```

2.3.3 Adding Labels

Vectors, after being created for each line for both the files, are then combined with their labelled sentiment. ie. "[1,0]" for Positive Sentiment, and "[0,1]" for Negative Sentiment. Therefore, the vector now transforms into a 2 dimentional aspect, one containing the vector of words in the line, and the other containing the label it carries.

```
Let a vector be:
    vec = [1, 0, 1, 0]
Now let's say that it was in the positive.txt file,
therefore it will be labbeled as [1,0]
    final_vec = [[1,0,1,0], [1,0]]
```

2.3.4 Creating Featuresets

All the final vectors formed are combined in one giant object, named 'featureset'. Note that this object contains all the lines in both the files in vector forms along with their labels. After creating this object, we randomize entry. This is to ensure in the future that when the Neural Network will be training on this data, It won't be trained to the extreme for one sentiment (More on this in later sections). After Randomizing, we split the object in 4 Objects, diving them to form training and testing data. The code is self explanatory.

```
    train_x = list(features[:,0][:-testing_size])
    train_y = list(features[:,1][:-testing_size])
    test_x = list(features[:,0][-testing_size:])
    test_y = list(features[:,1][-testing_size:])
```

2.3.5 Exporting Final Data

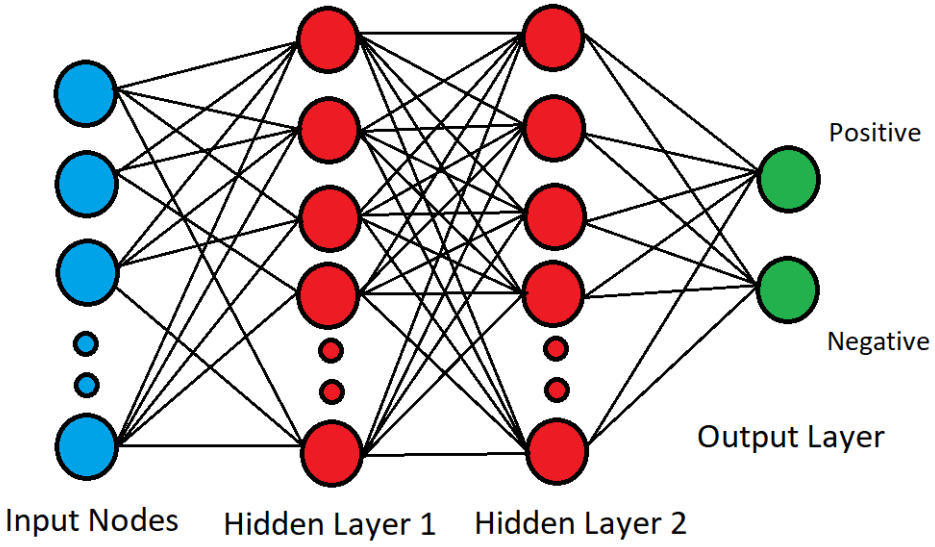
After the division of the featureset into the 4 aforementioned objects, the objects are are encapsulated using the 'Pickle' Library, and are exported. Pickle data are converted from python objects to byte stream. It helps serialize all the data, and makes storage and access of python objects easier.

2.4 Building The Neural Network

We used Neural Networks due to it's expandability and flexibility regarding fitting data points and it's accuracy over larger datasets. The standard Neural Network Model we build consisted of 1 Input, 2 Hidden, 1 output layer. Number of nodes in Input layer is decided by the length of 'train-x' features from the pickle. Output Layer consists of a 2 nodes, 1 for each class. Since it's a classification problem (whether a line exhibits positive or negative sentiment) the output node has 2 classes, one for each sentiment.

2.4.1 Neural Network Model

In this fully connected Neural Network, each node is connected to all the nodes in the layer before and after it's own layer. The diagram makes it easier to visualize. Each connection between 2 nodes has a weight and a bias value associated with it, which is initialized to random numbers. We also used Rectified Linear (or ReLu) as an activation function for each connection.



2.4.2 Feed-forward and Back-propagation

After the Neural Network Model is built, we set it up to train it with our data. We used Softmax Cross Entropy with Logits, to reduce cost and AdamOptimizer instead of classical stochastic gradient which is a combination of Adaptive Gradient Algorithm(AdaGrad) and Root Mean Square Propagation (RMSProp), and a standard in TensorFlow. We Then trained the Model in Batches of 100 lines per batch. Do Note that TensorFlow is a library which handles back propogation all by itself with the provided model in a given session. Thus, the labour intensive part was handled by the library itself.

```
#This is the session's code where Neural Network is Trained
with tf.Session() as sess:
        sess.run(tf.global_variables_initializer())
        for epoch in range(hm_epochs):
            epoch_loss = 0
            i=0
            while i < len(train_x):
                start = i
                end = i + batch_size
                batch_x = np.array(train_x[start:end])
```

```
                batch_y = np.array(train_y[start:end])

                #Sess.run handles backpropagation and reduces the cost of the Mod
                _, c = sess.run([optimizer, cost], feed_dict={x: batch_x, y: batc
                epoch_loss += c
                i += batch_size
            print('Epoch', epoch+1, 'completed out of',hm_epochs,'loss:',epoch_lo
        correct = tf.equal(tf.argmax(prediction, 1), tf.argmax(y, 1))
        accuracy = tf.reduce_mean(tf.cast(correct, 'float'))

#Epoch is the number of times the Model is being trained by the same data
```

2.5 Testing and Results

Training and Testing gave us alot of insights other than accuracy. Following subsections shows extracted and plotted data of cost and also shows the result of our experimentation on the Neural Network Model by modifying multiple parameters like the Number of Epochs, Number of Hidden Layers, Nodes in the Hidden Layers etc. All of the results can easily be visualized with the Graphs below.

2.5.1 Lexicon Dictionary and Input

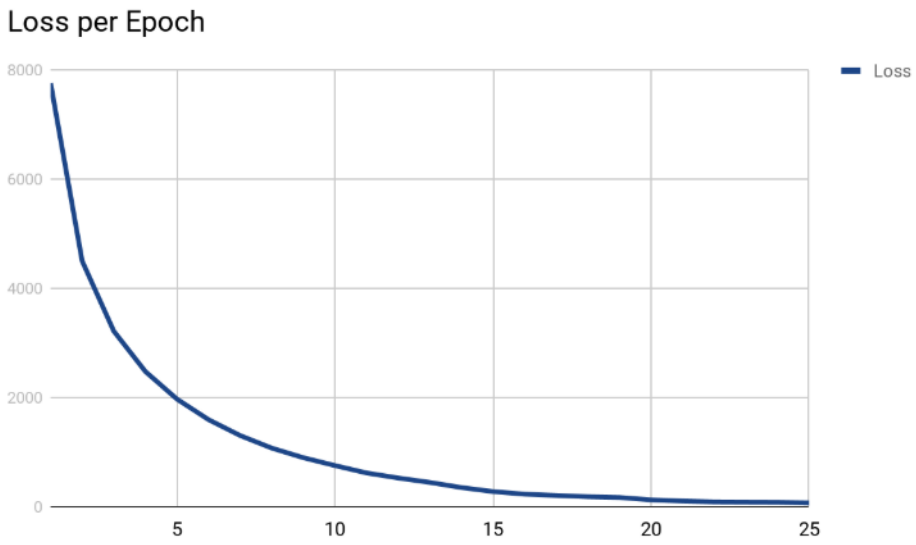
After the preprocessing of the data, we get the final lexicon as well as featuresets and labels for both, testing and training data.

```
Length of Lexicon:  423
Lexicon:
 ['american', 'since', 'been', 'when', 'screen', 'both', 'just',
 'character', 'part', 'charm', 'obvious', 'too', 'sense' ...
 'boring', 'dull']

#Below are examples of one of each, of the following-
train_x (Training Data):
[1, 1, 2, 1, 1 ...  0, 0]

train_y (Training Label):
[1, 0]

test_x (Testing Data):
[0, 0, 0, 2, 0, 0 ... 1, 0]
```

```
test_y (Testing Label):
[0,1]
```

2.5.2 Cost Reduction

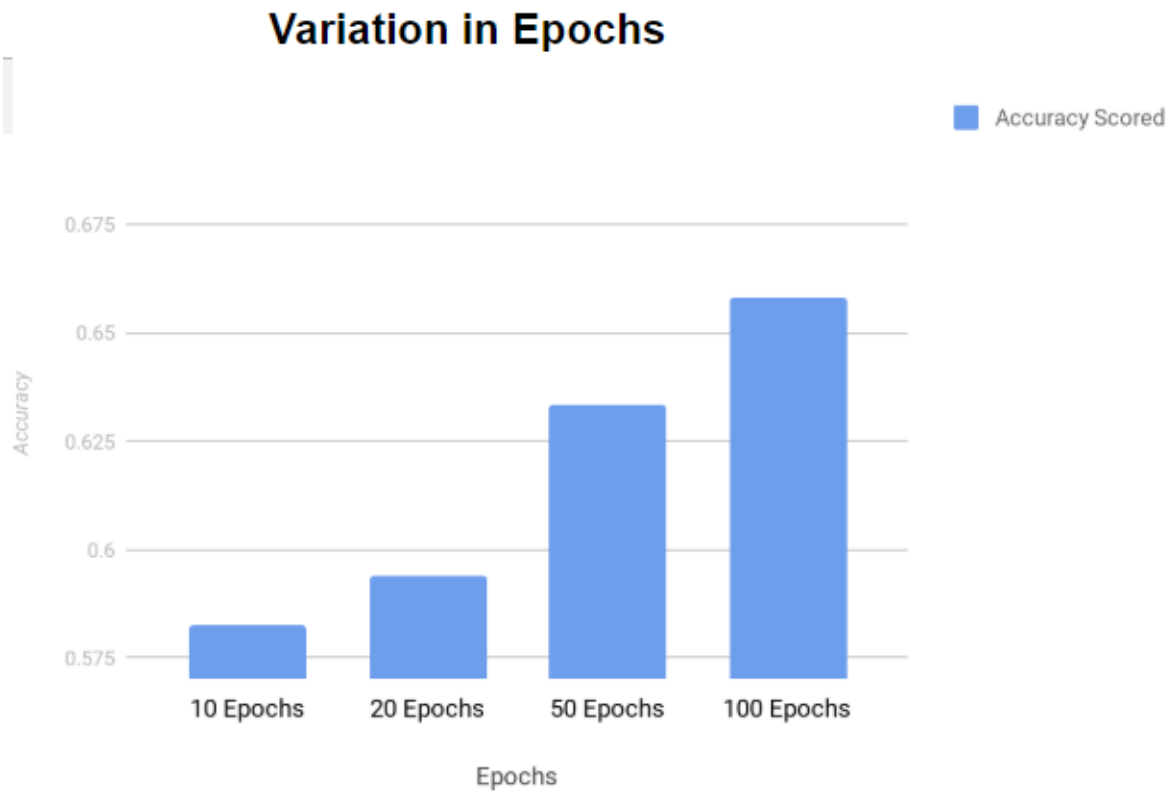
The whole point of Neural Network (or Gradient Descent) is to minimize the Cost function efficiently. Here we see the Neural Network Model's improved cost after each individual epoch. The figure resembling a hyperbola suggests a good backpropagation and reasonably efficient Neural Network Model.



2.5.3 Varying Epochs

Epochs are the number of loops the training data is feeded into the Neural Network Model. Epoch iterations are a key element to get a Neural Network more precise output, with reference to the training data. Here, we vary epoch from 10 to 50, keeping everything else constant, to get an overview of the varied accuracy with respect to number of Epochs.

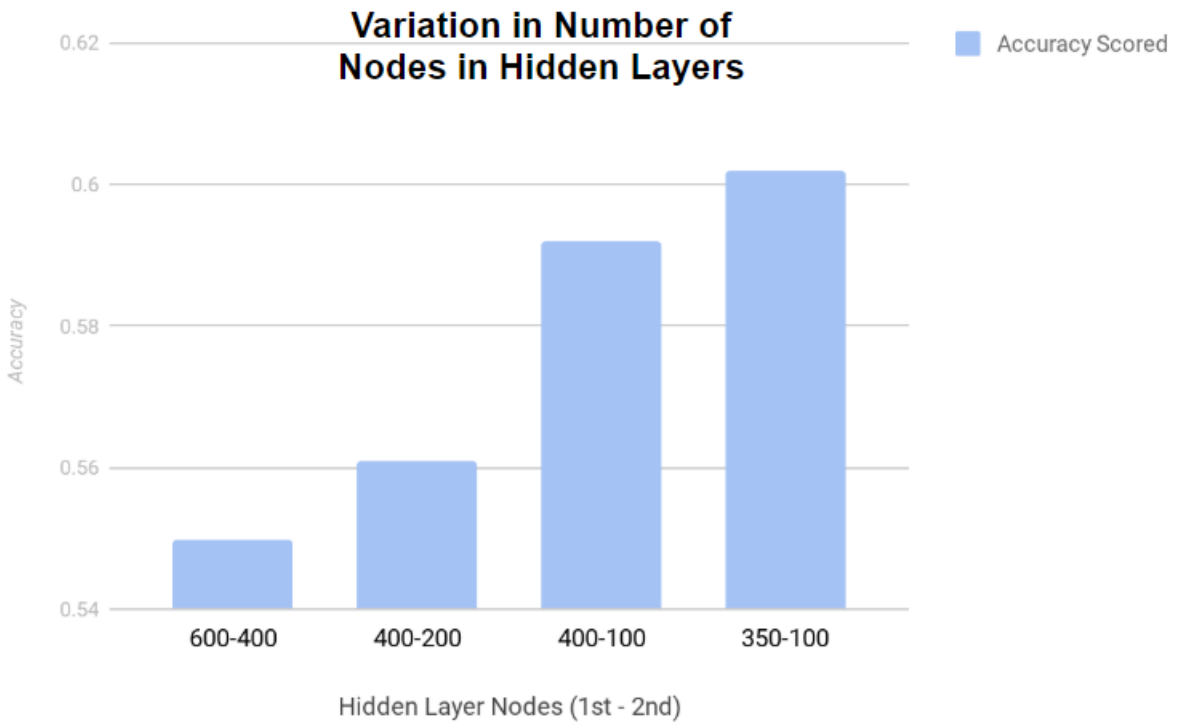
Number of Epochs	Variable
Number of Hidden Layers	2
Number of Hidden Nodes	350, 100



2.5.4 Varying Hidden Layer Nodes

Number of hidden layer builds up complexity of the Neural Network Model. More the number of hidden layer's nodes, the more inter-node connections are required, ie. more Weights and Biases. This elevation in complexity shall help in complex sentence's determination of sentiments, but requires more computation in creating the model.

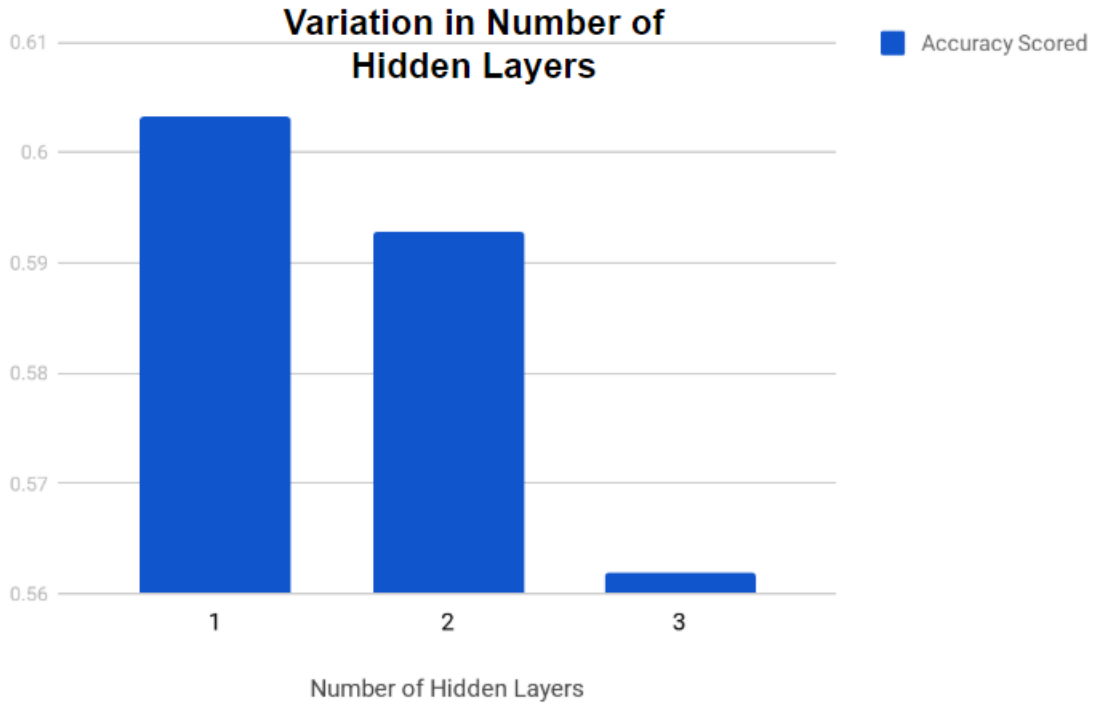
Number of Epochs	25
Number of Hidden Layers	2
Number of Hidden Nodes	Variable



2.5.5 Varying Number of Hidden Layers

Similar to varying the number of hidden nodes, adding more hidden layers contribute significantly in making the model more complex. However it is evident from the resulting graph that more layers results in less accuracy, since more time will be needed to train the extra variables induced due to variable hidden layers.

Number of Epochs	25
Number of Hidden Layers	Variable
Number of Hidden Nodes	350 - 100



2.6 Conclusion

From the above results we can conclude that the effect of either of the parameters is not that significant, as it's in the margin of error of the random value of the weights and biases in the initialization process since the Standard Deviation of the random tensor is 1.0.

```
tf.random_normal
(
    shape,
    mean=0.0,
    stddev=1.0,
    dtype=tf.float32,
    seed=None,
    name=None
)
```

Along with this, we can conclude that most of the deviations of accuracy, which was withing +-5, is not note-worthy.
Since Neural Network's heart lies in it's data, we strive to get more data to train our Neural Network. In the next chapters to come, we move from data of just 5331 lines of unstructured Natural Language, to 1.6 Million lines of unstructured labelled Data from Tweets.

Bibliography

[1] Q. Cao, W. Duan, and Q. Gan. Exploring determinants of voting for the helpfulness of online user reviews: A text mining approach. *Decision Support Systems*, 50(2):511–521, 2011.

[2] C. C. Chen and Y.-D. Tseng. Quality evaluation of product reviews using an information quality framework. *Decision Support Systems*, 50(4):755–768, 2011.

[3] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391, 1990.

[4] A. Fahrni and M. Klenner. Old wine or warm beer: Target-specific sentiment analysis of adjectives. In *Proc. of the Symposium on Affective Language in Human and Machine, AISB*, pages 60–63, 2008.

[5] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM, 2004.

[6] N. Hu, I. Bose, N. S. Koh, and L. Liu. Manipulation of online reviews: An analysis of ratings, readability, and sentiments. *Decision Support Systems*, 52(3):674–684, 2012.

[7] H. Kang, S. J. Yoo, and D. Han. Senti-lexicon and improved naïve bayes algorithms for sentiment analysis of restaurant reviews. *Expert Systems with Applications*, 39(5):6000–6010, 2012.

[8] S.-M. Kim and E. Hovy. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1367. Association for Computational Linguistics, 2004.

[9] Y.-M. Li and T.-Y. Li. Deriving market intelligence from microblogs. *Decision Support Systems*, 55(1):206–217, 2013.

[10] I. Maks and P. Vossen. A lexicon model for deep sentiment analysis and opinion mining applications. *Decision Support Systems*, 53(4):680–688, 2012.

[11] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, 3(4):235–244, 1990.

[12] S. Mohammad, C. Dunne, and B. Dorr. Generating high-coverage semantic orientation lexicons from overtly marked words and a thesaurus. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 599–608. Association for Computational Linguistics, 2009.

[13] R. Moraes, J. F. Valiati, and W. P. G. Neto. Document-level sentiment classification: An empirical comparison between svm and ann. *Expert Systems with Applications*, 40(2):621–633, 2013.

[14] G. Qiu, X. He, F. Zhang, Y. Shi, J. Bu, and C. Chen. Dasa: dissatisfaction-oriented advertising based on sentiment analysis. *Expert Systems with Applications*, 37(9):6182–6191, 2010.

[15] J. Read and J. Carroll. Weakly supervised techniques for domain-independent sentiment classification. In *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, pages 45–52. ACM, 2009.

[16] M. Van de Camp and A. Van den Bosch. The socialist network. *Decision Support Systems*, 53(4):761–769, 2012.