Mapping Text-based Review to Star Rating

Supervisor Dr. Amna Dridi

Nirvan kamble (21133614)
Master project

Contents

List of figures:

Abstract:

Before using a service such as going to the movies or making a purchase such as a product in today's day and age, it is absolutely necessary to read reviews first. It is essential to conduct sentiment analysis in order to determine how individuals feel about a specific product, movie, commentary, etc. Online retailers face the additional challenge of dealing with fake users who post misleading reviews of products and give them low ratings in order to harass legitimate retailers and profit illegally. Sentiment analysis is a method of opinion analysis that makes use of the concept of Natural Language Processing (NLP) in order to provide us with information about how customers will behave in relation to a product. The three types of reviews that are most frequently encountered are positive, negative, and neutral. The reviews contained in this work were evaluated using word embedding techniques like TF-IDF, Word2vec and BERT. Instead of dividing the reviews into positive, negative, and neutral categories, we assigned each review a rating out of five stars. And also, we will build model to detect fake reviews Using deep learning models and machine learning models.

Keywords – Word Embedding, Word2vec, TF-IDF, BERT, SVM, LSTM, Imbalanced data.

Acknowledgement

To begin, I would like to convey my deepest thanks to my supervisors, Supervisor Dr. Amna Dridi for their constant support throughout the duration of this master's dissertation, in addition to their patient guidance and extensive knowledge. she provided helpful recommendations and words of support, but I also want to thank her for bringing up challenging issues that motivated me to expand the scope of my research and consider it from a variety of different points of view. I would like to express my sincere appreciation Dr. Amna Dridi for providing me with the opportunity to have direct conversations with various university stakeholders. Because of this, I was able to provide constructive comments as well as project recommendations.

In closing, I would want to express my gratitude to my entire family as well as my close circle of friends for all of the support they have shown during this endeavour.

1. Introduction

1.1 Background:

Opinion mining is an extension of information mining that involves extracting people's perspectives on a given topic and then effectively breaking them down, whereas the goal of opinion investigation is to uncover the hidden feelings or, on the other hand, feelings that are present in the evaluation. Opinion investigation seeks to uncover these feelings. People's actions in relation to a specific product or administration can be controlled from a variety of perspectives using techniques such as opinion mining and assessment mining. Both of these methodologies are essential elements that make up a dynamic cycle. Written reviews and star ratings of commercial businesses have been used for quite some time to gauge the level of satisfaction and dissatisfaction felt by customers. Customers are more likely to purchase a product if it has a high number of positive reviews and a rating of five stars or higher. However, occasionally customers will leave star ratings that do not correspond with the overall sentiment of the review that was written. They might give a service or product a high rating, but then write reviews in which they are extremely critical of that service or product. A disparity in ratings of twenty percent was discovered after 10 thousand reviews of apps for Android were read by hand and analysed in a study. According to the findings of a study that investigated the detection of polarity mismatches, comments with moderate levels of severity tend to report more instances of polarity disagreement than comments with severe levels. Following the completion of the review analysis, a system employing machine learning architecture will be developed with the purpose of predicting discrepancies between customer ratings and reviews. When conducting an analysis of comment ratings, it is important to take into account the inherent qualities of the content being reviewed. The tone of the comment is, in our view, one of the most significant aspects that could be used in research to more effectively address this issue. This is one of the most significant elements that could be used. This strategy would result in more accurate rating forecasting, which in turn would assist users in selecting the most appropriate option. Another challenge that digital retailers must overcome is the fact that some users who are not authorised to do so do use the feedback system in order to illegally make money. These dishonest consumers, for example, leave deliberately negative reviews and low ratings on the things they consume, paying no attention to the effects that the goods' quality has on them. Later, criminals threatened to blackmail digital merchants in exchange for illegal gains in order to cause damage to the digital merchants and deceive regular consumers about the products in suggestions. This was done in order to harm the digital merchants and deceive regular consumers about the products in suggestions. On the other hand, dishonest individuals will scam e-commerce platforms by leaving excessively negative reviews. Because of this, dishonest customers put the credibility of online commerce as a whole at risk. Additionally, as a result of this negative publicity, recommendation engines will continue to be confused, which will result in disorganised recommendations for regular consumers, which are also known as scamming attacks. There are models that are able to recognise review masking, which allows for the problem described above to be solved. They give a positive rating, but they also give a negative review in order to prevent themselves from giving an excessive number of low ratings. As a consequence of this, their strategy may be puzzling to a prospective buyer who is

considering whether or not to purchase a product at the time they are reading this feedback. People will give a low score but positive feedback in order to avoid giving a significant amount of negative feedback. Criminals will use the technique of "features extraction" in order to justify their interactions by claiming that they were "mis-operations." The two methods that were just described make it possible for malicious users to impersonate regular users. The potential attacker who possesses a high level of skill has two separate goals in mind. 1) The overall rating and review level of a website is taken into consideration when determining the credibility of its merchants. Because of these ratings and reviews, an experienced cybercriminal with malicious intent had the opportunity to abuse the feedback system by posting fictitious poor ratings and negative reviews, which put in jeopardy the illegal profits made by the merchants. On the other hand, the websites ought to regard a user as malicious if they repeatedly give a high percentage of poor or negative ratings for no apparent reason at all. By utilising techniques for the detection of outliers, the company may be able to identify the fake user. It can be difficult to identify potential attackers because criminals may use a masking technique to manage the percentage of negative reviews or low ratings in order to prevent being discovered as fake accounts. This makes it more difficult to identify criminals. 2) A potential attacker may provide phoney ratings or reviews that are difficult to differentiate from genuine ratings in order to mislead users. The result of this is that the algorithms currently used for making recommendations produce recommendations that are unreliable, misleading, and ineffective. In the event that we are able to recognise fake users and get rid of fake reviews, ratings, and ratings, then the effectiveness models that are recommended should be improved. This paper intends to transform a text-based review system into a star-based system by addressing two significant issues—fake reviews and polarity mismatch reviews—and by doing so, it hopes to achieve its goal. In order to accomplish this, we will first extract features from the data utilising a natural language processing method, and then we will choose the most appropriate machine learning model from among a selection that includes SVM and deep learning model like LSTM.

1.2 Problem Statement:

When it comes to websites, user ratings and written reviews do not always correspond to one another. The most effective answer to this problem is an intelligent system that provides assistance to users in accordance with the requirements and priorities that they have specified. Because star ratings are such valuable additions to review information, it is essential to research intelligent review rating system prediction in order to accurately forecast review ratings.

The vast majority of polarity mismatches in reviews and star ratings are the result of fake users posting fake reviews in order to illicitly increase their own profits. Fake users will post positive reviews but will rate the product poorly, and vice versa, in an effort to avoid being discovered. Because reviews can be deceiving, it is imperative that these problems be fixed before moving on to converting text-based reviews into star ratings.

The majority of the models that have been developed find it much simpler to detect positive sentiment than negative sentiment. As a consequence of this, our analysis is biased, which may have an effect on the way in which we translate ratings into star ratings.

1.3 Aim:

The goal of this project is to propose a Machine Learning-driven method for mapping text-based reviews to sentiment scores.

1.4 Objectives:

• To experience polarity mismatches as a result of distinguishing between biased and unbiased data.

• To develop a model for converting a text review into a star-based rating using natural language processing (NLP).

• Multiple model comparison for better fit

1.5 Structure of the thesis:

This dissertation aims to develop, build, and test a system that will aid in the improvement of online review rating, making it a safer place for a seller to sell his product and a much safer place for customers to make a good purchase by providing genuine reviews. This project's goal is to detect fake reviews and predict text-to-star conversion. To accomplish this, it discusses the use of a variety of word embedding techniques in conjunction with machine learning and deep learning algorithms in the detection of fake reviews and text-to-star prediction.

The chapters are summarised below, with longer, more detailed explanations available further down in the various chapters. The main argument of this dissertation is that approaches like word embedding, data augmentation, and data pre-processing can all be used to build a model that can differentiate between fake review and star rating prediction in order to solve the problem that previous research had with fake detection and star rating prediction**.** The following outline constitutes the structure of the dissertation:

• Chapter 2 of the theoretical background provides extensive information on the deep learning strategies and algorithms that were utilised in the research.

In Chapter 3, the part titled "Related Work," we detail the research that was necessary to understand our work and how it relates to the existing body of study.

• The Methodology, which is essentially an expanded version of the Project Proposal, is discussed in Chapter 4. outlines the process that was used to create the system based on the concepts that were proposed in the chapter 3 conclusions.

In Chapter 5, the results of the experiments that were conducted to determine whether or not the algorithms were correct are presented, followed by functional testing.

• In Chapter 6, the evaluation that was presented in Chapter 5 is expanded to cover the outcomes of the project.

In Chapter 7, the findings and recommendations for the further steps are presented.

1.6 Why star rating prediction is so important

One of these three feedback systems is utilised by the vast majority of companies operating in the technical sector. In addition to the more traditional text-based feedback submission, this also allows for the submission of star ratings and reactions using emoji.

A. criticism in its various forms

The most effective method for gathering feedback is to send out a series of specific questions and follow up with a request for in-depth responses. This should only be given to one person, and that person should be someone we know for a fact will take the time to provide feedback. It is essential that consumers keep visualising uses that are ideally suited for our product. They may have the impression that your software or application is missing certain features, that the user interface could use some improvement, or that there is a bug. They won't submit an excessive number of tickets into our ticket support system. This happens only when the problem is of a significant magnitude.

B. Rating

The contribution of customers is absolutely necessary for the expansion of businesses' revenue. For a brand to retain its customers, the overall quality of its products and/or services must meet or exceed the expectations of those customers. One of the simplest things we can do to ensure that our customers are satisfied is to inquire about their thoughts and opinions. It is imperative that competitors provide feedback in order for a company's brand to build and maintain a solid reputation among its competitors. Not only does it give us valuable information about what our customers prefer, but it also has the potential to assist the company in the creation of new services and products. Companies are smart to take advantage of the fact that customer feedback is significant in order to improve their products and services and increase overall customer satisfaction.

C. Purpose

A great number of organisations make use of various methods, such as connections on social media platforms, Google forms, website forms, and even their very own web applications, in order to collect feedback. The Excel workbook that will be used for the analysis of the feedback is uploaded by the brand feedback analyst and then sent to the processor. Quick results of the complete answer are generated by the feedback analysis system in accordance with the function that is selected (Sentiment, Emotion, Intent, Keyword). The primary goal of the proposed framework is to provide feedback analysis by making use of NLP algorithms in an approachable way. Another objective is to classify the overall feedback into four key themes that are required by the majority of organisations. Text feedback allows users to leave comments in the form of words rather than emoji or stars, and the process is very similar.

2. Related work:

Conducting sentiment analysis by mining textual data is one of the most frequent and successful approaches to extract relevant information from user evaluations. Sentiment analysis entails categorising structured language as positive, neutral, or negative to investigate the user's perception of a product, service, or application (Cambria et al., 2017). A lot of studies in the subject of sentiment analysis have been conducted utilising machine learning algorithms on user reviews, with promising results considering 3 scale ratings. To provide better understanding of the user feedback, a 5-scale star system could be more advantageous

Pervious literatures performed the classification of numeric rating into either "Biased rating" or "Unbiased rating" to categorize customer's review (Sadiq et al., 2021) and one of the papers as extended it to star classification or prediction using deep learning approach (Sadiq et al., 2021). In this paper, we sought to transform unbiased data into a numeric star rating for better estimation user's feedback using different machine learning approach.

Since many buyers rely on text data while making purchases, numerical ratings are generally combined with textual reviews. Nonetheless, such textual information has generally played a secondary role, with the ratings receiving most of the attention (Beuscart, Mellet, & Trespeuch+--, 2016; Luca, 2011)

Even though multiple studies have shown that emotional expression in online evaluations affects sales (Ren & Nickerson, 2014; Ullah, Zeb, & Kim, 2015). For these reasons, various approaches in the MPC challenge have evolved, attempting to capture users' preferences by inferring numerical ratings from textual reviews (Ganu et al., 2009; Pang & Lee, 2005). For increasing more accuracy and speed this paper proposed the Virtual machine concept for the prediction of the review in star rating they used combination of naive Bayes and neural network for better accuracy by using the Lexicon star-based rating over cloud with embedding the naive Bayes and neural network they reviewed faster text per ms and they got very promising result in very less time (Kumar and Kumar, 2018)

Included a star rating to text sentiment polarity. They discovered that ratings deviate greatly from user reviews at times. They presented a standard strategy to solving this problem by considering five stars rating and numeric rating. They determined the sentiment polarity value by taking the average of the star rating and the numerical rating. They presented a diversified approach that can be used for sentiment classification in many fields, as reviews differ amongst apps (Islam, 2014).

Review spam, also known as fake reviews, has become increasingly common in recent years. a lot of focus in recent times the ability to spot fake reviews is developing new levels of sophistication. Popular area of investigation, and a plethora of findings have been obtained (Fang, Wang, Zhao, Yu & Wang, 2020).

There have been both unsupervised and semi-supervised methods used. (Wang et al., 2020) proposed. investigated the problem of fake news. Review detection using a model of the sequence that includes the both the content of the review and the reviewer's behaviour Following investigation, He built his model using six time-related characteristics that he extracted.

(Neisari, Rueda and Saad, 2021) Detection of spam through review utilising convolutional neural networks as well as self-organizing maps Computers & Security.

3. Thermotical methodology

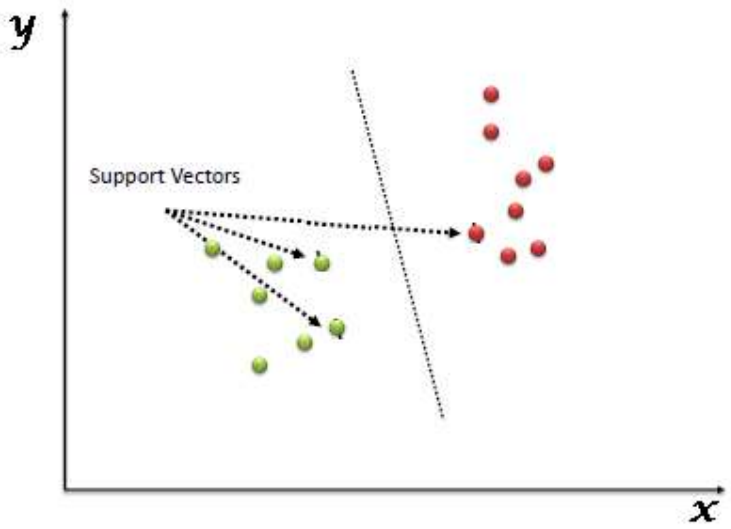
3.1 Data Augmentation:

Our performance will be better the more data we have. Annotating a substantial amount of training data, on the other hand, is a luxury. As a consequence of this, making use of relevant data augmentation in your model can help it perform better. Augmentation is a method that is frequently used in computer vision. It is possible to easily improve images by flipping them, adding salt, and utilising other features with the assistance of image augmentation libraries such as imaging. It is well known that one of the key components that contributes to the success of computer vision models is known as augmentation. Text augmentation is a challenging task in the field of natural language processing (NLP) because of the high level of complexity that language possesses. It is not possible to substitute certain words for others, including a, an, and the. In addition, not every word can be replaced by another word. Even a small change in word choice can have a significant impact on the overall meaning. On the other hand, the field of computer vision makes the process of creating an augmented image relatively straightforward. Even though there is noise in the image or parts of it have been cropped out, the model is still able to correctly classify it. The authors tried a variety of approaches to achieve the same goal of generating more data for the purpose of model training given that we do not have an infinite number of humans to use as a source of training data. In this discussion, we will investigate the methods that different authors use to augment natural language processing (NLP) tasks by generating additional text data in order to strengthen the models.

3.2 Traditional machine learning model:

SVM (For the classification of fake reviews)

The "Support Vector Machine" (SVM) is an example of a useful supervised machine learning algorithm that can be utilized for classification and regression tasks respectively. However, the most common application for it is in the context of classification problems. In the SVM algorithm, the value of each feature is represented by the value of a specific coordinate, and each piece of data is plotted as a point in n-dimensional space. n is the number of dimensions (where n is the number of features you have). The next step in the classification process involves locating the hyperplane that provides the clearest demarcation between the two classes (look at the below figure).



To determine a support vector, the coordinates of each individual observation are used in the computation. The SVM classifier can be thought of as a boundary that, in practice, divides the two categories (hyper-plane and line).

SVM for text classification

Given that the goal of this project is to identify fake reviews and Because of SVM, it is possible to classify the vectors that exist in multidimensional space. The first thing that needs to be done in order to use this algorithm for text classification is to convert a section of text into a vector of numbers so that the support vector machine can analyze it. What characteristics, to put it another way, have to be present in order for SVM to be able to classify texts?

The most common response is to look at word frequencies, which is what Naive Bayes does. As a consequence of this, every word in a text is assigned a feature, and we approach a text as if it were a collection of words. The significance of that characteristic will be determined, in large part, by the number of times that word appears in the text.

This method consists of doing nothing more than counting the number of times each word appears in a given text and dividing that total by the total number of words. In the sentence,

"All kitchen set are primates, but not all primates are kitchen set," the word kitchen set appears two times out of ten, or 0.2, while the word but appears only once out of ten, or 0.1.

We can also use TF-IDF, which is an approach that is more complex than other ways of calculating frequency.

The following thing that needs to be done is to make a representation of every single text that is included in our dataset in the form of a vector that has thousands (or tens of thousands) of dimensions, each of which represents the number of times a different word appears in the text. Perfect! This is input into the SVM so that it can learn. We can make this situation significantly better by utilizing preprocessing techniques such as stemming, stopping words, and n-grams. According to the nature of the dataset, there are two distinct types of support vector machine algorithms: linear and non-linear. On the other hand, for this specific project, we will use linear SVM.

WHAT Exactly Is a Linear Support Vector Machine?

A linear support vector machine (SVM) is used on a dataset that can be partitioned linearly and only contains one feature, as the name suggests. A linear SVM classifier is used to break the dataset up into its individual features. Consider a dataset that contains only one feature, like the weight of each individual. In this particular scenario, the support vector machines algorithm should have no trouble generating a hyperplane. The support vector machine (SVM) analyses newly added data to determine which side of the hyperplane it falls on, which is then used to determine whether or not a review is fake or is real. Within the scope of this project, the linear kernel has been applied. The syntax is **K (x, y) = sum(x*y).** Both X and Y can be thought of as vectors. Text classification is an example of a separable dataset that makes use of this technique because it has a large number of features. The linear kernel is the one that processes data at the highest speed.

3.3 Deep learning model

LSTM

For long-term and short-term memory, use LSTM. The recurrent neural network known as the long short-term memory (LSTM) uses significantly less memory than its more conventional counterparts. LSTMs are significantly more effective than other types of models because they are good at remembering particular patterns. As the LSTM, just like any other NN, travels through each hidden layer, the information that is pertinent to the problem at hand is remembered, while any and all information that is not relevant is discarded in each and every cell.

Why LSTM?

The ability to remember things in the short term is challenging for standard neural networks. The problem with the gradient that quickly disappears is yet another significant disadvantage. (The gradient will decrease during backpropagation until it is almost equal to zero; this will render the neuron ineffective for further processing.) LSTMs improve performance in a number of ways, including the effective memorization of essential information and the identification of patterns.

LSTM for text classification:

Why use LSTM for text classification when there are numerous other traditional classification algorithms that are just as effective, such as SVM, RFR, and decision trees? One justification for employing LSTM is the fact that it is effective in assisting individuals in remembering important information. Because they are trained on multiple words as separate inputs, other classification methods that do not use neural networks make their predictions about the class based on statistics rather than meaning. This is because the words do not have any actual meaning when combined into a sentence. In a different way of putting it, each word is placed into one of the categories. This is not the case with the LSTM method. In LSTM, a multi-word string may be substituted for a single word to determine the category to which it belongs. This is of great assistance when dealing with natural language processing. If the LSTM is configured with the appropriate layers of embedding and encoding, the model will be able to determine the exact meaning of the input string and provide the highest level of accuracy in its output class. The following constitutes the LSTM's components:

Forget Gate f (NN with sigmoid as activation function).

Candidates for the gth Layer (NN with tanh as activation function).

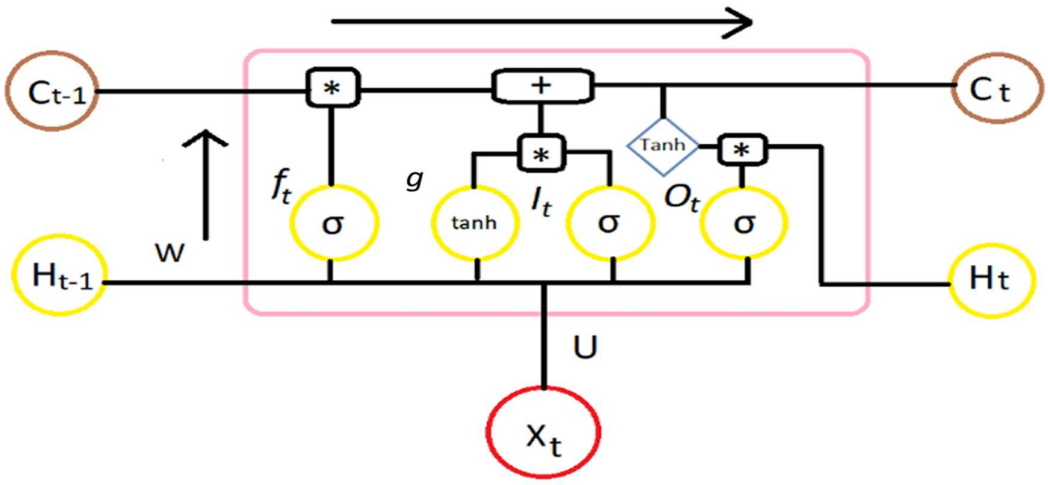
Input Gate I (NN with sigmoid as activation function).

Output Gate O (NN with sigmoid as activation function).

Hidden State H (vector) (vector).

Memory State C was used (vector).

This is the LSTM diagram for **the t-time step**, which can be found below.



(X t) = Input vector for the Time-t variable.

Previous state of concealment (H t1)

The prerequisite for retrieving older memories is (C t1).

The current hidden state can be determined by looking at (H t).

The phrase "Current Memory state" can be shortened to (C t).

[*] denotes a multiplication operation in the mathematical expression.

The sign for addition in mathematical notation is [+].

In light of this, the input of every LSTM module is made up of (X t) (the current input), (H t 1), and (C t 1). The answer is going to be (H t), and (C t).

Input, forget, and output gates are represented by the following symbols: (l, f, O). The only difference between the input, forget, and output functions is in the matrix parameters. All three functions share the same function formula (sigmoid). This suggests that the output of the gate is a vector whose values range from 0 to 1, as indicated by the previous sentence. A value of one indicates that all of the information is included, while a value of zero indicates that all of the information is completely blocked. The gate input will tell you how many of the states you have just calculated for the current input that you want to pass on to the next state. The forget gate is responsible for deciding the maximum number of prior states that should be allowed to pass. When all is said and done, the number of internal states that are visible to the network is decided by the gate output (higher layer & next time step). Every gate has the exact same measurements as the hidden state, serving as a measurement for the hidden state itself (etc.). The output of the sigmoid gate will be multiplied by the value in question in order to get a better idea of how much of the other value is being used.

(I= \sigma (x_tU^I + s_{t-_1} W^I))

(f= \sigma (x_tU^f + s_{t-_1} W^f))

(O= \sigma (x_tU^O + s_{t-_1} W^O))

(g) represents a "candidate" hidden state because it takes into account both the current input and the previous hidden state.

The memory contained within the device is (c t). The previous memory (c t-1) is multiplied by the forget gate, and the newly computed hidden state g is multiplied by the input gate in order to produce it. As a consequence of this, it combines previous memory with new input in a manner that is straightforward.

(c_t=f_t*c_{t-_1} + I_t*g)

BERT:

BERT stands for Bidirectional Encoder Representations from Transformers. The utilisation of bidirectional training of Transformer, which is a well-liked attention model, for the purpose of language modelling is the primary technical advancement made by BERT. Studies that were conducted in the past looked at either a right-to-left text sequence or a combination of right-to-left and left-to-right instruction. The findings of this research show that language models that are trained in only one direction are unable to comprehend the context and flow of language to the same degree as language models that are trained in both directions. The researchers present a novel method that they call Masked LM (MLM) in the paper that they wrote about it. This method enables bidirectional training in models, which was not possible in the past.

How BERT Works?

Transformer is an attention mechanism that is utilised by BERT. It is responsible for discovering contextual connections between words or sub-words in a text. In its most fundamental manifestation, Transformer is made up of two separate mechanisms: an **encoder**, which is responsible for reading the text input, and a **decoder**, which is responsible for producing a task prediction. Because the end goal of **BERT** is to produce a language model, the only component that is required is the encoder mechanism.

The Transformer encoder does not read the text input sequentially like directional models do; rather, it reads the entire sequence of words all at once right-to-left or left-to-right. As a consequence of this, it is considered to be bidirectional, despite the fact that a more accurate

description would be non-directional. This function enables the model to comprehend the context of a word by looking at the elements that are immediately adjacent to it (left and right of the word).

The Transformer encoder is outlined in considerable detail in the chart that can be found below. A number of tokens are inserted into vectors before the input is processed by the neural network. It can be shown that each H-dimensional vector in the output corresponds to a single token in the input that has the same index.

It can be difficult to define a prediction objective when developing language models. The directional approach taken by many models, which predicts the next word in a sequence, is incompatible with context learning and therefore should be avoided (for instance, "The child returned from --------- "). **BERT** employs two different types of training in order to overcome this obstacle:

1: Masked LM (MLM)

2: Next Sentence Prediction (NSP)

The Sentence Prediction Training Model is applied here for the purpose of the project.

Next sentence prediction:

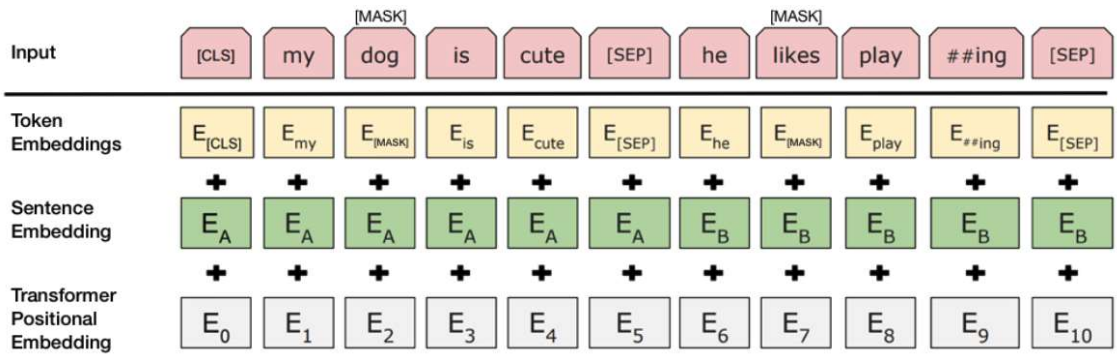
During the training phase of BERT, the model is presented with sentence pairs and instructed to determine whether or not the second sentence in each pair corresponds to the following sentence in the source document. This is accomplished by feeding the model sentences in pairs. During training, half of the inputs will be sentence pairs in which the second sentence is the next sentence in the original document, and the other half will be randomly selected sentences from the corpus. Input pairs will make up half of the inputs. It is presumed that the seemingly unrelated sentence does not have any connection to the one that came before it.

The input is processed in the following manner prior to being fed into the model in order to assist the model in differentiating between the two sentences while it is being trained:

There is a [CLS] token at the beginning of the first sentence, and there is a [SEP] token at the end of each subsequent sentence.

Tokens are assigned sentence embeddings, which are conceptually comparable to token embeddings but have a vocabulary size of only two, and these embeddings designate whether the token belongs to Sentence A or Sentence B.

A positional embedding will be given to each token so that its position in the sequence can be determined. The Transformer paper explains both the concept of positional embedding and the method for implementing it.


Input
[MASK]
[MASK]
[CLS] my dog is cute [SEP] he likes play ##ing [SEP]
Token Embeddings
E[CLS] Emy E[MASK] Eis Ecute E[SEP] Ehe E[MASK] Eplay E##ing E[SEP]
Sentence Embedding
EA EA EA EA EA EA EB EB EB EB EB
Transformer Positional Embedding
E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 E10

4. Research Methodology

4.1 Data set:

We were able to carry out this research by utilizing historical data that was openly accessible to the general public archives. The following is a list of the general criteria that we used in the selection process for product and service reviews: 1. Category of kitchen and home product 2. label with content 'CG', "OR". CG stands for computer generated review which is fake and OR stands for Human write review which is considered as real 3.text review given by the customer. Secondly The purchase is not an insignificant one for the consumer, and thirdly the option in regard to the product or service appeals not only to the customers' rational side, but also to their emotional side components. 4. Star rating of the product

4.2 Data visualize:

We also visualize the dataset in such a way that you can see the numerical star ratings that were given by the users. It can be seen in Fig. 2 that the majority of ratings received from users were a 5. This rating appears to be skewed or to have been made up. The numerical rating that unidentified users gave review rating is an important issue that needs to be looked into. The frequency of numeric star ratings in each category was analyzed, and the results are depicted in the graph. Our findings revealed that casual mobile applications had a high numeric rating

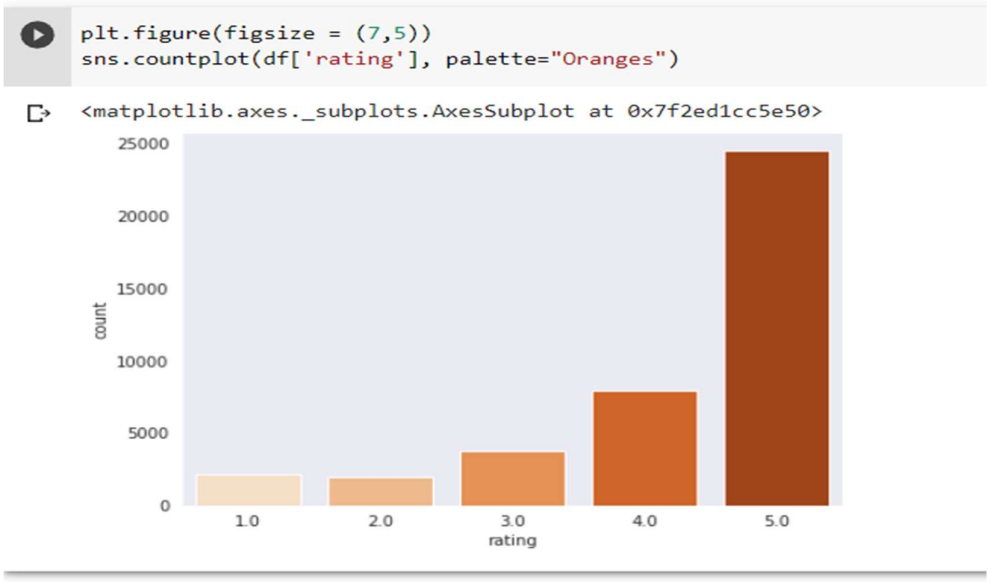


Figure 1 star rating

4.3 Data Pre- processing:

Before the text is converted into a vector, it must first go through some pre-processing steps. During pre-processing, tasks such as filtering (which involves removing punctuation, hyphens, and extra spaces), converting uppercase letters to lowercase letters, correcting spelling, removing stop words, and stemming are carried out (replacing a word with its semantic base). The Python 3 Spelling Corrector is a tool that can be used to make spelling corrections. It was determined that 421 stop words could be removed from the English text that was on the Fox list for stop word removal. As delimiters, whitespace and characters that aren't letters or numbers are used in tokenization. The following is how we apply the heuristic rules that are outlined such as (1) Make the necessary changes to these characters! "# $% & * = >? @ | separated from one another;" (2) Remove any ": ;" characters that appear following a space:, (3) Take the brackets off the () and the [] (4) remove the quotation marks from the phrase; (5) Take out the apostrophes and slashes if there is a space after them in the sentence.

4.4 Feature extracting:

Bringing out key characteristics this study makes a significant effort to mine the textual characteristics of reviews so that it can produce the best classification result possible. Before beginning pre-processing, linguistic characteristics of each review are collected. These characteristics include the number of words, stopwords, punctuation, and the average length of the word, the average length of the sentence, and the average length of the paragraph. additional characteristics of the language, such as after a certain amount of clauses, passive voice, nouns, adjectives, and verbs have been accumulated in the sentence. pre-processing. For the purpose of obtaining Fake review detection and Star prediction, this paper makes use of a pre trained word embeddings model like Word2vec, BERT and TF-IDF. The topic distribution as an indicator of the latent characteristics of the review topic Once the text features have been obtained, the next step is to select the top N most important features and eliminated other features based on the importance.

4.5 Evaluation metrics

Five evaluation criteria were used when analysing the experiment's results.

There was consideration given to various indicators. The acronyms are as follows: Acc (for accuracy), P (for precision), R (for recall), F (for Fscore), and AUC. The term "precision" refers to the same thing as "accuracy rate." rate, or the ratio of the number of questions that were answered correctly to the total number of questions. results to the entire result in the returned results. results to the returned results. retrieval. There is no difference between the recall rate and the recall rate, which means that the percentage of actual, accurate numbers contained in

the proportion of actual, accurate numbers that are returned by the search. the entirety of the data set (retrieved and not retrieved). Given that the There appears to be a contradiction between the accuracy and recall rates. The F-score allows for a more in-depth comparison of the two.

$$Acc = \frac{(T\,P + TN)}{(T\,P + FP + F\,N + TN)}$$

$$P = \frac{T\,P}{(T\,P + FP)}$$

$$R = \frac{T\,P}{(T\,P + FN)}$$

$$F = \frac{2P\,R}{(P + R)}$$

The value of TP indicates the total number of genuine examples among them.FP refers to the actual number of samples that tested positive for something else.FN stands for "actual number," which refers to the total number of false negative samples.The term "actual number of true negatives" is abbreviated as "TN." samples.

4.6 Word Embeddings:

The method of language modelling known as word embedding involves representing individual words or phrases as vectors of real numbers. The words are grouped together in order to produce a similar representation for words that have meanings that are comparable to one another. In order to construct the representation, the word embedding process first studies the relationship that exists between the words. This is accomplished through the utilization of a wide variety of methods, some of which include the co-occurrence matrix, probabilistic modelling, and neural networks. It has emerged as a central concept in natural language processing over the course of its development. One-hot vectors are a good example of a representation that is relatively straightforward. The dimensionality of the problem grows

proportionally with the number of words, but it is possible to use a single hot vector to represent each individual word. Word embeddings reduce the dimensionality of word vectors in a variety of ways, such as how words occur collectively, such as King and Queen, or words that can be used alternatively, such as car and vehicle, and so on. Word embeddings also take into account how words can be combined to form new words, such as car and vehicle. Because of this, words that are very similar to one another have very similar vector representations. The number of dimensions that the vectors have is thus decreased. Even though it's not difficult to make one-hot vectors, using them to represent a large collection of words isn't something you should do very often. This is as a result of the fact that it makes it possible to express similarities between words that are contained in the corpus. Word2Vec and GloVe are two examples of well-known word embeddings. The BERT word embedding is one that was developed most recently. There are two distinct categories that can be utilized to classify word embeddings. Some examples of frequency-based embeddings include the Count vector, the Co-occurrence vector, the HashingVectorizer, and the TF-IDF. Word2Vec, GloVe, BERT, and fastText are some examples of word embeddings that have already undergone pre-training. In this paper, we proposed the TF-IDF method for detecting fake reviews, as well as pre-trained models for predicting star ratings, such as Word2Vec and BERT.

4.7 TF- IDF:

ONE KEY WORD Significant words in the review text are given ratings based on the results of a term frequency-inverse document frequency (TF-IDF) analysis. In addition to this, TF IDF (1) The word "frequency" refers to the number of times an occurrence occurs in a given time period. negative document the word "frequency" can be found in a document. an analysis of how applicable a term is across the world, according to the When a word or phrase that has a high TF-IDF score appears in the text, the following is true: It is common for it to be mentioned in one article but not in others. is suitable for utilization in the capacity of a classifier. In addition, the rich semantic relationships in reviews can be captured by the deep embedding module through the utilization of N-Gram representation.

The number of times a word is used throughout the content of the document is referred to as its term frequency (TF).

TF is equal to the number of times a word appears in a given document divided by that number (Number of words in the document)

IDF stands for "inverse document frequency," and this process makes it possible to condense the size of words that appear multiple times in a document.

IDF is equal to log e (the number of documents divided by the total number of documents that contain the term).

TF-IDF essentially works by elevating the value of significant words or tokens that are present in the document.

Documents that are relevant to a search query are ranked by search engines using a formula called **TF-IDF.**

After combining **TF-IDF** and **LSTM** neural networks, our research revealed an outstanding level of **accuracy** for the detection of fake reviews in terms of your dataset.

4.8 BERT

Late in 2018, the authors presented their work in, in which they described the BERT word embeddings model. It is a novel model of pre-trained language representations that enables the tuning of word vector representations to the meaning that the word has in a given context, thereby overcoming the ambiguity problems that are associated with words. One of the most well-known examples is typically discussed in conjunction with the word bank. Compare and contrast the following two sentences: "The man was accused of robbing a bank" and "The man went fishing by the bank of the river." Because the newly proposed word embedding models use the same word embedding to describe the word bank—that is, they express all the possible meanings of a word using the same vector—they are unable to differentiate between the various senses of a word based on the context in which it is used. On the other hand, BERT generates two distinct word embeddings, each of which is capable of producing representations that are more accurate when applied to the respective meanings. In order to accomplish this goal, BERT computes context-tuned word embeddings. As a result, the resulting representations are more accurate, which may result in improved model performances. The Bert 24 1024 16 BERT model that was trained on book corpus wiki en cased is used in this work. The model is then fine-tuned by using the Bert embedding library, which can be found at https://pypi.org/project/bert-embedding/ (accessed on 11 February 2021).

4.9 Emotional detection

Level in terms of emotion. This is the percentage of times that a review contains emotional words in comparison to other words, as described "emotionwords" is shorthand for "emotional intensity." All of the words Is there a correlation between the number and the emotional depth of the review? the number of words that are used to convey feelings throughout the review and integrated into the evaluation system The intensity of an individual's feelings has a negative correlation with the amount of the review is usually spot-on, although there are exceptions. As was previously stated, honest reviewers frequently commit errors in their evaluations. impartial assessments based on their own personal experiences False reviewers, on the other hand, have a propensity to recommend or subscribe to a product in an excessive manner, and as a result, they frequently use a greater number of words. distinct emotions, such as "wonderful," "terrible," "beautiful," and so on. “Poor.” The HowNet Emotion Dictionary is what we end up using as a result. Make a tally of the number of positive and negative words in the passage. First and foremost, emotional characteristics are easier to develop. using a dictionary as a reference source.

4.10 Sentimental Analysis

A person's emotions can be determined through the use of sentiment analysis by beginning with the content of the text that is provided. A person's thoughts, feelings, judgments, or perspectives can all be described using sentiment. Sentimental analysis, which is also known as opinion mining, has the ability to influence the ways in which people view particular entities. The information that can be found on the internet is extensive. Interpretation of the tone of the text. The concern of understanding is at the heart of emotional analysis. the reason behind the user's actions the analyst compiles information from social media platforms, online sources, and tweets. uses it to analyze data from the real world obtained from online communities such as blogs, forums, comments, and user reviews. meaning as well as how the general public evaluates what they have to offer We were successful in obtaining the data frame consisting of emotionally positive and negative reviews. analysis. On the other hand, before we began, we learned from other papers that sentimental When a review is written in a sarcastic tone, it is impossible for classification to understand what the reviewer is trying to say. written. Take, for example: If the review says something along the lines of "This place was so good that I would never come back here," then it won't be read. create the appearance of a positive attitude even though the actual meaning of the sentence is negative We suggested applying part-of-speech (POS) tagging and n-gram classification to our dataset in order to correct that misclassification comments made by users.

4.11 N-gram classification:

In activities that involve natural language processing, n-grams are utilised quite frequently. It is In a nutshell, they are a compilation of sentimental analysis aspect-level mining. during the process of analysing a given body of text using the n-gram method, one word, in particular, stood out. is advanced, which causes the word of the preceding n-final gram to become the first word. The ability to classify things in this way is helpful for comprehension. The positive-negative datasets have been translated into the text's actual meaning so that everyone can understand it. To improve the accuracy of the classification process, tried using unigrams, bigrams, trigrams, and quadgrams. Parameters.

4.12 Part-of-Speech (POS) tagging:

POS tagging" refers to the process of dividing words up into their respective parts of speech and then labelling them with the appropriate category. Lexical categories and word classes are also utilized, in addition to the various parts of speech, It makes use of a method known as aspect-level sentimental classification. POS tagging involves analysing each word in the

sentence to determine its function within the sentence, such as whether it is a noun, verb, conjunction, etc. This makes it easier to comprehend the true meaning of the text and can be applied to the process of constructing a machine train. After performing high-level text classification on each individual review taken from the raw dataset, we then used POS tagging to improve and more precisely predict how users would rate the product.Once the POS tagging and n-gram classification processes are complete, the final step in the analysis of sentiment is to determine the sentiment score for each review. We classified each word according to its positive or negative polarity using a system called sentiment polarity classification. Because of the POS tagger, which assisted in the process of preparing the dataset for machine learning, we were able to handle polarity classification with significantly improved accuracy and sentiment score.

5. Experiments and results

In this chapter, the reader will be presented with a comprehensive analysis of the primary findings acquired from each of the investigations that were carried out for the purpose of this study. an overview of the pre-trained and sequential models used in this work, along with a discussion of the parameters that were applied, is presented. we looked at the machine learning model as well as pre-trained models such as BERT, Word2vec, and TF-IDF to classify the fake detection and text to star rating prediction: one is balanced and the other is unbalanced, and the dataset is split into 70:30 for training and validation and 70:30 for training and testing for pre-trained models.

5.1 Experimental Environment

In order to carry out the experiments that were required for this study, the most recent release of the Python programming language was utilised. All of the experiments were executed on a Google Collaboratory system that was equipped with 64-bit Windows 10, an Intel i5-9300HF CPU operating at 2.4 GHz, and 8GB of RAM. Additionally, the deep learning tools TensorFlow/Keras and Scikit-learn were utilised during the testing process.

5.2 Result:

In this section, you will see how the text review classification was implemented, as well as the results it produced, by combining various word embedding methods with conventional machine learning models and deep learning models for the purpose of fake detection and star prediction.

5.2.1 For classification of fake review:

For the aim of detecting fake reviews, the following model has been proposed.

1. **Word2vec with LSTM.**
2. **Word2vec with SVM**.

5.2.2 For text review to star rating classification:

The following model has been proposed for text review to star rating which is multi classification**.**

1. **TF-IDF with SVM.**
2. **Word2vec with LSTM with pre-processing.**
3. **Word2vec with LSTM without pre-processing**.

4. **Bert.**

5.2.3 Classification of fake review:

Word2vec with LSTM

In this particular model, we initially made use of seaborn to determine whether or not the label was balanced, and we found that it was. After that, we pre-processed the data by removing all stop words and punctuation such as.,! $() *% @, Removed URLs, stop words, Lower casing, Tokenization, Stemming, and Lemmatization by removing punctuation such as.,! $() *% @, Removed URLs, stop words, Lower casing, Tokenization, Stemming, and Lemmatization by removing URLs, Removed URLs, stop words, Lower In the aftermath of the review, we updated the data frame with the clean text and gave it the name clean text. After that, the pre-trained word embedding model word2vec we used. In this classification we took two features as input 1. Text review 2. Star rating and output feature we took label.

. **Splitting into Train and Test Sets:**

The findings from this study data are divided in two parts: the train data and the test data, with 70% of the data found in the train part and 30% found in the test part. When splitting data, it is preferable to have a distribution of classes that is equal in both the train data and the test data. The function train test split that comes with the scikit-learn package is what's being used here.

Importing pre-trained model word2vec:

In this model, we will construct the model by utilising genism's Word2Vec.the following parameters we used to build the model.

size refers to the number of dimensions that the embeddings have, with 100 being the default value.

window refers to the maximum distance that can exist between a target word and the words that surround the target word. The standard viewport size is 5.

min count is the minimum number of times a word must occur before it can be considered for use in training the model; words whose frequency is lower than min count will be disregarded. The value of min count is set to 5 by default.

And we trained the algorithm with the skip gram.

In the following step, we reload the model in the variable to use the **"w2v model.wv. vocab"** to access the word2vec dictionary.

Generating Word2Vec Vectors

Word2Vec vectors are generated for each review in the train data as they are traversed through the X train dataset. Simply by applying the model to each individual word in the corpus under consideration, we are able to obtain the word embedding vectors for each word. To accurately

represent a sentence from our dataset, we take an average over all of the word vectors that are contained within a sentence. These vectors are stored in a format known as csv. This can be accomplished in a data frame in an immediate fashion After tokenizing the word, we add padded sequences to it before generating the embedding matrix for the embedding layer that will be used in the LSTM layer. The embedded layer is the first one, and it employs a total of one hundred length vectors to represent each individual word. In NLP models, the variational dropout operation is carried out by SpatialDropout1D.The LSTM layer comes next, and it has 100 memory units to its disposal. The output layer is responsible for producing two output values, one for each of the classes. In order to classify multiple categories, the activation function used is SoftMax. The categorical cross-entropy is used as the loss function because this is a multi-class classification problem. After compiling the model, we get the accuracy of 90.2%

Word2vec with SVM

Text Classification Using Machine Learning Models We Perform the Same Step That We Did in the LSTM Model the Only Difference Is That for the SVM Model, We Don't Need an Embedding Matrix After Getting word2vec, We Apply That Vector Directly to the SVM Model. For SVM model we get the accuracy of 69.75% of test accuracy.

5.2.4 Comparison of both model accuracy.

We found that the LSTM model was the best fit for the fake review classification after applying word2vec text classification to the same dataset.

Model	Train accuracy	Test accuracy
SVM with Word2vec with one feature as input (“text review”)	**25%**	**13%**
LSTM with Word2vec with one feature as input (“text review”)	**40.35%**	**32.75%**

SVM with Word2vec with two features as input ("text review" and "star rating")	**78.24%**	**69.97%**
LSTM with Word2vec with two features as input ("text review" and "star rating")	**98.75%**	**91.20%**

Key highlight:

When we used only the text review as the input feature, we discovered that the accuracy of the model was extremely poor as it was getting based on the data. It was unable to identify the polarity mismatch between the reviews and the star ratings.

5.3 Classification of text review to star rating

TF-IDF with SVM.

In this model, we initially made use of seaborn to determine whether or not the star rating was balanced, and we found that it was not unbalanced as the % star rating was more as compared to other star rating. After that, we pre-processed the data by removing all stop words and punctuation such as.,! $() *% @, Removed URLs, stop words, Lower casing, Tokenization, Stemming, and Lemmatization by removing punctuation such as.,! $() *% @, Removed URLs, stop words, Lower casing, Tokenization, Stemming, and Lemmatization by removing URLs, Removed URLs, stop words, Lower In the aftermath of the review, we updated the data frame with the clean text and gave it the name clean text. After that, word embedding model TF-IDF model we used. In this classification we took one features as input 1. Text review 2. Star rating and output feature we took star rating.

Balancing the data:

Following the execution of the TF-IDF vector on the text clean review, the text review will be vectorized after being processed. After the text has been converted, we will adjust the data so that it is more evenly distributed across the stars. We used SMOTE (), which will sample over under sampled feature, so that the data are more evenly distributed. And that will be adjusted according to all of our data.

Following the balancing step, we then applied the balanced data to the SVM model and obtained an accuracy score of 65.33 percent.

Word2vec with LSTM with pre-processing.

We repeated the same pre-processing step that we used in the earlier models, and we even used the same step that was described in the section that came before this one to generate the vector. This step is covered in more detail above. And accuacry we got after following this step was very poor which was the something interesting point. The accuracy which achieved was **45.5%.**

Word2vec with LSTM without pre-processing.

We repeated the step that we used in the earlier models, and we even used the same step that was described in the section that came before this one to generate the vector. This step is covered in more detail above. But this time we did not pre-processed which can change its grammatical meaning in the text which can later get difficult to the model to understand the sentiment within the text. And accuacry we got after following this step was good which was the something interesting point. The accuracy which achieved was **61.91%.**

Bert.

Tokenization is a process that takes raw texts and separates them into tokens, which are pieces of data that are numerical representations of words. Creates a tokenizer using the BERT algorithm. Using data from WordPiece Instantiate a pre-trained BERT model configuration to encode our data. We will process the train data and the validation data in two separate phases: first, we will convert all of the titles from text into encoded form using a function called batch encode plus, and then we will proceed with the conversion. The text of the title is passed in as the first parameter of the function that was just described. If add special tokens=True is given as an argument, it indicates that the sequences will be encoded with the special tokens that are relevant to their model. When we batch together a number of sequences, we make sure to set return attention mask=True. This causes it to return the attention mask in accordance with the particular tokenizer that is specified by the max length attribute. In addition to this, we want to pad out all of the titles to a particular maximum length. However, for the sake of caution, we will set max length=256 even though it is not strictly necessary. PyTorch will be returned if return tensors='pt' is specified. The next step is to separate the data into the input ids, attention masks, and label columns. Last but not least, once we have the encoded data set, we will be

able to create training data as well as validation data. The bert-base-uncased model is a more compact pre-trained version. Utilizing the num labels variable to indicate the number of labels to be output. The output attentions variable is of little concern to us. Additionally, we do not require output hidden states. Data Loaders,In order to generate an iterable that traverses the dataset that has been provided, the class known as DataLoader combines a dataset with a sampler. Validation is handled by the SequentialSampler, while training is handled by the RandomSampler. Because of the memory limitations in my environment, I decided to set the batch size to 3. Optimizer & Scheduler, in order for an optimizer to perform its intended function, it must be provided with an iterable that specifies the parameters to be optimised. Following that, we will be able to specify options that are unique to the optimizer, such as the learning rate, epsilon, and so on. The epochs=20 setting seems to work well with this data set, as I discovered. Construct a schedule with a learning rate that decreases linearly from the optimizer's initial learning rate to 0 after a warm-up period in which the learning rate increases linearly from 0 to the optimizer's initial learning rate. This should follow a warm-up period in which the learning rate increases linearly from 0 to the optimizer's initial learning rate. After fitting the model, we got accuracy **72%.**

6. Discussion:

In this section of the article, we will investigate which models performed admirably across the board in all of our tests, and we will do so by looking at their individual results. Regarding the detection of fake reviews, we came to the conclusion that using only the text review as a feature input causes the accuracy of the model to drop, and it then becomes based on the fact that the model is unable to identify a polarity mismatch between the text review and the rating. When we incorporate both the star rating and the text review into the model's input features, we see improved performance from the model. The performance of LSTM was significantly higher when compared to that of SVM with word2vec.

With SVM, LSTM, and Bert, we were able to implement two different word embedding techniques for the star rating. Following the rebalancing of the dataset, we obtained an accuracy of 65.33% using a combination of TF-IDF and SVM. Found something interesting while applying word2vec with LSTM: when the word2vec with LSTM model is built with pre-processing, we get output with poor accuracy; however, when we didn't pre-process before applying the text review to the wor2vec, we got better accuracy as compared to the pre-processed model. This was discovered while applying word2vec with LSTM. It was discovered that sometimes pre-processing can change the grammatical meaning of the review, which can indirectly change the sentiment of the review, which can affect how the star rating is predicted. When it came to BERT, it turned out to be the most accurate model, with a score of 72%.

7. Conclusion and Future work:

We propose a deep learning model with word embedding techniques that can detect fake reviews by utilising word embedding in conjunction with high-dimensional sparse data. According to the findings, the approach that makes use of deep learning is superior in terms of its ability to spot fake reviews.

By demonstrating the important aspects of reviews using pre trained embedding model as an example to explain the model, we are able to gain a deeper comprehension of the aspects of reviews.

We have discovered that pre-processing can, at times, lead to confusion within the model when applied to this dataset, as the model did not find the highest level of accuracy when it was pre-processed while attempting to forecast star ratings

.In addition, this method is only able to identify fake reviews on the online review dataset; however, it is portable enough to be applied to other datasets in order to identify fake reviews. In addition to this, it can enlarge certain aspects of the dataset in order to improve the overall performance of the model.

As previously stated, we proposed two distinct models for star prediction and fake review classification. We can combine these two models with the help of the many input function and many output function model, such like BERT pre trained model which can have a greater impact on the digital review system and make it a much safer place for businessmen and customers.

8. References:

Cambria, E., Das, D., Bandyopadhyay, S. and Feraco, A., 2017. Affective Computing and Sentiment Analysis. *A Practical Guide to Sentiment Analysis*, pp.1-10.

Centeno, R., Fresno, V. and Chaquet, J., 2018. From textual reviews to Individual Reputation Rankings: Leaving ratings aside solving MPC task. *Expert Systems with Applications*, 114, pp.1-14.

Xu, Y., Yang, Y., Wang, E., Zhuang, F. and Xiong, H., 2020. Detect Professional Malicious User with Metric Learning in Recommender Systems. *IEEE Transactions on Knowledge and Data Engineering*, pp.1-1.

Sadiq, S., Umer, M., Ullah, S., Mirjalili, S., Rupapara, V. and Nappi, M., 2021. Discrepancy detection between actual user reviews and numeric ratings of Google App store using deep learning. *Expert Systems with Applications*, 181, p.115111.

Islam, M., 2014. Numeric rating of Apps on Google Play Store by sentiment analysis on user reviews. *2014 International Conference on Electrical Engineering and Information & Communication Technology*.

Beuscart, J., Mellet, K. and Trespeuch, M., 2016. Reactivity without legitimacy? Online consumer reviews in the restaurant industry. *Journal of Cultural Economy*, 9(5), pp.458-475.

Sadiq, S., Umer, M., Ullah, S., Mirjalili, S., Rupapara, V. and Nappi, M., 2021. Discrepancy detection between actual user reviews and numeric ratings of Google App store using deep learning. *Expert Systems with Applications*, 181, p.115111.

Kumar, S. and Kumar, K., 2018. LSRC: Lexicon Star Rating system over Cloud. *IEEE*,.

Esuli, A., & Sebastiani, F. (2007). SentiWordNet: a high-coverage lexical resource for opinion mining. Evaluation, 1-26.

Sentiment Analysis and Classification Based on Textual Reviews—IEEE Conference Publication.
Ieeexplore.ieee.org. (2018). Retrieved 17 Feb 2018, from http://ieeexplore.ieee.org/document/6508366/

Statistical and Sentiment Analysis of Consumer Product Reviews—IEEE Conference Publication.Ieeexplore.ieee.org. (2018). Retrieved 19 Feb 2018, from http://ieeexplore.ieee.org/

Miner, G., Delen, D., Elder IV, J., Fast, A., Hill, T., Nisbet, R.: Practical Text Mining and Statistical Analysis for Non-Structured Text Data Applications [online]. Elderresearch.com. (2012). Available at: https://www.elderresearch.com/hubfs/Whitepaper_The_Seven_Practice_Areas_of_Text_Analytics_Chapter_2_Excerpt.pdf. Accessed 8 Jun 2018

Goldani, M., Safabakhsh, R. and Momtazi, S., 2021. Convolutional neural network with margin loss for fake news detection. *Information Processing & Management*, 58(1), p.102418.

Neisari, A., Rueda, L. and Saad, S., 2021. Spam review detection using self-organizing maps and convolutional neural networks. *Computers & Security*, 106, p.102274.

Ieeexplore.ieee.org. 2022. *Sentiment analysis and classification based on textual reviews*. [online] Available at: <https://ieeexplore.ieee.org/document/6508366/> [Accessed 12 September 2022].

Goldani, M., Safabakhsh, R. and Momtazi, S., 2021. Convolutional neural network with margin loss for fake news detection. *Information Processing & Management*, 58(1), p.102418.

Goldani, M., Safabakhsh, R. and Momtazi, S., 2021. Convolutional neural network with margin loss for fake news detection. *Information Processing & Management*, 58(1), p.102418.

Cuizon, J. and Agravante, C., 2020. Sentiment Analysis for Review Rating Prediction in a Travel Journal. *Proceedings of the 4th International Conference on Natural Language Processing and Information Retrieval*,.

Medium. 2022. *Simple Word Embedding for Natural Language Processing*. [online] Available at: <https://towardsdatascience.com/simple-word-embedding-for-natural-language-processing-5484eeb05c06> [Accessed 12 September 2022].

Appendices

Experiment 1:

For fake review detection

Fake review classification using word2vec with LSTM

```
df.head()
```

	category	rating	label	text_
0	Home_and_Kitchen_5	5.0	CG	Love this! Well made, sturdy, and very comfor...
1	Home_and_Kitchen_5	5.0	CG	love it, a great upgrade from the original. I...
2	Home_and_Kitchen_5	5.0	CG	This pillow saved my back. I love the look and...
3	Home_and_Kitchen_5	1.0	CG	Missing information on how to use it, but it i...
4	Home_and_Kitchen_5	5.0	CG	Very nice set. Good quality. We have had the s...

Figure 2 dataset

```
# Tockonizing the sentences
tokenizer = nltk.data.load('tokenizers/punkt/english.pickle')
sentences=[]
sum=0
for text_view in text:
  sents=tokenizer.tokenize(text_view.strip())
  sum+=len(sents)
  for sent in sents:
    sentences.append(sent.split())
```

Figure 3 tokenizing the text

```
[ ] from gensim.models import word2vec

[ ] # Applying the word embeddings
    w2v_model = gensim.models.Word2Vec(sentences=sentences,size=100,window=10,min_count=1)

[ ] # Total number of words
    vocab=w2v_model.wv.vocab
    print("The total number of words are : ",len(vocab))

    The total number of words are :  103821

    # Number of key-value pairs
    word_vec_dict={}
    for word in vocab:
      word_vec_dict[word]=w2v_model.wv.get_vector(word)
    print("The no of key-value pairs : ",len(word_vec_dict)) # should come equal to vocab size

    The no of key-value pairs :  103821

[ ] # To pad, we need to find the maximum lenght of any document.
    maximum = -1
    for i, text_sentence in enumerate(y):
      tokens = text_sentence.split()
      if(len(tokens) > maximum):
        maximum = len(tokens)
    print(maximum)

    373
```

Figure 4 word2vec processing

```
]  # CREATING LSTM MODEL
   deep_inputs = Input(373)
   embedding_layer = Embedding(vocab_size, 100, weights=[embed_matrix], trainable=False)(deep_inputs)
   LSTM_Layer_1 = LSTM(128)(embedding_layer)
   dense_layer_1 = Dense(3, activation='softmax')(LSTM_Layer_1)
   model = Model(inputs=deep_inputs, outputs=dense_layer_1)

   model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['acc'])

   # CREATING LSTM MODEL
   model=Sequential()
   model.add(Embedding(input_dim=vocab_size,output_dim= 100 ,input_length=maximum, weights = [embed_matrix]))
   model.add(LSTM(128))
   model.add(Dense(3, activation='softmax'))
   model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['acc'])
```

Figure 5 model building of LSTM

```
[ ]

   history = model.fit(X_train, y_train, batch_size=128, epochs=10, verbose=1, validation_data=(X_test, y_test))

   Epoch 1/10
   222/222 [==============================] - 218s 971ms/step - loss: 0.7122 - acc: 0.4999 - val_loss: 0.6942 - val_acc: 0.5000
   Epoch 2/10
   222/222 [==============================] - 210s 947ms/step - loss: 0.6966 - acc: 0.5024 - val_loss: 0.6933 - val_acc: 0.5017
   Epoch 3/10
   222/222 [==============================] - 210s 947ms/step - loss: 0.5979 - acc: 0.6333 - val_loss: 0.3596 - val_acc: 0.8575
   Epoch 4/10
   222/222 [==============================] - 213s 960ms/step - loss: 0.2436 - acc: 0.9046 - val_loss: 0.2348 - val_acc: 0.9063
   Epoch 5/10
   222/222 [==============================] - 224s 1s/step - loss: 0.1434 - acc: 0.9476 - val_loss: 0.2053 - val_acc: 0.9210
   Epoch 6/10
   222/222 [==============================] - 251s 1s/step - loss: 0.0875 - acc: 0.9704 - val_loss: 0.2140 - val_acc: 0.9177
   Epoch 7/10
   222/222 [==============================] - 210s 946ms/step - loss: 0.0636 - acc: 0.9799 - val_loss: 0.2770 - val_acc: 0.9185
   Epoch 8/10
   222/222 [==============================] - 208s 939ms/step - loss: 0.1526 - acc: 0.9533 - val_loss: 0.2602 - val_acc: 0.9059
   Epoch 9/10
   222/222 [==============================] - 208s 937ms/step - loss: 0.0742 - acc: 0.9757 - val_loss: 0.2562 - val_acc: 0.9143
   Epoch 10/10
   222/222 [==============================] - 210s 944ms/step - loss: 0.0428 - acc: 0.9875 - val_loss: 0.2905 - val_acc: 0.9120
```

Figure 6 model fit and epochs

```
plt.title('Loss')
plt.plot(history.history['loss'], label='train')
plt.plot(history.history['val_loss'], label='test')
plt.legend()
plt.show();
```

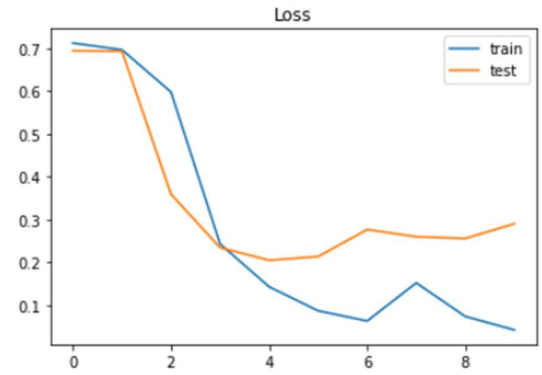


Figure 7 lose graph

```
plt.title('Accuracy')
plt.plot(history.history['acc'], label='train')
plt.plot(history.history['val_acc'], label='test')
plt.legend()
plt.show();
```

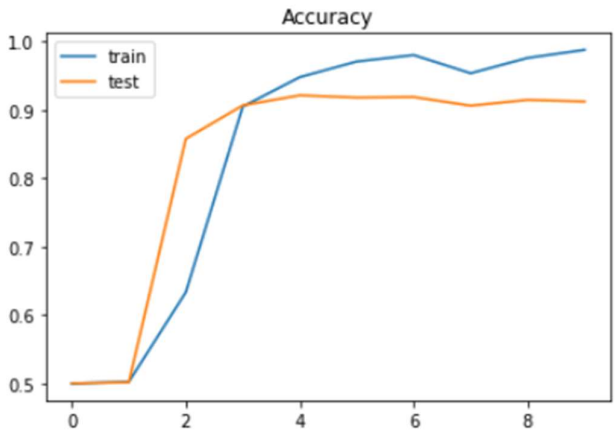


Figure 8 Accuracy of LSTM model with word2vec