

Sentiment Analysis Machine Learning & Deep Learning based approach

Salah eddine Nacer

September 15, 2020

Acknowledgment

First, I want to bow down, thanking Allah the Almighty for the good health and well being that were necessary to complete this work.

I would like to express my sincere gratitude to my advisor Prof. — for the continuous support of my Thesis work, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. —, Prof. —, and Dr. —, for their insightful comments and encouragement, but also for the hard question which incited me to widen my research from various perspectives.

I deeply thank all the teachers who encouraged and supported me during my studies.

Last but not the least, I would like to thank my family: my parents for supporting me spiritually throughout writing this thesis and my life in general.

Dedication

I dedicate this project to Allah Almighty my creator, my strong pillar, my source of inspiration, wisdom, knowledge and understanding. He has been the source of my strength throughout this program.

I also dedicate this modest work to the memory of my father, I miss him every day, My Allah have mercy on his soul. To my mother as a testimony of my gratitude for her support, efforts, sacrifices and her constant encouragement throughout my years of study and all my family, teachers and friends as they have been also a great support for me.

Thank you all.

Abstract

With the huge amount of data being generated since the emergence of technology and the appearance and growth of social networks. the need to analysis this large scale data has become a necessity for different types of businesses, as getting feedback or opinions from customers will make a hug impact in improving products.

Many researches in the fields of AI and NLP has been done, and techniques like SVM, decision trees and recurrent neural networks were implemented, all of these techniques gave different accuracy results according to how they were implemented and what type of data was used.

In this work we have tried to implement different techniques based on machine learning and deep learning specifically Recurrent neural network and Long short term memory , and comparing it also to svm technique to see the improvment acheived wth lstm, all of this was analyze English reviews provided by different users for multiple consumable products like mobiles and cinema products like movies,and eventually extract relevant sentiments with maximum accuracy.

Keywords : Sentiment Analysis,Recurrent neural networks , Natural language processing ,Deep learning, Machine learning.

Contents

Acknowledgment	iii
Dedication	iv
Abstract	v
Contents	vi
List of Figures	x
List of Tables	xii
1 Introduction	1
1.1 Key Insights on sentiment analysis	2
1.2 Problems and research challenges	2
1.3 Structure of the Thesis	3
2 Sentiment Analysis and Natural Language Processing	5
2.1 Introduction	6
2.2 Natural Language Processing	6
2.2.1 Overview on NLP	6
2.2.2 Applications of NLP	7
2.3 Sentiment Analysis and Opinion Mining	8
2.3.1 Sentiments and emotions	9
2.3.2 Opinions	9
2.3.3 Different Types of Opinions	9
2.3.3.1 Regular and Comparative Opinions	9
2.3.3.2 Explicit and Implicit Opinions	10
2.3.4 Sentiment analysis vs Opinion Mining	10

2.4	Sentiment analysis levels	11
2.4.1	Document level	11
2.4.2	Sentence level	12
2.4.3	Entity and Aspect level	12
2.5	Sentiment analysis applications	12
2.6	Sentiment analysis approaches	13
2.6.1	Machine Learning approach	13
2.6.1.1	Supervised	13
2.6.2	Lexicon-Based approach	15
2.6.2.1	Dictionary based approach	15
2.6.2.2	Corpus based approach	15
2.6.3	Hybrid approach	15
2.6.4	Summarization of different SA techniques	15
2.7	Sentiment analysis Related Work	15
2.8	Conclusion	17
3	Machine Learning and Deep Learning classification algorithms	19
3.1	Introduction	20
3.2	Machine Learning	20
3.2.1	Definition	20
3.2.2	Machine Learning Classification Algorithms	21
3.2.2.1	Support Vector Machine	21
3.2.2.2	Logistic Regression	23
3.2.2.3	Decision Tree	24
3.2.2.4	Naive Bayes	25
3.2.2.5	KNN (k-Nearest Neighbors)	27
3.3	Deep Learning	29
3.3.1	Definition	29
3.3.2	Feed-forward neural networks	30
3.3.3	Recurrent neural networks	31
3.3.3.1	Long short-term memory	33
3.4	Conclusion	33
4	System Design	35
4.1	Introduction	36

4.2	General System Architecture	36
4.3	Detailed System Architecture	37
4.3.1	Introduction	37
4.3.2	Data Collection/Gathering	37
4.3.3	Data Preprocessing and preparing	38
4.3.4	Training the Machine Learning Classifier and Building The model	42
4.3.5	Visualization of results and Implementation	43
4.4	Conclusion	43
5	System Implementation	45
5.1	Introduction	46
5.2	Work Environment and Development Tools	46
5.2.1	programming languages	46
5.2.1.1	Python	46
5.2.1.2	Javascript	46
5.2.2	Machine learning kit	47
5.2.2.1	Tensorflow	47
5.2.2.2	Keras	47
5.2.2.3	Gensim	48
5.2.2.4	NumPy	48
5.2.2.5	NLTK	48
5.2.2.6	Matplotlib	49
5.2.3	Frameworks and tools	49
5.2.3.1	Flask	49
5.2.3.2	Google Colaboratory	49
5.3	Preparing collected data	50
5.3.1	Preprocessing data:	51
5.3.2	Merging data:	51
5.3.3	Combined Dataset stats:	52
5.3.4	Splitting Dataset:	52
5.4	Sentiment analysis in SVM	53
5.4.1	Feature Extraction	53
5.4.2	Training model	53
5.4.3	Model evaluation	53
5.5	Sentiment analysis in RNN-LSTM	54

5.5.1	Vectoring data	54
5.5.2	Building and Training RNN-LSTM model	56
5.5.3	Model evaluation	58
5.6	Result Comparison	58
5.7	Implementation and deployment	59
5.7.1	Exporting models	59
5.7.2	Sentiment analysis web interface	59
5.7.2.1	User interface preview	59
5.7.2.2	User interface Detailed Explanation	60
5.7.2.3	A Simple Usage Scenario	60
5.8	Conclusion	61
6	Conclusion and Future work	63
	Bibliography	65
	Erratum	68

List of Figures

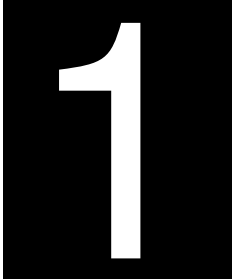
1.1	Thesis Structure.	4
2.1	Broad Classification of NLP	7
2.2	Analyzing Sentiments of an English Product Review	11
2.3	Some Applications of Sentiment Analysis	13
2.4	Sentiment Analysis approaches	14
3.1	Machine Learning Algorithms	20
3.2	Machine Learning Applications	21
3.3	General classification hyperplane representation of SVM-algorithm	22
3.4	SVM For non Linear Data	22
3.5	Svm with non Linear Data	23
3.6	Logistic function transformation example	24
3.7	decision tree representation	25
3.8	Bayes'Theorem	26
3.9	Machine Learning VS Deep Learning	30
3.10	Artificial Neural Network	31
3.11	Recurrent Neural Networks with a loop.	31
3.12	Recurrent Neural Network	32
3.13	unrolled recurrent neural network	32
4.1	The General system architecture	36
4.2	Dataset Collection	37
4.3	Data Vectorization and feature mining step.	39
4.4	Simple CBOW Model with one context word as input.	41
4.5	Simple CBOW Model with Multiple context words as input.	41
4.6	Simple Skip-Gram Model.	42
4.7	Training classifier and building ML Model process.	43

4.8	Sentiment analysis implementation process.	43
5.1	Python language logo	46
5.2	Javascript language logo	47
5.3	TensorFlow Logo	47
5.4	Keras Logo	48
5.5	Gensim logo	48
5.6	Numpy Logo	48
5.7	Flask Logo	49
5.8	Google colab Logo	50
5.9	Clean Dataset function	51
5.10	Python script to Merge data	51
5.11	Positive and Negative reviews pie chart stats	52
5.12	Splitting data script.	52
5.13	TF-IDF Feature Extractor.	53
5.14	SVM training.	53
5.15	Testing Svm accuracy.	54
5.16	Confusion matrix.	54
5.17	Word2Vec Vectorizer.	55
5.18	Word2Vec results testing.	56
5.19	Lstm network.	57
5.20	Lstm Training.	57
5.21	Model loss/accuracy evaluation.	58
5.22	Lstm Confusion matrix.	58
5.23	Exporting models.	59
5.24	Sentiment analysis web interface.	59
5.25	Sentiment analysis web interface components.	60
5.26	Negative review	61
5.27	Positive review	61

List of Tables

2.1 Sentiment classification approaches 16

4.1 Datasets collection metadata 38

Chapter 

Introduction

'If we have data, let's look at data. If all we have are opinions,
let's go with mine.'

Jim Barksdale

With the major growth of the Internet that we have seen in the 21st century, with a number of users worldwide that reached 4.54 billion in 2020, and with the rise of Web 2.0 platforms like social media with a 3.8 billion active users worldwide [24], which provided a medium or a huge network to people to connect to each other and express their opinions for different topics, also gave means to provide feedback for different types of products, in an easy and a simple manner. these mediums caused an explosive growth of subjective information (i.e., personal opinions)intended for books, products, political issues ...etc. With all this data available at hand ,it has become a necessity to analyze its contents and extract its meaningful knowledge.

1.1 Key Insights on sentiment analysis

Sentiment analysis is very important topic to keep track users in social media because it allows us to get insights of global public opinion for certain topics. Sentiment analysis has a broad set of powerful applications. Extracting useful knowledge and insights from social data is a practice that is being widely adopted by firms, companies and different organizations across the glob. The value and ability of shifting of sentiments on social media did actually show to affect and correlate shifts in the stock market.

As widely known implementation of sentiment analysis was on the political scene of the Obama administration for his presidential campaign of 2012 in which they've used sentiment analysis to gauge public opinion to get feedback for different announced policies. This gave them the ability to quickly see the sentiments behind everything from facebook posts to news and forums articles of potential voters, this also means that they had the power to make dicisions and plans and strategies for the future.

This can also be an extremely an effective part for sellers to make clear market research and customer service approach. You can view informations about what people thinks about your products, that's not all but you also can know what they think about your competitors too. Sentiment analysis can reveal a huge portion of the customer experience of your clients and/or users understand consumer attitudes quickly and reacting accordingly is something that companies like in the music industry took advantage of when they noticed that negative feedback was steadily increasing to the music used in one of their television adverts, this played as game changing card for their products otherwise they've would loose a lot.

1.2 Problems and research challenges

While the internet keeps growing , and the data keep pouring to data centers, the research on sentiment analysis is a promising and very interesting field, and that may come with challenges

and issues that we will try get into in this work and try to provide a platform for doing sentiment analysis with a high performance and accuracy to get better results.

We summarize our objectives for this work as follows.

- ✓ Study the concept and philosophy of sentiment analysis, its importance in real world applications.
- ✓ Study machine learning techniques and see how we can use them for sentiment analysis.
- ✓ Propose a platform for sentiment analysis, with machine learning support for English languages.

1.3 Structure of the Thesis

Apart from the introduction and the general conclusion which are the first and the last chapters this thesis is made up of four other chapters organized as follows

The Second chapter «Sentiment Analysis and *Natural Language Processing*»: This chapter starts by introducing the concept of sentiment analysis and its different aspects, and its importance in modern world.

The Third chapter «Machine Learning & Deep Learning for Sentiment analysis »: In this chapter we will present what is Machine Learning also Deep Learning, how they work and how we can implement several of its algorithms to do sentiment analysis.

The Fourth chapter «System Design»: Here we will create and design an architecture for our sentiment analysis system, which comprises its different aspects and functionalities.

The Fifth chapter «System implementation and results evaluation» This part consists in presenting the software environment on which the system will be implemented, as well as the details of implementation of our application. We will give a textual and graphic description of some interfaces of the system. We will also take a look of the performance and accuracy of our system based on the result we will have.

The figure [1.1] below illustrates the structure of the thesis in a simple manner:

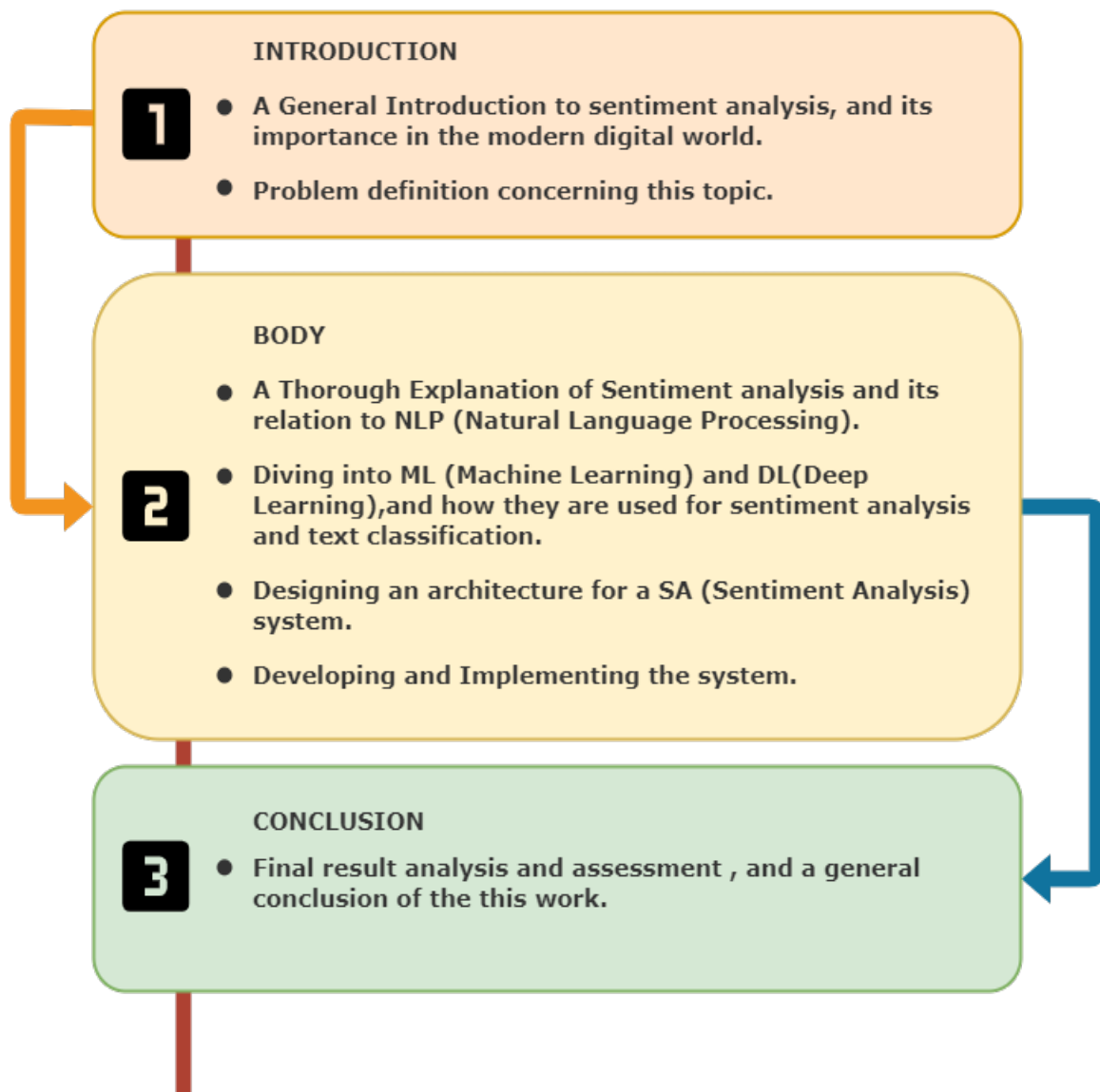


Figure 1.1: Thesis Structure.

Chapter

2

Sentiment Analysis and Natural Language Processing

“Everyone is entitled to his own opinion,
but not to his own facts.”

Daniel Patrick Moynihan

2.1 Introduction

In the past 20 years, the internet has provided us with a huge amount of data generated by people all around the globe in social media, forums etc. . . , this data which has various sizes and structures is available in many different formats including as text, images sound. . . Scientists realized the importance of such data especially textual ones, and in order to make computers understand, interpret and manipulate human's natural language. A field of AI¹ emerged called *natural language processing* or NLP for short. It achieved great results in tasks such as machine translation, email spam detection, and information extraction. And as a result the need to categorize the opinions and thoughts carried in data, process named sentiment analysis comes to role [13].

In this chapter we are going to talk about the concept of natural language processing and the philosophy of sentiment analysis, its importance in the field of AI and text classification, and its real world application.

2.2 Natural Language Processing

2.2.1 Overview on NLP

NLP² is usually a multidisciplinary science, as it can be considered very related to linguistics theory it can also relate to fields like philosophy, psychology, cognitive science but within computer science, NLP falls into formal language theory, compiler techniques and mainly into human-computer interaction and AI, specifically machine learning, NLP in CS is concerned with interaction process between machines or computers and humans, through making them able to make efficient, correct and easy communication by comprehending or understanding the human natural language, this process is done automatically by processing language input and produce output that makes sense. [17]

What can define or characterize language as a set of rules or a set of symbols. A symbol is combined and used for passing on data or broadcasting the data. Natural Language Processing essentially can be classified into two main parts i.e. *Natural Language Understanding* and *Natural Language Generation* that is used as the assignment or task to understand and produce the content or in other words generate the content. The next figure ([2.1]) shows the Broad Classification of NLP.[2]

¹Artificial Intelligence

²Natural Language Processing

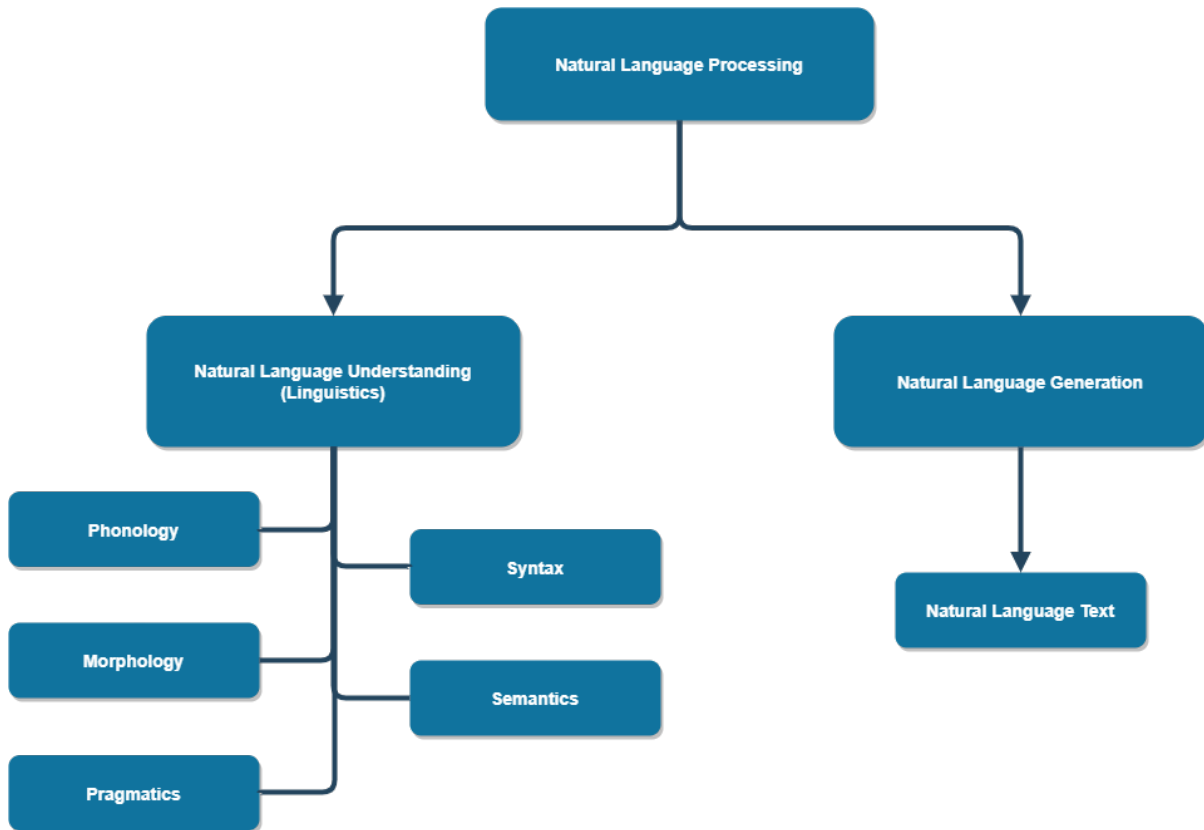


Figure 2.1: Broad Classification of NLP

2.2.2 Applications of NLP

We can apply NLP in many and various sorts of fields and areas, areas like Email spam detection, Speech Recognition, Machine Translation, Information extraction, Text Summarization etc. Here we mention some examples:

- Email Spam Detection:** Spam email or also refereed as Junk mail is an email messages sent to people by bulk were this email is unwanted and unneeded or has no value, usually this email is used for phishing or promoting for commercial reasons. Detecting spam mail requires applying text classification and in recent time techniques specifically in machine learning were used to process and analyze textual data to to detect and identify spam content (anti-spam filtering) like Naïve Bayes algorithm, Rule Learning, Memory based Learning, Support vector machines, Decision Trees, Maximum Entropy Model

We get better and accurate results when we apply machine learning techniques with the previously mentioned algorithms, than making our classifiers learn by simple programmed rules, a preffered approach by experiments is the naïve bayes as it performs well despite of its simplicity[2]

- **Text Summarization:** As the digital world is overloaded with information, and our capacity to understand this huge amounts of data , has reached a critical levels , the need to summarize data and keep its meaning intact has become a necessity The "Text Summarization" sentence means doing a technique or a method for keeping shortening long portions of textual data, this intents to create a coherent and fluent summary that have only main and important points mentioned and clarified. Tow types of main techniques used for summarization first is *Extraction-based summarization* this method pulls out key-phrases from input documents and combines them to make a summary. The extraction is made through a specified metrics, in which they don't affect the original text, and we can notice that the final result may have some grammatical inconsistencies in some cases. The second method is: *Abstraction-based summarization* this technique calls for shortening and paraphrasing parts of the input document, when this is applied with deep learning it beats the grammatical inconsistencies which happens in the first technique, in other terms this technique creates new phrases that highlights the most useful informations of the source input document, as humans actually do in summarizing texts or articles.[2] Therefor it performs better than extractions thought its more difficult to do.
- **Language Translation:** As The world became more connected, making the data available to all people has turned to a challenge, one of those major challenges is the language barrier, with thousands of human spoken languages in the world, wich every language has its own structure, grammar and specifics. Machine Translation is generally translating phrases from one language to another with the help of a statistical engine like Google Translate. The challenge with machine translation technologies is not directly translating words but keeping the meaning of sentences intact along with grammar and tenses, As for Google, in 2016, announced a new machine translation system based on Artificial neural networks and Deep learning . In recent years, various methods have been proposed to automatically evaluate machine translation quality by comparing hypothesis translations with reference translations.[13] Examples of such methods are word error rate, position-independent word error rate, generation string accuracy , multi-reference word error rate , BLEU score , NIST score

2.3 Sentiment Analysis and Opinion Mining

Sentiment analysis mainly studies opinions that express or imply positive or negative or sometimes neutral sentiment. We use the term opinion as a broad concept that covers sentiment, evaluation, appraisal, or attitude, and its associated information such as opinion target and the

person who holds the opinion, and use the term sentiment to mean only the underlying positive or negative feeling implied by opinion. Due to the need to analyze a large volume of opinions.[27]

2.3.1 Sentiments and emotions

Sentiments and emotions may seem a very similar concepts, but actually they have different meaning. Emotion is a conscious mental reaction (such as anger or fear) subjectively experienced as strong feeling usually directed toward a specific object and typically accompanied by physiological and behavioral changes in the body, it also defined as a state of feeling or as the effective aspect of consciousness (feeling) [5] Where Sentiments are defined as an attitude, thought, or judgment prompted by feeling, or a specific view or notion (opinion), refined feeling, delicate sensibility especially as expressed in a work of art[8]

2.3.2 Opinions

An opinion on the other hand could be defined as follows: A belief that a person has formed about a topic or issue.[6] , a view, judgment, or appraisal formed in the mind about a particular matter, or a belief stronger than impression and less strong than positive knowledge [7] Opinion could be found or made in product reviews, social media, real life interactions, they can be viewed as positive or negative or maybe neutral here are some examples:

I would like to know your opinions on the country Algeria ?

In my opinion, Algeria is a great country.

2.3.3 Different Types of Opinions

So far we have discussed the *regular opinion* type. Another type of opinion is named *comparative opinion* [10]. Actually we can also classify opinions based on how they are expressed in text, explicit opinion and implicit (or implied) opinion.

2.3.3.1 Regular and Comparative Opinions

- **Regular opinion:** in simple literature a regular opinion is just referred to as an opinion which has tow main sub-types

1-*Direct opinion:* This type of opinions is referring to an opinion directly expressed or given for a certain entity or sample of an entity in some aspect e.g., “The blue ocean is so beautiful.”

2-*Indirect opinion:* This means an opinion which expressed indirectly on an aspect or entity based on its effects on some other entities. This type is mainly found in domains such as the medical domain. As an example, the sentence “After getting injected, my joints felt

worse than before” this latter sentences show the undesirable and bad effects of the drug on “my joints”, thus resulting in an indirect negative opinion or sentiment to the drug. In the case, the aspect is the effect on joints and the entity is the drug. Researchers in the field mostly focus on Direct opinions. As they are a lot more simpler and easy to handle ,Indirect opinions on the other hand are often harder to deal with. For example, in the drug domain, one needs to know whether some desirable and undesirable state is before or after using the drug. For example, the sentence “Since my joints were painful, my doctor put me on this drug” does not express any sentiment or opinion on the drug because “painful joints” (which is negative) happened before using the drug. [25]

- **Comparative opinion:** A comparative opinion expresses a relation of similarities or differences between two or more entities and/or a preference of the opinion holder based on some shared aspects of the entities. For example, the sentences, “Fanta tastes better than Hamoud Boualem” and “Hamoud Boualem tastes the best” express two comparative opinions. A comparative opinion is usually expressed using the comparative or superlative form of an adjective or adverb, although not always (e.g., prefer).[25]

2.3.3.2 Explicit and Implicit Opinions


- **Explicit opinion:** An explicit opinion is a subjective statement that gives a regular or comparative opinion, e.g., “Hamoud Boualem tastes great,” and “Coca tastes better than Hamoud Boualem.”[25]
- **Implicit (or implied) opinion:** An implicit opinion is an objective statement that implies a regular or comparative opinion. Such an objective statement usually expresses a desirable or undesirable fact, e.g., “I bought the mattress a week ago, and a valley has formed,” and “The battery life of Nokia phones is longer than Samsung phones.” Explicit opinions are easier to detect and to classify than implicit opinions. Much of the current research has focused on explicit opinions. Relatively less work has been done on implicit opinions [25].

In a slightly different direction, Some researchers studied the influence of syntactic choices on perceptions of implicit sentiment. For example, for the same story, different headlines can imply different sentiments.

2.3.4 Sentiment analysis vs Opinion Mining

Sentiment analysis or opinion mining or emotion AI, all of these terms refers to the same science or field AI,that uses natural language processing, text analysis, computational linguistics,

and biometrics to systematically identify, extract, quantify, and study affective states and subjective information. Sentiments are actually emotions of users. It can be good, excellent, bad or neutral. The analysis of such emotions of users is sentiment analysis. Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine [10].in figure [2.2] we see an Example of sentiment analysis of a phone review.



Apple iPhone XR, 64GB, Red - Fully Unlocked (Renewed)
 by Amazon Renewed
 ★★★★★ 2,595 ratings | 1000+ answered questions

Price: **\$424.99** + \$169.52 Shipping & Import Fees Deposit to Algeria [Details](#)

 Raymond B. Thompson
 ★☆☆☆☆ **It sucks**
 May 20, 2019
 Style: Fully Unlocked | Color: Black | Size: 64GB | **Verified Purchase** 
 World's worst thing ever never get on this its a scam

 chris crawford
 ★★★★★ **theyre very good looking and not glitchy**
 April 3, 2019
 Style: Fully Unlocked | Color: Yellow | Size: 128GB 
 i absolutly LOVED this iPhone XR it was wonderful!!!!!!!!!!!!

Figure 2.2: Analyzing Sentiments of an English Product Review

2.4 Sentiment analysis levels

Many researchers in the field have studied Sentiment analysis mainly in a set of three levels, *Document level sentiment classification*, *Sentence level sentiment classification*, *Aspect-based sentiment classification*.

2.4.1 Document level

The document level sentiment analysis classifies the entire document opinion into different sentiment, for a product or service. This level classifies opinion document into a positive, negative or neutral sentiment. For example, given a product review our system detects the overall positive or negative opinion about the product, We note that in this case we consider the whole document as a single basic information unit and this document is contains opinions on a specific certain entity, this is known as the document-level sentiment analysis task, and this is not allowed for documents which evaluate multiple entities. [1]

2.4.2 Sentence level

The sentence level sentiment analysis determines whether each sentence expresses a positive, negative or neutral opinion, for a product or service. This type is used for reviews and comments that contain one sentence and written by the user and it only covers individual sentences in a document. [1]

2.4.3 Entity and Aspect level

Aspect level is the opinion mining and summarization based on feature. The classification concerns by identifying and extracting product features from the source data. This type is used when we need sentiments about desired aspect/feature in a review. [1]

2.5 Sentiment analysis applications

One of the most important needs of businesses and organizations in the real world is to find and analyze consumer or public opinions about their products and services . Knowing the opinions of existing users regarding a specific product is also interesting for individual consumers. Sentiment analysis paves the way to several and interesting applications, in almost every possible domain. Other projects have as a long-term goal the clarification of politicians' positions, such as what public figures support or oppose, to enhance the quality of information that voters have access to. figure [2.3] shows some of real world applications of sentiment analysis.[14]



Figure 2.3: Some Applications of Sentiment Analysis

2.6 Sentiment analysis approaches

Different techniques or approaches are used for sentiment analysis, the following figure [2.4] shows these techniques.

2.6.1 Machine Learning approach

This is an automatic classification technique. Classification is performed using text features. Features are extracted from text. It is of two types- supervised and unsupervised[19].

2.6.1.1 Supervised

Supervised learning is our main and successful plus effective key in classification and has been highly adopted and investigated for opinion mining with great and important outcome. In this case our system is trained using labeled training examples. Each class of them represents different features and has a label associated with it. When a input is provided as a textual form, it is preprocessed and then a set of specific features are extracted and compared and

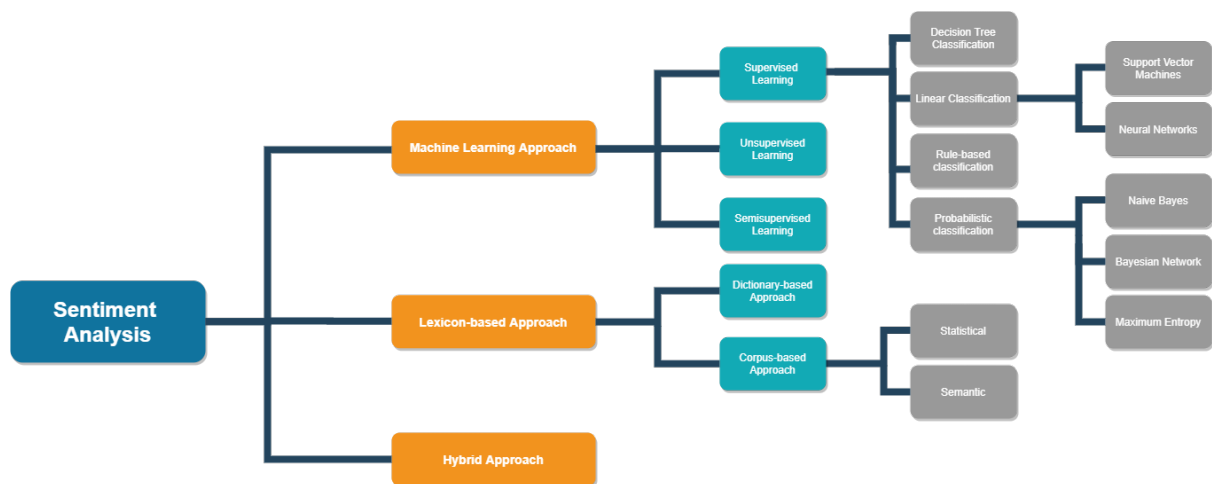


Figure 2.4: Sentiment Analysis approaches

labeled with a certain class with the most matching accuracy.[18] We mention some supervised classification techniques as following:

- Probabilistic classifier:** This classifier is able to foresee a probability function over a set of classes for a given input data. It does not give only the most likely classes but a probability function over all classes. These are some probabilistic classifiers used: Naïve Bayes classifier (NB), Bayesian Network (BN), Maximum Entropy classifier (ME)[18]
- Linear classifier:** It can perform classification based on the linear combinations value of the characteristics. Let $W = w_1, w_2, w_3, \dots$ is word frequency for example, and vector $C = c_1, c_2, c_3, \dots$ is linear coefficient vector and S is a scalar then output of linear predictor will be $LP = W.C + S$. This predictor is called hyperplane which separates two classes. \times SVM: Support vector machine is a supervised learning model which is used for classification, The main objective of (SVM) is to find out linear separators which suited for separating different classes without prior assumptions.[18]
- Decision tree classifier:** In this classification technique, a condition is applied to divide the data. Data which satisfy the condition is placed in one class and rest data in other class. It is based on a recursive approach or procedure. There are a number of splits: single attribute split which searches for particular word presence or absence to perform classification, similarity based multi attribute split which matches the given document words with predefined words to perform classification and Discriminate based multi-attribute split use discriminates to perform split.[26]
- Rule based classifier:** This classifier makes use of certain rules as IF, THEN. It can be written as IF condition THEN decision. The rules can be generated during training phase

depending on our requirements.[26]

2.6.2 Lexicon-Based approach

Lexicon-Based or Subjective lexicons approach are collection of words where each word has a score indicating the positive, negative, neutral and objective nature of text. In this approach, for a given piece of text, aggregation of scores of subjective words is performed i.e. positive, negative, neutral and objective word scores are summed up separately. In the end there are four scores. Highest score gives the overall polarity of the text.[26]

2.6.2.1 Dictionary based approach

In this approach a set of opinion words are manually collected and a seed list is prepared. Then we search for dictionaries and thesaurus to find synonyms and antonyms of text. The newly found synonyms are added to the seed list. This process continues until no new words are found. Disadvantage: difficulty in finding context or domain oriented opinion words.[26]

2.6.2.2 Corpus based approach

Corpus is collection of writings, often on a specific topic. In this approach, seed list is prepared and is expanded with the help of corpus text. Thus it solves the problem of limited domain oriented text. It can be done in two ways.[26]

- **Statistical approach:** This approach is used to find co-occurrence words in the corpus. Idea is that if the word appears mostly in positive text, then its polarity is positive. If it mostly occur in negative text, then its polarity is negative.[26]
- **Semantic approach:** This approach calculates sentiment values by using the principal of similarity between words. Wordnet can be used for this purpose. Synonyms and antonyms of given word can be found using this and sentiment value can be calculated.[26]

2.6.3 Hybrid approach

It is a combination of machine learning and lexicon-based approach and is frequently use with sentimental lexicon express major role in different important method.[26]

2.6.4 Summarization of different SA techniques

Table 2.1 show a summarization of the techniques, and their advantages/disadvantages.

2.7 Sentiment analysis Related Work

Many researchers worked on Sentiment analysis filed of study, building tools and systems makes it an important sources in decision making for all sorts of businesses. although it has

SENTIMENT CLASSIFICATION APPROACHES		FEATURES/ TECNIQUES	ADVANTAGES AND LIMITATIONS
Machine Learning	Bayesian Networks Naive Bayes Classification Maximum Entropy Neural Networks Support Vector Machine	Term presence and frequency Part of speech information Negations Opinion words and phrases	ADVANTAGES the ability to adapt and create trained models for specific purposes and contexts LIMITATIONS The low applicability to new data because it is necessary the availability of labeled data that could be costly or even prohibitive
Lexicon Based	Dictionary based approach Novel Machine Learning Approach Corpus based approach Ensemble Approaches	Manual construction, Corpus-based Dictionary-based	ADVANTAGES wider term coverage LIMITATIONS finite number of words in the lexicons and the assignation of a fixed sentiment orientation and score to words
Hybrid	Machine learning Lexicon based	Sentiment lexicon constructed using public resources for initial sentiment detection Sentiment words as features in machine learning method	ADVANTAGES lexicon/learning symbiosis, the detection and measurement of sentiment at the concept level and the lesser sensitivity to changes in topic domain LIMITATIONS noisy reviews

Table 2.1: Sentiment classification approaches

some challenges and needs more research and more attention. In the following, we will discuss some related work done with sentiment analysis and its challenges.

Singla et al[21] have done Sentiment Analysis work for Customer Product Reviews Using Machine Learning, after collecting and preprocessing data from e-commerce website Amazon.com in a textual csv form, they have used three classification models to classify reviews, these classifiers are Naïve Bayes, SVM and Decision Tree and got the best result out of SVM model with an accuracy of 81.75%.

Hu et al [12] They have proposed a deep neural network based framework to perform Sentiment Analysis with a collection of different sources of data, review data such as movies and hotel reviews collected from amazon and imdb in-order to evaluate their algorithm, and done a comparison to SVM and got a great accuracy with the increase of size of data

Souza et al [22] these group of researchers have done sentiment analysis following a deep learning approach applied to Hotel's Reviews What they have done is collecting a corpus of 69,075 reviews about hotels located at Rio de Janeiro city, after performing preprocessing and data cleaning, they made dense vectors numerical representation to the words to act as an input to our DL model by using GloVe technique as the chosen it, also they chosen to classify according to three classes positive, negative and neutral eventually they used keras framework to apply a convolutional network as their model. finally after applying tests on five different sub-corpses they got varying accuracy results for each test.

Dholpuria et al [9] another group or researchers done Sentiment analysis through deep

learning approach for a movie review they also applied many different supervised learning models like classifiers like KNN, ensemble methods, and logistic regression, and also proposed a CNN approached classifier to get the better accuracy, in which the latter did actually give higher results reaching 99% of accuracy.

2.8 Conclusion

In this chapter we have seen a lot of sections on the topic and philosophy of SA, we have previewed its different aspects and applications, and the different techniques used to create systems for analyzing sentiments. we also seen some previous of work done by researchers in the field and the results they got. Sentiment analysis is a good and important field of AI for research due it's great impact for our current digital world.

Chapter

3

Machine Learning and Deep Learning classification algorithms

“Success in creating AI would be the biggest event in human history.
Unfortunately, it might also be the last, unless we learn how to avoid the risks.”

Stephen Hawking

3.1 Introduction

The work on text classification and sentiment analysis in NLP field has evolved rapidly in the past decade, due to huge growth in data and its urgent importance for businesses, and with the emergence of machine learning techniques, they have proven to be the better solution for creating more accurate and robust systems for providing better results.

Machine learning and Deep Learning are currently the major players in the field of sentiment analysis and NLP in general, due to their impressive results compared to old techniques used in the field.

In this chapter we are going to show case some popular ML¹ algorithms that are used in the field, furthermore we will dive into deep learning and explain their concepts and how they work and how we need to use them in our work.

3.2 Machine Learning

3.2.1 Definition

Machine learning is defined by (Arthur Samuel, 1959) as *the field of study that gives computers the ability to learn without being explicitly programmed*. from this definition we can see that we can perfectly apply it to the text classification problems. The following Figure [3.1] shows an overview of classes of some machine learning algorithms in the field.

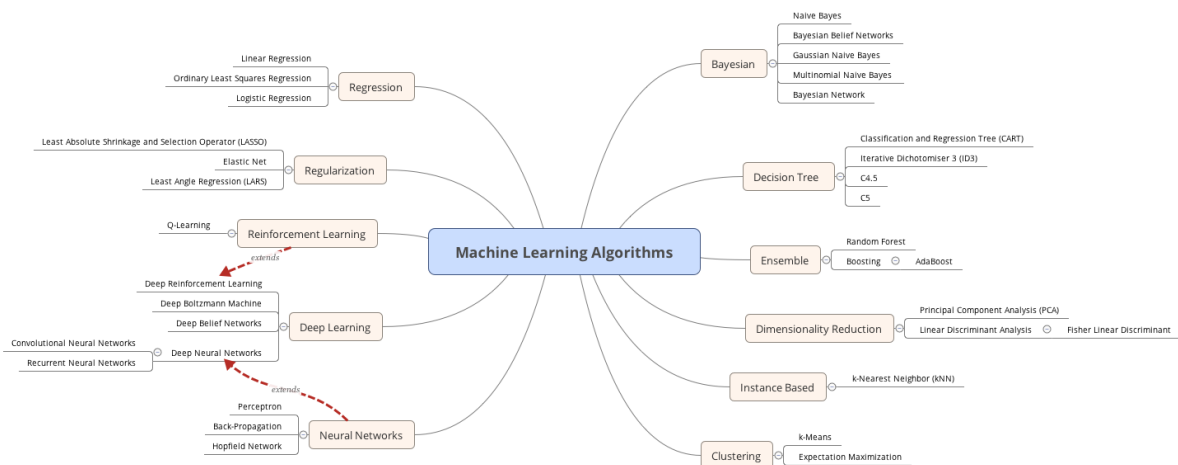


Figure 3.1: Machine Learning Algorithms

¹Machine Learning

3.2.2 Machine Learning Classification Algorithms

In machine learning and statistics, classification is a supervised learning approach in which the computer program learns to categorize data into distinct class based on some label from the input and then uses this learning to classify new unseen observations. This data set may simply be bi-class or binary classifier (like identifying whether the tweet is positive or negative or that the mail is spam or non-spam) or it may be multi-class[15]. like for example classifications on types of crops or types of music genre eg:(POP or Country music or rock). Some practical examples of classification problems are: speech recognition, handwriting recognition, bio metric identification, document classification Sentiment analysis etc. some of these applications are shown in Figure [3.2].

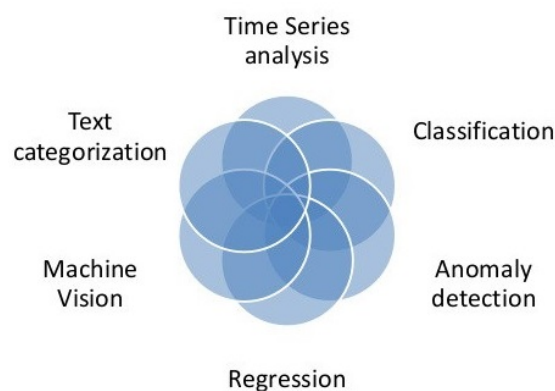


Figure 3.2: Machine Learning Applications

3.2.2.1 Support Vector Machine

Support Vector Machines shortly known as SVM² is a supervised machine learning classification algorithm ,which is one of the most used ML algorithms in image and text classification.

It transforms the data points into a higher dimensional space so that the data points can be separated linearly,it attempts to split the feature space into optimal class segments SVM builds model by finding a hyperplane that best separates the points. Its main objective is to reduce the sum of the distances of all the data points to that hyperplane. Functions called kernels, decides the hyperplanes to be chosen. If the data can be linearly separated by hyperplane then linear kernel is used. For non-linear data, RBF³ kernel is used. The division of the feature space is referred to as training the machine. A trained SVM can then be used for classification of new examples by assigning them a class based on which segment of the feature space they are located in.[16]

²Support Vector Machine

³Radial Basis Function

Basically SVM is an algorithm for classifying binary problems, of course when we deal with linearly separable classes, Figure [3.3] show a hyperplane created by the svm to divide the feature space into class segments, the hyperplane will have the largest possible margin between the two classes. This margin *maximization* is the essential concept of SVMs.[16]

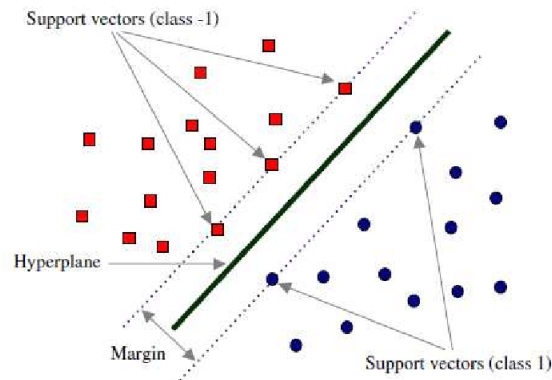


Figure 3.3: General classification hyperplane representation of SVM-algorithm

The closest data points of both classes, parallel to the vector defining the hyperplane, constitute the support vectors. These give the algorithm its name.

We can also apply the algorithm to problems where the examples are not linearly separable. In Figure [3.4] we can see clearly the data is linearly unseparable or there's not a linear decision boundary (a single straight line that separates both tags).[16]

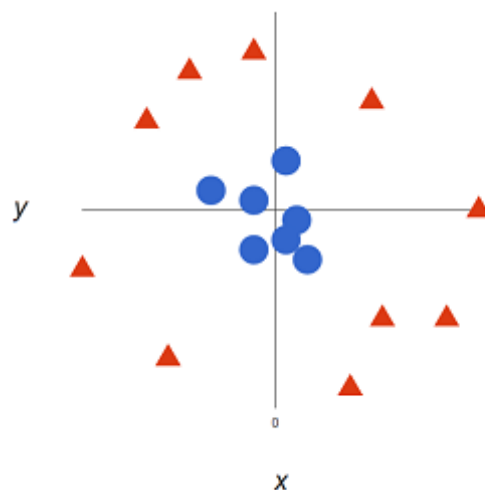


Figure 3.4: SVM For non Linear Data

To solve this issue we can do a mapping of our space to a higher dimension, so by adding a third dimension. Up until now we had two dimensions: x and y . We create a new z dimension. Figure [3.5] shows the transformation.

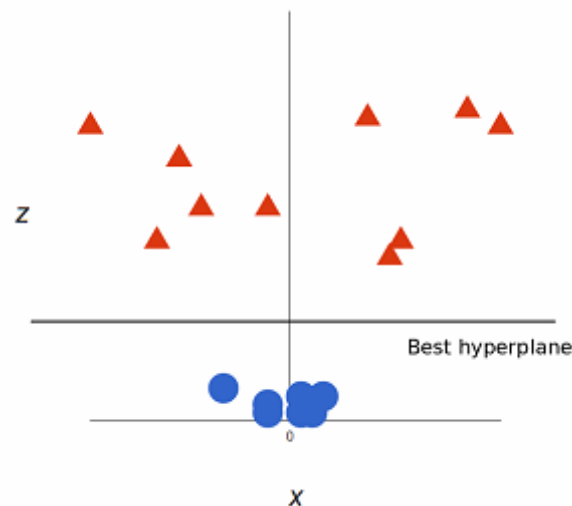


Figure 3.5: Svm with non Linear Data

Note that since we are in three dimensions now, the hyperplane is a plane parallel to the x axis at a certain z (let's say $z = 1$).

In the previous example we added a new dimension to separate our data but that transformation can get pretty computationally expensive. The *kernel trick*, which allows us to sidestep a lot of expensive calculations. Normally, the kernel is linear, and we get a linear classifier. However, by using a non-linear kernel (like above) we can get a nonlinear classifier without transforming the data at all: we only change the dot product to that of the space that we want and SVM will happily chug along. The kernel trick as we see isn't actually part of SVM. It can be used with other linear classifiers such as logistic regression. Finding the decision boundary is the thing that a support vector machine only tries to take care of. [16]

3.2.2.2 Logistic Regression

This technique in machine learning which is called logistic regression was borrowed from the field of statistics, as it is very suitable for technique for binary classification or bi-class problems (problems that have 2 classes of values for classification).

The name for logistic regression was brought from a function in mathematics and statistics called the logistic function also called the sigmoid function, this logistic function was invented by statisticians for the purpose of describing the properties of growth of populations in ecology, the fast increase and reaching the maximum carrying capacity of the environment, Logistic function is an S-shaped curve which is capable of mapping any real-valued number into a value between 0 and 1, and not touching those limits. [4]

$$\frac{1}{1 + e^{-value}}$$

Where e is representing the natural logarithm, or Eulers number (EXP()) and *value* is the transformable numerical value, we see bellow a plot 3.6 that transforms numbers between -8 and 8 into a range of 0 to 1 using the sigmoid or logistic function

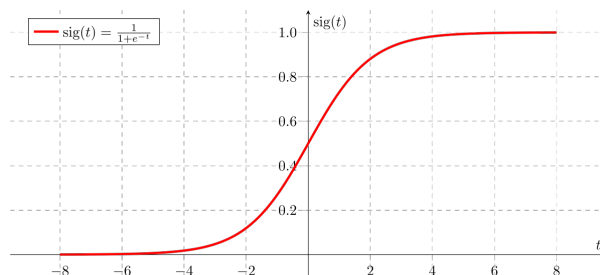


Figure 3.6: Logistic function transformation example

Despite the name Logistic regression it is considered as a classification algorithm, we use it when we have dependent variable the are binary, this means they can be either one of tow categories or values for example yes or no , true or false, one or zero. It combines weighted input features in a linear fashion which applied to the sigmoid function. This sigmoid function is main player of logistic regression and it can map value into the range of 0 to 1. **Representing Hypothesis** When representing or using the linear regression we use the following formula:

$$h_{\Theta}(x) = \beta_0 + \beta_1 x$$

In other hand when using logistic regression the previous formula will be modified a bit as the following:

$$\sigma(Z) = \sigma(\beta_0 + \beta_1 X)$$

Finally we get the end hypotheses for logistic regression as follows:

$$h_{\theta}(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

3.2.2.3 Decision Tree

Another machine learning algorithm for classification is decision tree which is mainly applied as a construction of tree of possibilities where branches of this tree represents decisions and its leaves represents labels the presents classes or categories for classification. Creating homogeneous samples of same type in each branch is the purpose of a decision tree , It does it by splitting samples of data according to specific attributes that increase homogeneity in branches. Such attributes builds the decision nodes along which samples are separated, This process will move-on until all of the samples are correctly predicted as they represented by the leaves of our tree.[4]

For getting better knowledge of how decision tree works, Let have a look at an example that demonstrates its capability. In our example will use the titanic sink scenario and we will be using its dataset to predict whether a passenger will survive or not and represent our tree based on that, our model it using three features from the dataset which are sex, age, sibsp(spouses or child number).

A top down form is the representation of a decision tree, where the root is at the top ,in the figure 3.7 the condition/internal node is represented by bold black text based on which the tree splits into branches/ edges, and the red and green text for nodes that doesn't split anymore is the decision/leaf, in this case, whether the passenger died or survived.[23]

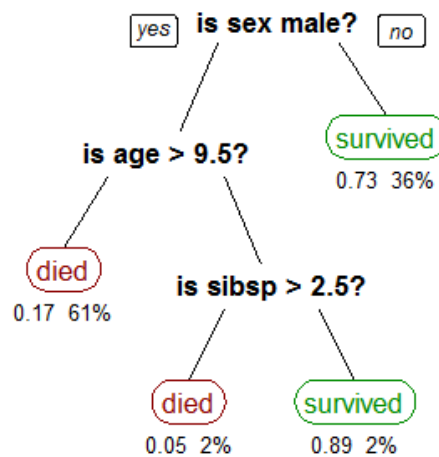


Figure 3.7: decision tree representation

the tree will be bigger than this as the dataset is big and contains a lot more features,so whats happening in the background is that the tree will grow as as deciding on what kind of features to choose and what are the conditions to use for splitting.

Decision trees are also referred to as CART⁴, and its important to mension some of its advantages and disadvantages, first as an advantage it is very simple to understand, visualize and interpret, it has the ability to hand both numerical and categorical data, it also requires not much effort from the user to prepare the data, as disadvantages we mention that it can get over-complex and don't generalize as this also called *overfitting*. [23]

3.2.2.4 Naive Bayes

Bayes Theorem

⁴Classification and Regression Trees

We regularly In ML keen on choosing the best theory or hypothesis (c) given input or data (x). In a classification issue, our speculation (c) might be the class to allocate for another data instance(x). Perhaps the simplest methods for choosing the most plausible speculation given the information that we have that we can use as our earlier information about the issue. Bayes' Theorem gives away that we can figure the likelihood of a theory given our earlier information.[3] Bayes' Theorem is expressed in Figure[3.8]: Where:

The diagram shows the equation $P(c|x) = \frac{P(x|c)P(c)}{P(x)}$. Arrows point from the terms to their labels: $P(x|c)$ is labeled 'Likelihood', $P(c)$ is labeled 'Class Prior Probability', $P(c|x)$ is labeled 'Posterior Probability', and $P(x)$ is labeled 'Predictor Prior Probability'.

$$P(c | X) = P(x_1 | c) \times P(x_2 | c) \times \dots \times P(x_n | c) \times P(c)$$

Figure 3.8: Bayes' Theorem

- $P(c|x)$ is the probability of hypothesis (c) given the data (x). This is called the posterior probability.
- $P(x|c)$ is the probability of data (x) given that the hypothesis (c) was true.
- $P(c)$ is the probability of hypothesis (c) being true (regardless of the data). This is called the prior probability of (c).
- $P(x)$ is the probability of the data (regardless of the hypothesis).

From the prior probability p(h) with P(x) and P(x|h) we now are interested in calculating the posterior probability of P(c|x), you can select the hypothesis with the highest probability after calculating the posterior probability for a number of different hypotheses, This is the maximum probable hypothesis and may formally be called the maximum a posterior i(MAP) hypothesis. and we can write it as:

$$MAP(c) = \max(P(c|x))$$

$$MAP(c) = \max(P(x|c) \times P(c)P(x))$$

$$MAP(c) = \max(P(x|c) \times P(c))$$

The P(x) represents the normalizing term that allows us to calculate the probability. when we are interested in the most probable hypothesis we can drop it. In classification if we got an

even number of instances in each class in our training data, then the probability of each class (e.g. $P(h)$) will be the same value for each class (e.g. 0.5 for a 50-50 split). Again, this would be a constant term in our equation and we could drop it so that we end up with: [4]

$$MAP(c) = \max(P(x|c))$$

Naive Bayes Classifier

Naive Bayes is considered a classification algorithm intended for binary (bi-class) and multi-class classification problems. When describing our input values as binary or categorical it makes it easiest to understand, the name naive Bayes or sometimes called idiot bayes comes from making the calculations of probabilities for each hypothesis simplified to turn their calculation tractable, this is better than trying to calculate the values of attribute value $P(x_1, x_2, x_3|c)$, since they are considered to be conditionally independent and calculated as

$$P(x_1|c) * P(x_2|c)$$

and so on, given the target value. [4]

Naive Bayes representation is a set of probabilities. and then such probabilities are saved in a file for a learned naive Bayes model. and This includes:

- Class Probabilities: The probabilities of each class in the training dataset.
- Conditional Probabilities: The conditional probabilities of each input value given each class value.

A naive Bayes model is able to learn fast and efficient from our learning data. the training process is rapid because the set of probability of each class with the different input (x) values need to be calculated. And with optimization procedures its not needed to fit any coefficients. When ever we need predictions in real time Naive Bayes classifier is suitable solution for this such as fraud detection on money transactions, it also performs well in spam detection and sentiment analysis. [4]

3.2.2.5 KNN (k-Nearest Neighbors)

The model representation for KNN is the entire training dataset. It is as simple as that. KNN has no model other than storing the entire dataset, so there is no learning required. Efficient implementations can store the data using complex data structures like k-d trees to make look-up and matching of new patterns during prediction efficient. Because the entire training dataset is stored, you may want to think carefully about the consistency of your training data. It might be

a good idea to curate it, update it often as new data becomes available and remove erroneous and outlier data.

KNN makes predictions using the training dataset directly. Predictions are made for a new data point by searching through the entire training set for the k most similar instances (the neighbors) and summarizing the output variable for those k instances. For regression this might be the mean output variable, in classification this might be the mode (or most common) class value.[3]

To determine which of the k instances in the training dataset are most similar to a new input a distance measure is used. For real-valued input variables, the most popular distance

measure is Euclidean distance. Euclidean distance is calculated as the square root of the sum of the squared differences between a point a and point b across all input attributes i .

$$EuclideanDistance(a,b) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2}$$

Other popular distance measures include:

- Hamming Distance: Calculate the distance between binary vectors.
- Manhattan Distance: Calculate the distance between real vectors using the sum of their absolute difference. Also called City Block Distance.
- Minkowski Distance: Generalization of Euclidean and Manhattan distance.

There are many other distance measures that can be used, such as Tanimoto, Jaccard, Mahalanobis and cosine distance. You can choose the best distance metric based on the properties of your data. If you are unsure, you can experiment with different distance metrics and different values of k together and see which mix results in the most accurate models. Euclidean is a good distance measure to use if the input variables are similar in type (e.g. all measured widths and heights). Manhattan distance is a good measure to use if the input variables are not similar in type (such as age, gender, height, etc.).

The value for k can be found by algorithm tuning. It is a good idea to try many different values for k (e.g. values from 1 to 21) and see what works best for your problem. The computational complexity of KNN increases with the size of the training dataset. For very large training sets, KNN can be made stochastic by taking a sample from the training dataset from which to calculate the k -most similar instances. KNN has been around for a long time and has been very well studied. As such, different disciplines have different names for it, for example:

- **Instance-Based Learning:** The raw training instances are used to make predictions. As such KNN is often referred to as instance-based learning or a case-based learning (where each training instance is a case from the problem domain).
- **Lazy Learning:** No learning of the model is required and all of the work happens at the time a prediction is requested. As such, KNN is often referred to as a lazy learning algorithm.
- **Non-parametric:** KNN makes no assumptions about the functional form of the problem being solved. As such KNN is referred to as a non-parametric machine learning algorithm.

KNN can be used for regression and classification problems.

KNN for Regression When KNN is used for regression problems the prediction is based on the mean or the median of the k-most similar instances.

KNN for Classification When KNN is used for classification, the output can be calculated as the class with the highest frequency from the k-most similar instances. Each instance in essence votes for their class and the class with the most votes is taken as the prediction. Class probabilities can be calculated as the normalized frequency of samples that belong to each class in the set of k most similar instances for a new data instance. For example, in a binary classification problem (class is 0 or 1):

$$p(\text{class} = 0) = \frac{\text{count}(\text{class} = 0)}{\text{count}(\text{class} = 0) + \text{count}(\text{class} = 1)}$$

If you are using k and you have an even number of classes (e.g. 2) it is a good idea to choose a k value with an odd number to avoid a tie. And the inverse, use an even number for k when you have an odd number of classes. Ties can be broken consistently by expanding k by 1 and looking at the class of the next most similar instance in the training dataset.[3]

3.3 Deep Learning

3.3.1 Definition

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the human brain called artificial neural networks. Scientists don't actually have the full picture on how the human brain works, but based on the prevailing model, it could be very helpful in machine learning.[20]

Actually, Deep Learning can cover both Supervised Learning and Unsupervised Learning. That's because with Deep Learning you can do Regression, Classification, and Clustering. What's the difference then? Deep Learning really shines if you're analyzing massive datasets

especially those that contain text, image, audio, and even video data, also feature extraction is done automatically in deep learning see Figure[3.9]

It's possible to use traditional machine learning techniques for massive and complex datasets. But for large scale analysis and projects, Deep Learning is still the most popular approach whether in academics or business.

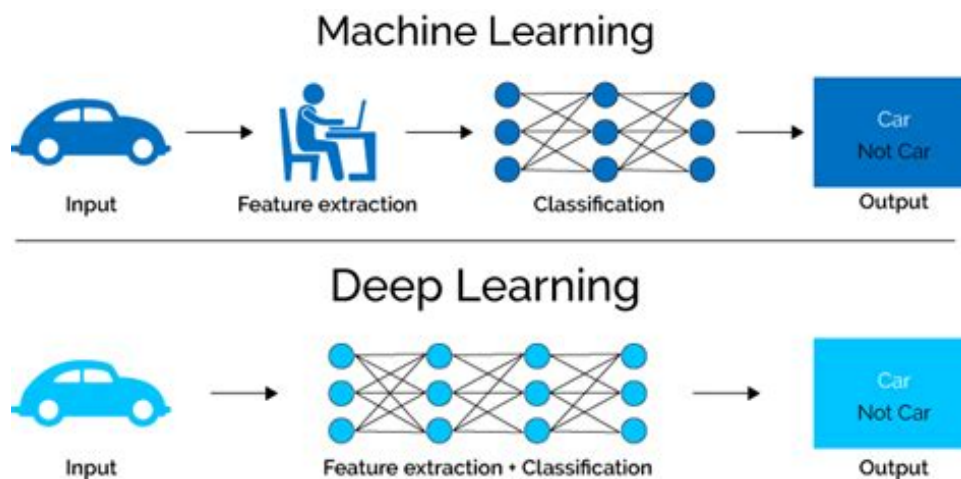


Figure 3.9: Machine Learning VS Deep Learning

Artificial neural networks(ANNs) are learning models inspired by biological neural networks that approximate functions that depend on a large number of inputs(features or data representation).Deep neural networks(DNNs) are ANNs with multiple hidden layers between the input and output layers.Thus, from a given input, they are able to learn features (hidden layers) and to give a classification result (output layer). They have been very successful in NLP for part-of-speech tagging, chunking,named entity recognition, and semantic role labeling [?]. DNNs are good examples of DL and are the focus of this paper. In the remainder of this section, we introduce the topology of conventional neural networks and their initialization.

3.3.2 Feed-forward neural networks

Artificial neural networks see Figure[3.10] are composed of small processing units, namely, neurons, which are connected to each other by weighted connections. Thus, a neuron is activated when it receives a signal,then it spreads out the activation to all the neurons connected to it. The feed-forward NN shown in [3.10] shows a simple type of network with one input layer i , one hidden layer h , and one output layer o .

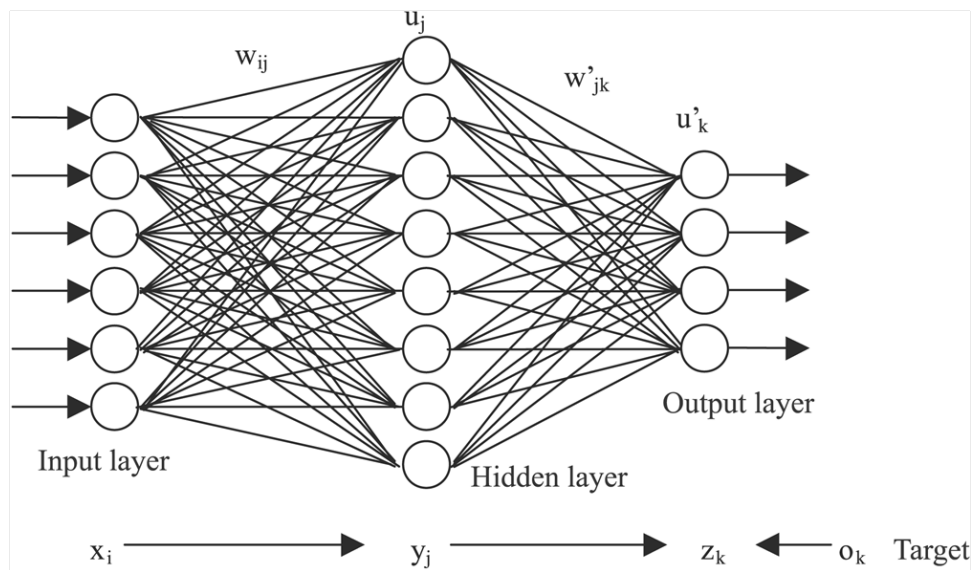


Figure 3.10: Artificial Neural Network

3.3.3 Recurrent neural networks

When you read a book you understand every word based on your understanding of previous words, you don't forget every previous thing and start over again. your thoughts have context and persistence. Old and traditional NN they can't do this, as an example try imagining that you want to track or classify every event that is happening in a video your watching, at every point it's unclear how NNs would solve this problem. because it doesn't have a technique on how to keep the reasoning of the previous events to inform the latter ones.

RNNs addresses this such problem see Figure[3.11], they are an artificial neural networks with loops in them, which allow information to stay in context and to persist.

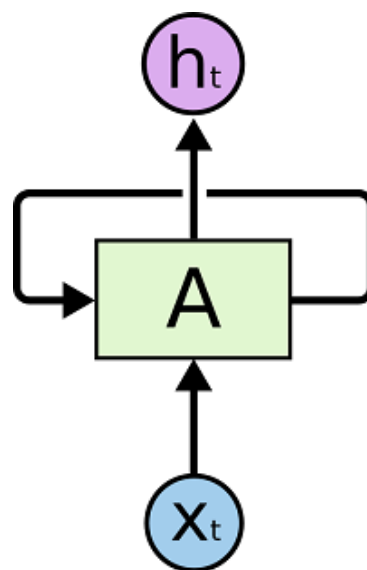


Figure 3.11: Recurrent Neural Networks with a loop.

The Figure[3.11], shows a set of Neural Networks A receives some input x_t and gives an output value of h_t , the loop permits for the information to be passed from one step of the network to the other or next.

An NN that contains direct cycles in their hidden connections is a recurrent neural network (RNN) [20].The hidden layer of an RNN is equal to:

$$h_t = \sigma W_i h_i_t + W_h h h_{t-i}$$

where t denotes the time step and $W_h h$ is the weight matrix representing the recurrent connection. Figure[3.12] illustrates the architecture of a simple RNN showing the connection between its hidden units.

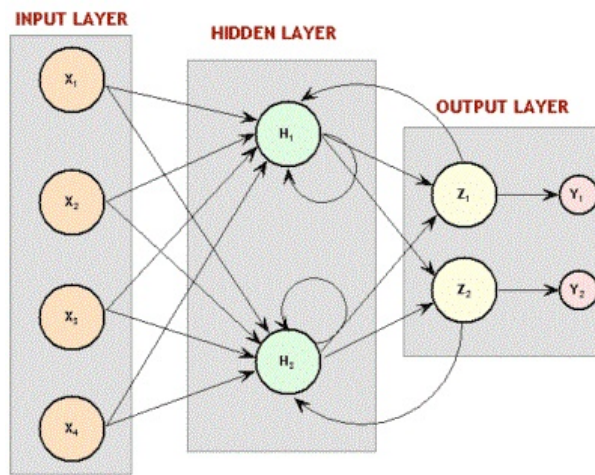


Figure 3.12: Recurrent Neural Network

The loop may look weird on first, but to clarify it think of it as a chunk or group of multiple copies of the same normal neural network , which by turn they pass its output to its successor see Figure[3.13]

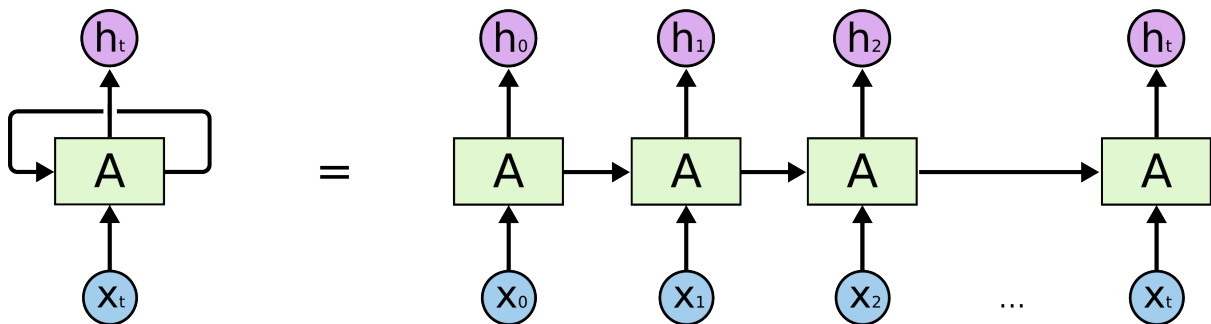


Figure 3.13: unrolled recurrent neural network

3.3.3.1 Long short-term memory

Long Short Term Memory networks – or as they are usually called "LSTMs" are a special kind of RNNs, which are capable of long term dependencies, they were invented by [1] and improved over time. they work well on a lot of problems, and currently are used.

The length of the sequences an RNN can process is limited, because of the *exploding or vanishing gradient problem*. Long short-term memory (LSTM) networks introduce a memory cell that is able to preserve states over long periods of time, overcoming the long-distance dependencies problem of RNNs, The core of the LSTM is a memory cell, c_t , which is recurrently connected to itself. It has three multiplication units: an input gate i_t , a forget gate f_t , and an output gate o_t . These gating vectors are in $[0,1]$. The cell makes selective decisions about what information is preserved, and when to allow access to units, via gates that open and close. [11]

The LSTM transition equations are the following:

$$\begin{aligned} i_t &= \sigma W^i x_t + U^i h_{t-1} + b^i \\ f_t &= \sigma W^f x_t + U^f h_{t-1} + b^f \\ o_t &= \sigma W^o x_t + U^o h_{t-1} + b^o \\ u_t &= \tanh W^u x_t + U^u h_{t-1} + b^u \\ c_t &= i_t o_t + f_t c_{t-1} \\ h_t &= o_t \tanh(c_t) \end{aligned}$$

Where h_t is the hidden unit at time step t , x_t is the input at the current time step, b is a bias, σ is the logistic sigmoid function, and o denotes element-wise multiplication. LSTM can be seen as an NN that contains smart memory units that can remember for an arbitrary length of time. They contain gates that determine when the input is relevant to remember, whether or not it should continue to remember it, and whether or not it should output the value. The value of the gating variables varies for each hidden vector h_t ; thus the model can learn to represent information over multiple time scales t [11].

3.4 Conclusion

In this chapter we have tried to cover some concepts and algorithms in the modern technique applied in the field of sentiment analysis which is Machine Learning and Deep Learning. We have focused in explanation mainly on SVM and LSTM techniques as they are the methods implemented in this work.

The coming section will contain the design of our SA system.

Chapter

4

System Design

“A goal without a plan is just a wish.”

Antoine de Saint-Exupéry

4.1 Introduction

Designing solid, maintainable, and scalable systems require following certain mechanisms and techniques. The modular approach is one of the most effective ones in the field, it is based on subdividing our system into small modules that can be developed and maintained easily.

In this chapter, we are going to introduce and explain the general architecture of our system, its design aspects, and the modular structure it composes of and how it connects all together, also we will get into every step of the general architecture and explain it in much detail.

4.2 General System Architecture

Generally for building a sentiment analysis system or any other system that involves mainly machine learning we need to divide our work into several steps, which allows us to work with each step separately and fix any issues that may appear in the future in a non complicated manner.

The global architecture of our system is composed of several layers or steps, each layer takes an input and provides an output that is needed in next layer in figure 4.1 shows this global architecture as a single entity.

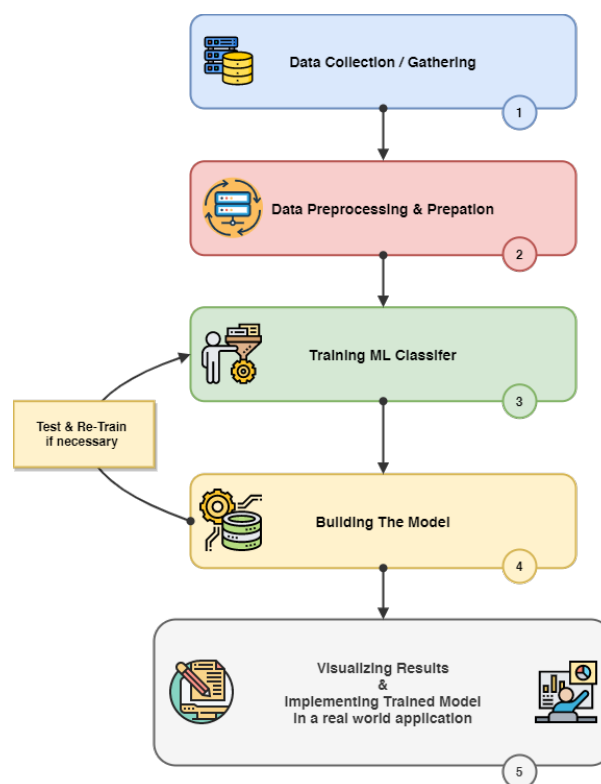


Figure 4.1: The General system architecture

Our system has six main parts starting from collecting data to implementation. *first part*

is the collection of data, here we will gather datasets from several sources, *second part* is preprocessing the collected raw data and extracting the important or relevant informations and removing the noise, this is done by a sequetial process, *third part* involves selecting a classifier and training and testing it, in the *fourth part* we will have our trained model exported and ready to be used. finally the *fifth part* uses the pre-trained model in a real world application.

4.3 Detailed System Architecture

4.3.1 Introduction

Data is the most important player in machine learning and data science fields, it represents the candle that enlightens your works and spits butter and accurate results, the more you have of it the more your work improves.

Several questions may come in mind when thinking about data, questions like where is the data ? how much do i need ? what kind do i need ? We address such questions by saying that data can be collected or found in free or paid third-party services or it can be scrapped manually through internet, concerning how much data you need , well this depends on several factors like features contained in data, regarding these factors we recommend to collect as much as possible specially for deep learning work.

Data may take different shapes and forms and come in different extension and different sources, it can be free or paid and what you need depends on what type of project your doing.

4.3.2 Data Collection/Gathering

In this SA work we will need a set of data represented in a textual form that comes as csv/tsv or simple txt extension as a raw dataset. So we have collected four data sets from different sources and relatively different content, but all of them fall into the field of product reviews of any type, our data is pre-labled according to the sentiment of the review or text as positive or negative, since our work will use supervised machine learning approach. and the table bellow 4.2 represents the collected data with their metadata.

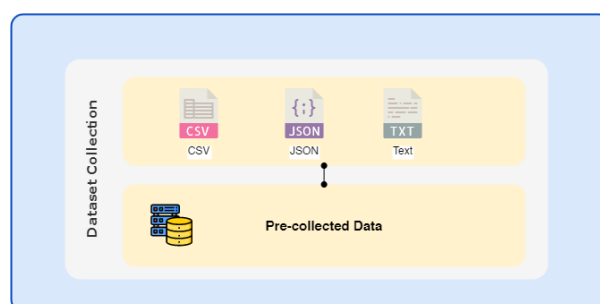


Figure 4.2: Dataset Collection

Dataset name/label	Dataset size	Total Reviews	Positive/Negative Reviews	Source file name/type
Amazon Alexa users reviews	502.69 KB	3150	257 Neg / 2893 Pos	amazon_alexas.tsv
Amazon Reviews: Unlocked Mobile Phones	125.77 MB	413,840 total/ 31,825 removed neutrals/	284,954 Pos / 97061 Neg	Amazon_Unlocked_Mobile.csv
IMDB Dataset of Movie Reviews	63.14 MB	50,000	25K Pos / 25K Neg	IMDB Dataset.csv
Sentence Polarity Dataset v1.0	1.18 MB	10662	5331 Pos / 5331 Neg	sentence_polarity .pos/.neg

Table 4.1: Datasets collection metadata

4.3.3 Data Preprocessing and preparing

Machines can't understand raw text, they understand zeroes and ones and providing our raw text into our machine learning algorithms will not do us any good, therefore processing this textual data into other form that in which our machine learning algorithms can digest is must, also its important to clean our data at first so we can remove all sorts of un-important noise or meaningless information contained in it,like stop words and so on, so here we will go through a process of steps required to provide the final optimal form of our data.

Data preparing

- Tokenization** Tokenization is the process of splitting or dividing the text into small blocks named tokens,numbers or words or punctuations and others can be considered as tokens
- Stop word filter** Common words existing in text can reduce performance and provide no extra value in overall meaning so removing them is necessity, such words are "a", "at" ,"the" and they are called stop words.
- Handling-Negation** Netation words such as "no" , "not", "less", "de-", "dis-" etc can change the polarity or sentiments of the text, thus it is very important to identify them and separate them from the stop words as they are contained in them.
- Stemming** Stemming is a way to reduce a words to its stem,base or root form, by identifying its prefix and stripping it, this important as it reduces the size of the vocabulary and increase performance,(for example, computers — computer, walked — walk)..
- Lemmatization.** Like stemming lemmatization tries also to reduce words to a base form, but it follows different approach, instead of stripping it uses a lexical knowledge to get the base form of words.

Data Vectorization and feature mining

In this step we will go through two approaches to represent data in a form that is suitable for our machine learning classifier. These two approaches are TF-IDF which will be used with SVM algorithm as a feature mining technique and Word2Vec will be used for the Recurrent Neural network, figure 4.3 shows the building blocks of this approach.

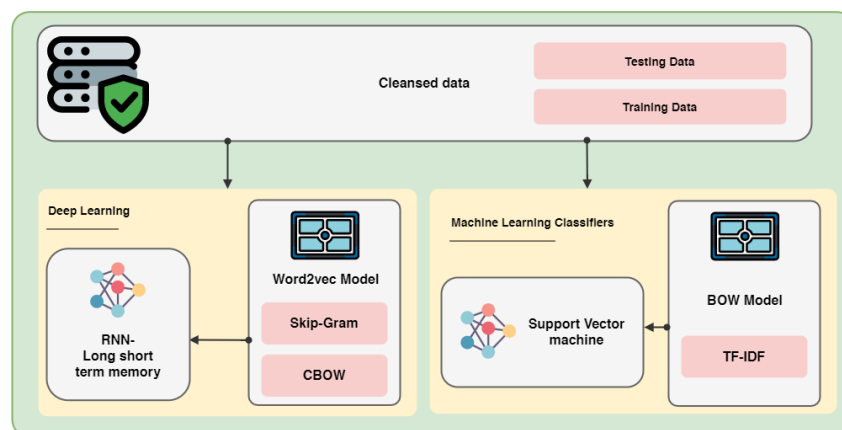


Figure 4.3: Data Vectorization and feature mining step.

- TF-IDF** TFIDF or $tf-idf$, short for term frequency-inverse document frequency, is a numerical measurement that is expected to reflect how significant a word is to a record in an assortment or corpus. The $tf-idf$ value builds relatively to the occasions or times a word shows up in the document and is balanced by the quantity of documents in the corpus that contain the word, which assists with modifying for the way that a few words show up more often in general. [[?]]

$tf-idf$ is one of the most mainstream term-weighting plans today; 83TF-IDF includes:

- Counts** Count the number of times each word appears in a document.
- Frequencies** Calculate the frequency that each word appears in a document out of all the words in the document.

Term frequency: Term frequency (TF) is employed in reference to information retrieval and shows how frequently an expression (term, word) occurs in a very document. Term frequency indicates the importance of a specific term within the general document. Its the number of a specific word W_i occurs in a certain text or review R_j with regard to the entire number of words in review R_j .

$$TF(W_i, R_j) = \frac{\text{No of times } W_i \text{ occurs in } R_j}{\text{Total number of words in } R_j}$$

Inverse document frequency: It represents how much rare or common a word in all documents, or in other words its a measure of how much informations a word pro-

vides. We use it to calculate the weight of rare words in all documents of the entire corpus, rare occurring words in the corpus will get a high IDF score this metric is obtained by dividing the *total number of our documents by the number of documents containing the term* and having the logarithm of that, the following equation represents that.

$$IDF(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

Where N: total number of documents $N = |D|$

$|\{d \in D : t \in d\}|$: number of documents where the term t appears

The final equation of **Term frequency–Inverse document frequency** will be:

$$tf - idf(t, d, D) = TF(W_i, R_j) * IDF(t, D)$$

- **Word2Vec** Word2Vec is a techniques or a set of models in NLP used for producing what so called word embeddings, which are representations of words in a form or large vectors usually several hundreds of size, word2vec requires large input corpus of text to build these representations , with the benefit of keeping the context of words intact and words close in meaning have basically similar vector representations, technically word2vec is a neural network model, or specifically a shallow neural network once it is trained, it can perform this vectoring tasks.[?]

Vector is a representation of quantity as a magnitude which have a direction in space, it makes us determine points in space relative to other. This vectors makes it easy to do math with, and allows us to compare words represented by these vectors for similarity by just calculating euclidean distance between them.

Word2Vec model can be implemented using a couple of methods, these methods are called *Skip Gram* and *CBOW*

CBOW Model: In this technique predicting the word corresponding our context, id done by having the context of each word as the input, lets take this example: *Thats a beautiful flower.*,So if we have the word *beautiful* as an input in the Neural Network and we try to predict the word *flower*, Then specifically we represent our input word as one hot vector and measure and compare the output error to this one hot encoded target word, as process is done we then get the vector word representation of our target word,it is notable that we can use a single context word4.4 or multiple context words4.5 as input.[?].

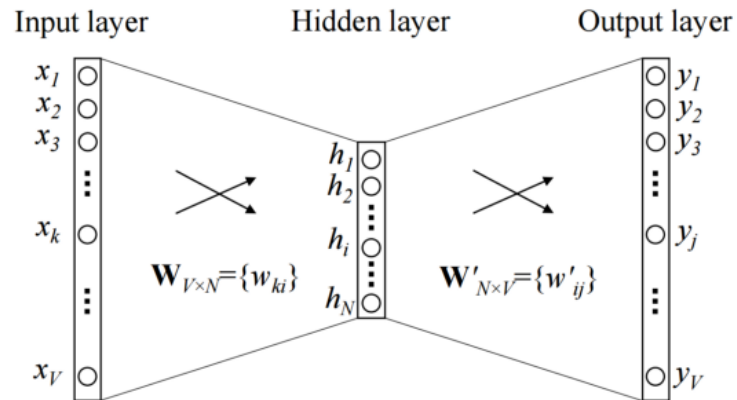


Figure 1: A simple CBOW model with only one word in the context

Figure 4.4: Simple CBOW Model with one context word as input.

The following figure 4.5 represents a multiple context words CBOW MODEL.

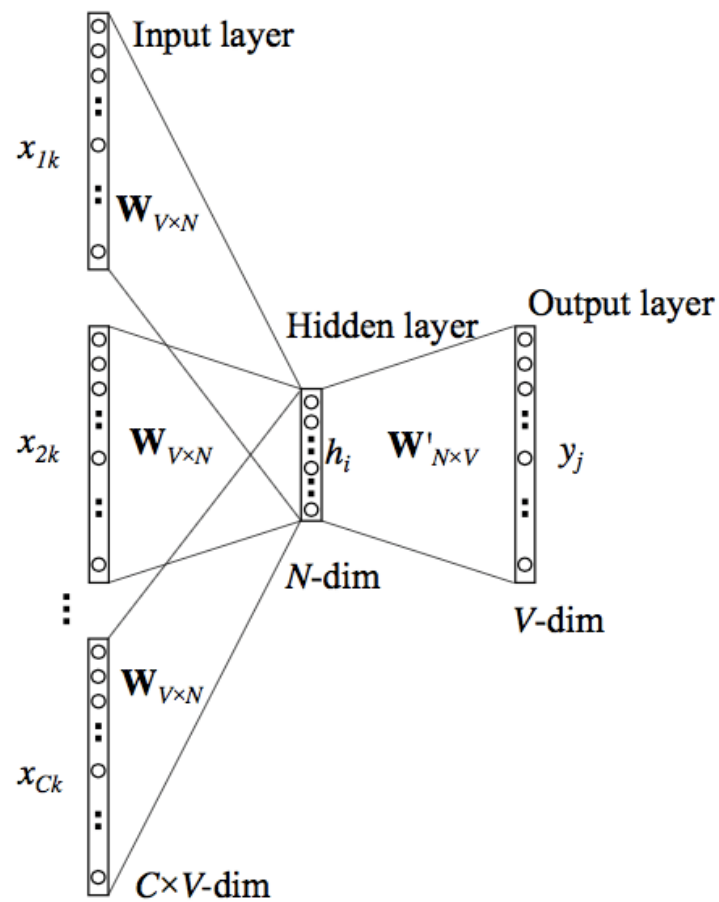


Figure 4.5: Simple CBOW Model with Multiple context words as input.

Skip-Gram model: We can notice that in Skip-Gram model 4.6 that it is a flipping multi CBOW , that is true as some extent, the input is a target word into the network, and we get a probability distribution, and notably both models uses back-propagation for the learning process.[?]

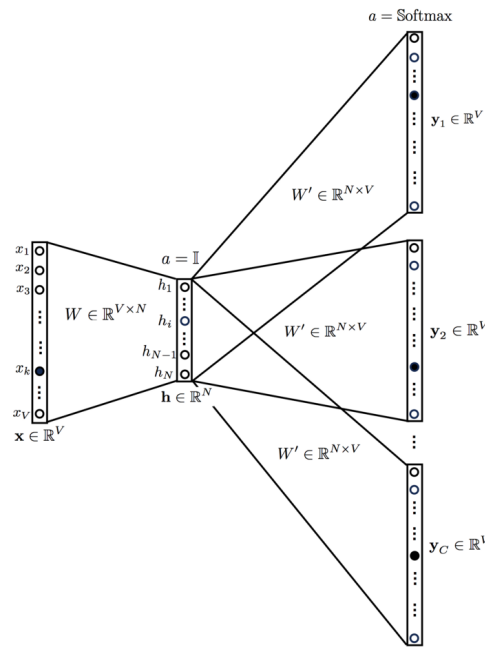


Figure 4.6: Simple Skip-Gram Model.

Which is better ? you may ask, well both have advantages and disadvantages, as for skip-gram it works best with small amount of data, and it can represent rare word too, and for CBOW is much more faster and can highlight representations of more frequent words.[?]

4.3.4 Training the Machine Learning Classifier and Building The model

As for training our model using the cleansed data and the its vectors representations from previous steps, we have chosen tow classifiers in which we have discussed in detail in previous chapter first is Support Vector Machine and the second is a deep learning approach called Long Short term memory or LSTM in short, which is best suited for text analysis by experiments. we have chosen these tow techniques as they have provided the best results in state of the art work in sentiment analysis figure 4.7 shows an illustration of this step of the work. as for getting insights of our classifier we will use the test data to get the accuracy of each classifier.

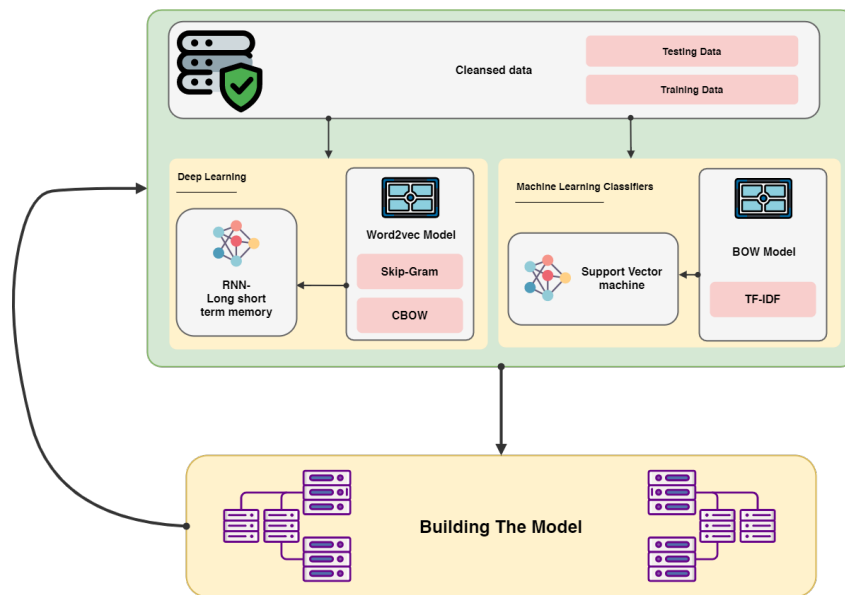


Figure 4.7: Training classifier and building ML Model process.

4.3.5 Visualization of results and Implementation

As for the final step of our work, we can use the saved pre-trained models in production by pushing new unseen raw textual reviews through to our preprocessing steps and then we push the cleansed data to the saved classifier model and get the result in negative or positive prediction figure 4.8 shows the approach.

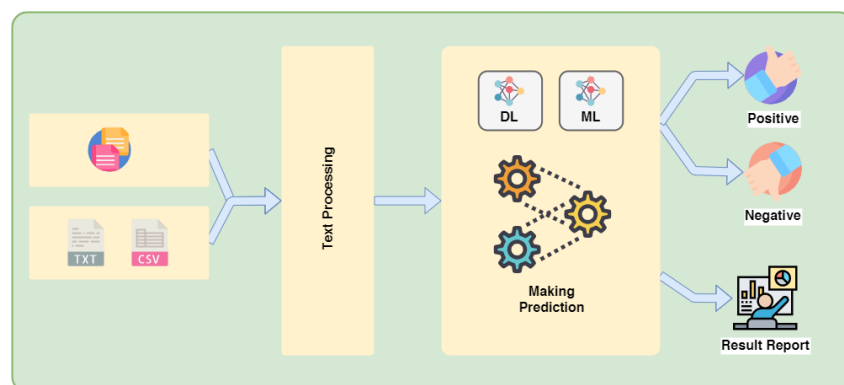


Figure 4.8: Sentiment analysis implementation process.

4.4 Conclusion

In this chapter we went through the design of our system and its steps in some detail, and we have shown its different aspects by steps, the next chapter we will talk about what technologies we have used to implement it.

Chapter

5

System Implementation

“Truth can only be found in one place:
the code. “

Robert C. Martin, Clean Code: A Handbook of Agile Software Craftsmanship

5.1 Introduction

In this chapter we will present the implementation aspects of our sentiment analysis system, we will show the used technologies and frameworks required to build it, also we will walk through all implementation details step by step.

5.2 Work Environment and Development Tools

5.2.1 programming languages

5.2.1.1 Python

Python is a high-level, general-purpose, interpreted, programming language. Created by Guido van Rossum with its first release in 1991, it possesses many great features as its core philosophy is to emphasize code readability and productivity. with the consistent usage of whitespace and tabulation, it supports the powerful object-oriented approach to help and clear the software building process for small or large projects.

Python supports multiple programming paradigms, including structured, oop and functional, it was inspired by languages such as ABC, Haskell, Java, Lisp, Icon, and Perl. python has tow main branches or versions Python 2.x and Python 3.x. Its important to mention that Python 3.X breaks backwards compatibility with Its previous releases like Python 2.X, this happened to correct some design flaws of the language. And in this works we will using Python 3.X.[?]

Python is a programming language that lets you work more quickly and integrate your systems more effectively.[?]



Figure 5.1: Python language logo

5.2.1.2 Javascript

JavaScript or abbreviated as (JS) is a programming language, it conforms to the ECMAScript specification. plus its a high-level, lightweight, interpreted and just-in-time compiled, its also a multi-paradigm It is single-threaded, supporting object-oriented,prototype-based object-orientation, dynamic language, imperative, and declarative programming styles.[?]

JavaScript is used mainly on the web to add functionality to web pages and its executed on the browser as the client side, latest technologies such as Nodejs uses javascript on the server side to program the back-end of web applications. In this work we will be using javascript alongside with HTML/CSS to program the user interface of our application.



Figure 5.2: Javascript language logo

5.2.2 Machine learning kit

5.2.2.1 Tensorflow

TensorFlow is an end-to-end open source library that helps develop and train models for machine learning and deep learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.[?]

TensorFlow was built by a google team named "Google brain", for an internal usage purposes, but they have released it on 2015.



Figure 5.3: TensorFlow Logo

5.2.2.2 Keras

Keras is an open source machine learning and deep learning library written in python language by François Chollet and others as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), it has the capability of running on top of other frameworks and libraries such as tensorflow, it was built with the philosophy of being user friendly, easy to use, modular and extensible library, all this makes it very easy to fast develop deep learning systems without having to dive into crazy details.[?]

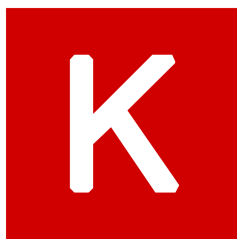


Figure 5.4: Keras Logo

5.2.2.3 Gensim

Gensim is a FREE Python library, used for topic modeling, and doing indexing and similarity retrieval for documents with a large corpora, using modern statistical machine learning. It is implemented in python specifically Cython, Gensim is designed to be scalable, cross platform and efficient which helps to handle large texts collections. it also includes implementations of Word2vec, fastText, tf-idf algorithms in a streamed parallelized fashion.[?]



Figure 5.5: Gensim logo

5.2.2.4 NumPy

NumPy is library for python language that adds support for efficient processing of large, multi-dimensional arrays and matrices, plus it contains a comprehensive collection of mathematical functions to use for working with these matrices and arrays. NumPy is open source,easy to use, performant and can support different hardware and computing platforms.[?]



Figure 5.6: Numpy Logo

5.2.2.5 NLTK

Natural Language Toolkit or NLTK for short is a one of the most powerful python NLP libraries which makes it easy for machines to understand human language data, it comes with an easy to use interface a considerable amount of corpora and lexical resources such as WordNet, plus a bundle of extra text processing libraries for tokenization, stemming etc...[?]

5.2.2.6 Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy, it makes it easy to embed plots into applications with the help of General purpose GUI ¹ tools like tkinter, Pyqt ..., with an object-oriented approach.[?]

5.2.3 Frameworks and tools

5.2.3.1 Flask

Flask is a web framework, or in other words it python module, that is lightweight WSGI which makes it very easy to get started with, and to build web application very fast, it has an ability to scale up to complex applications. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Poocco.

Flask is a microframework that doesn't include an ORM (Object Relational Manager) or such features.[?]



Figure 5.7: Flask Logo

5.2.3.2 Google Colaboratory

Google Colaboratory, sometimes called Colaboratory or google colab for short, is a cloud-based service provided by google with a similar concept to Jupyter Notebook that works in the cloud, no installation required to use it users can access it for free from the browser, with a google account.[?]

Google Colaboratory provides users with a complete python environment plus tons of machine learning libraries with a free RAM ², CPU ³, GPU ⁴ and even a TPU ⁵ to accelerate what

¹Graphical User Interface

²Random Access Memory

³Central Processing Unit

⁴Graphical Processing Unit

⁵Tensor Processing unit

you work on, writing code is done through a cell-oriented paradigm, also it allows to combine code and rich text in a single document, along with images, HTML, LaTeX and more, when creating colab notebooks they are saved automatically in the user google drive, thus making it easy to be shared among developers.



Figure 5.8: Google colab Logo

5.3 Preparing collected data

As discussed in previous chapter, we have collected four datasets to work on, all four have textual reviews which are labeled to "positive" or "negative" according to their sentiment. The following is a summary of our used datasets.

- **Amazon Alexa users reviews** with 3150 reviews, 257 Neg / 2893 Pos
- **Amazon Reviews: Unlocked Mobile Phones** with 382075 reviews, 284,954 Pos / 97061 Neg
- **IMDB Dataset of Movie Reviews** with 50,000 reviews, 25,000 Pos / 25,000 Neg
- **Sentence Polarity Dataset v1.0** with 10662 reviews, 5331 Pos / 5331 Neg

5.3.1 Preprocessing data:

Next we will preprocess or clean our datasets by tokenization and removing stop words etc... then we combine them into one file, the following function is used to preprocess the data 5.9.

```

1 lemma = WordNetLemmatizer()
keywords = []
def clean_text(text,doLemma = True,doStem = False,posTag = True):
    # Convert the text into lowercase
    text = text.lower().strip()
    #Tokenization or Split into list
    wordList = regep_tokenize(text,"[w']+")
    # Remove punctuation
    wordList = ["".join(x for x in word if (x not in punc)) for word in wordList if (word not in punc)]
    # Remove other keywords
    wordList = [word for word in wordList if word not in keywords]
    # Stemmer
    if (doStem):
        pass
    # Lemmatization
    try:
        if (doLemma):
            if posTag:
                wordList = [lemma.lemmatize(word,get_wordnet_pos(word)) for word in wordList]
            else:
                wordList = [lemma.lemmatize(word) for word in wordList]
    except Exception as e:
        print("error",e)
        print(text)
    # Remove stopwords except negations
    wordList = [word for word in wordList if word not in stop]
    return " ".join(wordList)

```

Figure 5.9: Clean Dataset function

5.3.2 Merging data:

This next python script is used to perform the Merging of the datasets as shown in figure 5.10.

```

[] preprocessed_dataset_location_sentence_polarity_prep = '/content/drive/My Drive/\
SENTIMENT_ANALYSIS_PREPROCESSED_DATASETS/sentence_polarity_prep.csv'
preprocessed_dataset_location_IMDB_Dataset_prep = '/content/drive/My Drive/\
SENTIMENT_ANALYSIS_PREPROCESSED_DATASETS/IMDB_Dataset_prep.csv'
preprocessed_dataset_location_Amazon_Unlocked_Mobile_prep = '/content/drive/My Drive/\
SENTIMENT_ANALYSIS_PREPROCESSED_DATASETS/Amazon_Unlocked_Mobile_prep.csv'
preprocessed_dataset_location_amazon_alexa_prep = '/content/drive/My Drive/\
SENTIMENT_ANALYSIS_PREPROCESSED_DATASETS/amazon_alexa_prep.csv'
datasets = [preprocessed_dataset_location_sentence_polarity_prep,preprocessed_dataset_location_IMDB_Dataset_prep,preprocessed_data

li = []
for d in datasets:
    df = pd.read_csv(d, index_col=None, header=0)
    li.append(df)

prep_dataset = pd.concat(li, axis=0, ignore_index=True)

print(prepare_dataset.shape)
print(prepare_dataset.dtypes)

[] (445887, 2)
clean_text    object
feedback      int64
dtype: object

```

Figure 5.10: Python script to Merge data

5.3.3 Combined Dataset stats:

The following figure 5.11 shows the stats of the final merged data, as we can see in the pie chart we have 127,495 Negative reviews, and 317.294 Positive reviews.

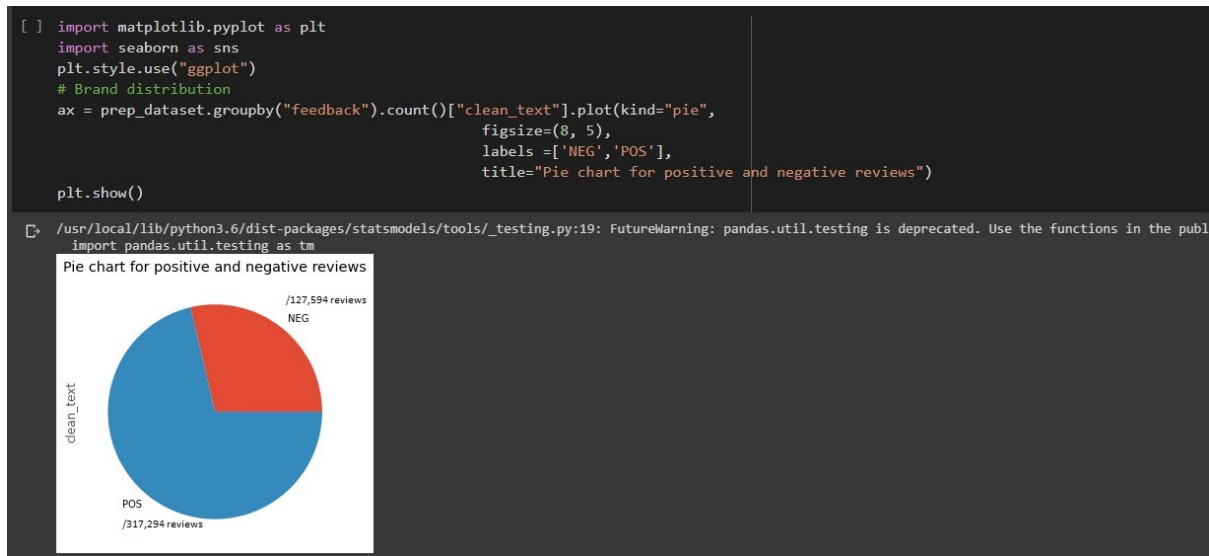


Figure 5.11: Positive and Negative reviews pie chart stats

5.3.4 Splitting Dataset:

Finally we split our data set into 80% *train* data and 20% *test* data as shown in following script 5.12.

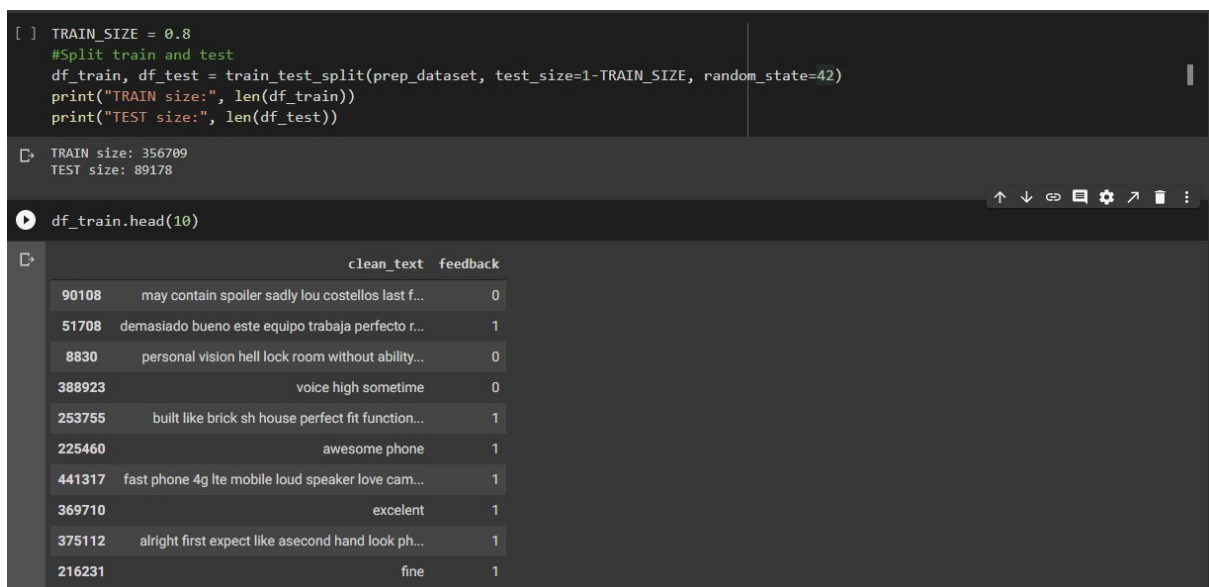


Figure 5.12: Splitting data script.

5.4 Sentiment analysis in SVM

In this section we are going to use Support Vector machine to build our sentiment analysis classifier.

5.4.1 Feature Extraction

Before training our model, feature extraction is a required step to represent the distinct features of the data into numerical form, in which our model will learn from, many techniques can be used, but in this case we have chosen use TF-IDF, the next figure 5.13 shows the script that performs this technique.

```
%%time
# Create feature vectors
vectorizer = TfidfVectorizer(min_df = 5,
                             max_df = 0.8,
                             sublinear_tf = True,
                             use_idf = True)
```

Figure 5.13: TF-IDF Feature Extractor.

5.4.2 Training model

Training SVM linear model.

```
[ ] svm_classifier = svm.LinearSVC(C=0.1)
vectorizer_pipeline = Pipeline(['vectorizer', vectorizer], ('classifier', svm_classifier))
vectorizer_pipeline.fit(X_train, y_train)
joblib.dump(vectorizer_pipeline, SVM_MODEL, compress=3)

CPU times: user 12.8 s, sys: 205 ms, total: 13 s
Wall time: 13 s
```

Figure 5.14: SVM training.

5.4.3 Model evaluation

Finally we evaluate the performance of our model through some statistics, the following code in figure 5.15 makes predictions on the test data and outputs the result, which represents 93% accuracy.

```

[ ] %%time
y_predictions = vectorizer_pipeline.predict(X_test['clean_text'])

print(classification_report(y_test, y_predictions))
tn, fp, fn, tp = confusion_matrix(y_test, y_predictions).ravel()
print("TRUE NEGATIVE : ",tn,"</#> TRUE POSITIVE : ",tp,"\\nFALSE NEGATIVE : ",fn,"</#> FALSE POSITIVE:",fp)

```

	precision	recall	f1-score	support
0	0.89	0.87	0.88	25728
1	0.95	0.96	0.95	63450
accuracy			0.93	89178
macro avg	0.92	0.91	0.92	89178
weighted avg	0.93	0.93	0.93	89178

```

TRUE NEGATIVE : 22284 </#> TRUE POSITIVE : 60778
FALSE NEGATIVE : 2672 </#> FALSE POSITIVE: 3444
CPU times: user 2.66 s, sys: 1.87 ms, total: 2.66 s
Wall time: 2.66 s

```

Figure 5.15: Testing Svm accuracy.

Also we check some points on the confusion matrix in figure 5.16, by observing the results we have in the confusion matrix we can see that we got 96% correct predictions as negative and 87% correct predictions as positive as a correct result, on the other hand we see 13% incorrect predictions as negative while they were positive, and less than 1% predicted as positive while they are negative, the overall predictions shows that the classifier gives good performance.

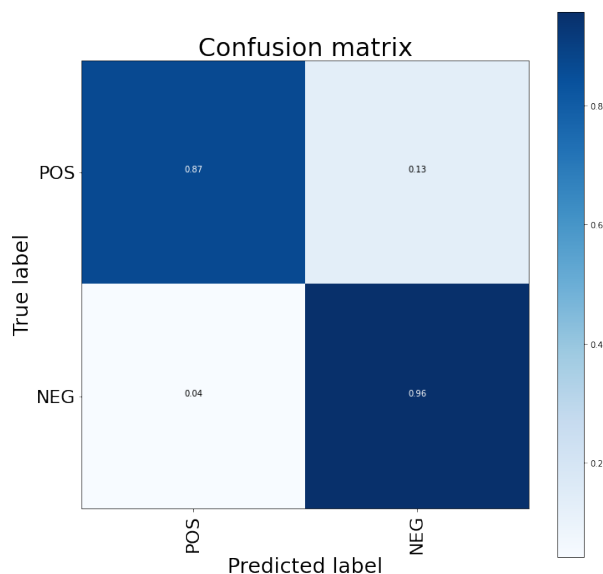


Figure 5.16: Confusion matrix.

5.5 Sentiment analysis in RNN-LSTM

5.5.1 Vectoring data

In this step we are going to transform our data into digestible form for our classifier, a word2vec vectorizer as discussed before is one of the best vectorizers out there, so we use it, and the next script in figure 5.17 using gensim library will train and build our vectorizer.

```
[ ] # WORD2VEC
W2V_SIZE = 300
W2V_WINDOW = 7
W2V_MIN_COUNT = 10

w2v_model = gensim.models.word2vec.Word2Vec(size=W2V_SIZE,
                                           window=W2V_WINDOW,
                                           min_count=W2V_MIN_COUNT,
                                           workers=8)

[ ] w2v_model.build_vocab(documents)

[ ] %%time
w2v_model.train(documents, total_examples=len(documents), epochs=W2V_EPOCH)

[ ] CPU times: user 24min 49s, sys: 4.87 s, total: 24min 54s
Wall time: 12min 44s
(327248525, 370239360)
```

Figure 5.17: Word2Vec Vectorizer.

The step above, builds the vocabulary, and starts training the Word2Vec model, Word2vec is a shallow neural network, which means a neural network with a single hidden layer, and what's happening behind the scenes is that we are training this neural network to make able to predict a word based on its context, but in this case we are going to abandon the neural network and not use it after the training process has ended, instead we are going to learn the weights of the hidden layer. Such weights represent the word vectors that we need and have tried to learn, another name for such vectors is embedding.

Explaining parameters:

The following is an explanation of the word2vec method parameters.

size: This parameter is used to define the size of vector which represents each word or token (i.e. the context or neighboring words).

window: This represents the max distance between the target word and its neighboring word.

min count: Minimum frequency count of words.

workers: Number of threads used.

epochs: Number of iterations (epochs) over the corpus.

Testing word2vec:

The following code 5.18 shows some results when testing for similarity of words.

```
w2v_model.wv.most_similar("suck")

[('horrible', 0.608102023601532),
 ('terrible', 0.5452688932418823),
 ('bad', 0.5334455966949463),
 ('crappy', 0.5224913358688354),
 ('awful', 0.5164670944213867),
 ('crap', 0.43713271617889404),
 ('stink', 0.41620346903800964),
 ('okay', 0.4016824960708618),
 ('ok', 0.393736869096756),
 ('stupid', 0.3842628598213196)]

[ ] w2v_model.wv.most_similar("love")

[('amaze', 0.532701313495636),
 ('awesome', 0.515298068523407),
 ('great', 0.4992554783821106),
 ('beautiful', 0.41500020027160645),
 ('perfect', 0.4141337275505066),
 ('enjoy', 0.40976789593696594),
 ('nice', 0.4060148000717163),
 ('wonderful', 0.40524429082870483),
 ('hate', 0.4036189317703247),
 ('inlove', 0.400452196598053)]
```

Figure 5.18: Word2Vec results testing.

5.5.2 Building and Training RNN-LSTM model

Defining the LSTM network architecture:

- **Embedding Layer:** a layer that converts our word tokens into Embeddings.
- **Dropout Layer:** We have used a dropout layer which drops or ignores units (i.e. neurons) during the training phase of certain set of neurons which is chosen at random, this is done to prevent overfitting.
- **LSTM Layer:** The long short term memory layer.
- **Dense Layer:** a regular deeply connected neural network layer with a sigmoid activation function.

The following python code 5.19 is for building this network:

```
[ ] #Building model
model = Sequential()
model.add(embedding_layer)
model.add(Dropout(0.5))
model.add(LSTM(100, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(1, activation='sigmoid'))

model.summary()
```

⌘ WARNING:tensorflow:Layer lstm will not use cuDNN kernel since it doesn't meet the cuDNN kernel criteria. Model: "sequential"

Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, 300, 300)	35687700
dropout (Dropout)	(None, 300, 300)	0
lstm (LSTM)	(None, 100)	160400
dense (Dense)	(None, 1)	101

=====
Total params: 35,848,201
Trainable params: 160,501
Non-trainable params: 35,687,700

Figure 5.19: Lstm network.

The next step is to Training the LSTM network through 8 epochs figure5.20 shows the training process:

```
[ ] history = model.fit(x_train, y_train,
                        batch_size=BATCH_SIZE,
                        epochs=EPOCHS,
                        validation_split=0.1,
                        verbose=1,
                        callbacks=callbacks)
```

⌘ Epoch 1/8
314/314 [=====] - ETA: 0s - loss: 0.2790 - accuracy: 0.8765WARNING:tensorflow:Early stopping conditioned on metric `val_loss` which has not moved in 3 epochs.
314/314 [=====] - 363s 1s/step - loss: 0.2790 - accuracy: 0.8765 - val_loss: 0.2136 - val_accuracy: 0.9178
Epoch 2/8
314/314 [=====] - ETA: 0s - loss: 0.2097 - accuracy: 0.9148WARNING:tensorflow:Early stopping conditioned on metric `val_loss` which has not moved in 3 epochs.
314/314 [=====] - 360s 1s/step - loss: 0.2097 - accuracy: 0.9148 - val_loss: 0.1893 - val_accuracy: 0.9256
Epoch 3/8
314/314 [=====] - ETA: 0s - loss: 0.1912 - accuracy: 0.9234WARNING:tensorflow:Early stopping conditioned on metric `val_loss` which has not moved in 3 epochs.
314/314 [=====] - 348s 1s/step - loss: 0.1912 - accuracy: 0.9234 - val_loss: 0.1788 - val_accuracy: 0.9319
Epoch 4/8
314/314 [=====] - ETA: 0s - loss: 0.1803 - accuracy: 0.9282WARNING:tensorflow:Early stopping conditioned on metric `val_loss` which has not moved in 3 epochs.
314/314 [=====] - 349s 1s/step - loss: 0.1803 - accuracy: 0.9282 - val_loss: 0.1655 - val_accuracy: 0.9355
Epoch 5/8
314/314 [=====] - ETA: 0s - loss: 0.1743 - accuracy: 0.9311WARNING:tensorflow:Early stopping conditioned on metric `val_loss` which has not moved in 3 epochs.
314/314 [=====] - 357s 1s/step - loss: 0.1743 - accuracy: 0.9311 - val_loss: 0.1608 - val_accuracy: 0.9394
Epoch 6/8
314/314 [=====] - ETA: 0s - loss: 0.1690 - accuracy: 0.9331WARNING:tensorflow:Early stopping conditioned on metric `val_loss` which has not moved in 3 epochs.
314/314 [=====] - 355s 1s/step - loss: 0.1690 - accuracy: 0.9331 - val_loss: 0.1561 - val_accuracy: 0.9402
Epoch 7/8
314/314 [=====] - ETA: 0s - loss: 0.1647 - accuracy: 0.9351WARNING:tensorflow:Early stopping conditioned on metric `val_loss` which has not moved in 3 epochs.
314/314 [=====] - 354s 1s/step - loss: 0.1647 - accuracy: 0.9351 - val_loss: 0.1557 - val_accuracy: 0.9403
Epoch 8/8
314/314 [=====] - ETA: 0s - loss: 0.1612 - accuracy: 0.9366WARNING:tensorflow:Early stopping conditioned on metric `val_loss` which has not moved in 3 epochs.
314/314 [=====] - 349s 1s/step - loss: 0.1612 - accuracy: 0.9366 - val_loss: 0.1499 - val_accuracy: 0.9423

Figure 5.20: Lstm Training.

5.5.3 Model evaluation

Finally we evaluate the performance of our model as shown in figure 5.21 and we see that we have reached 94% accuracy.

```
[ ] %%time
score = model.evaluate(x_test, y_test, batch_size=BATCH_SIZE)
print()
print("ACCURACY:",score[1])
print("LOSS:",score[0])

88/88 [=====] - 15s 173ms/step - loss: 0.1483 - accuracy: 0.9442

ACCURACY: 0.9442014694213867
LOSS: 0.148322194814682
CPU times: user 14.4 s, sys: 659 ms, total: 15.1 s
Wall time: 15.5 s
```

Figure 5.21: Model loss/accuracy evaluation.

We also view the confusion 5.22 matrix to get more insights,we can see in the results that we got 95% correct predictions as negative and 89% correct predictions as positive as a correct result, on the other hand we see 11% incorrect predictions as negative while they were positive, and less than 1% predicted as positive while they are negative, the overall predictions shows that the classifier gives good performance.

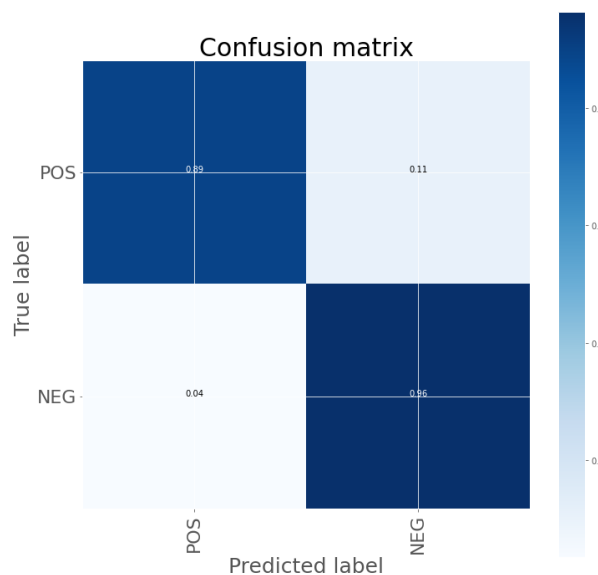


Figure 5.22: Lstm Confusion matrix.

5.6 Result Comparison

By Comparing results between SVM model and LSTM model we see that LSTM performs better than SVM, with an accuracy of 94% and through expereremations we see that LSTM

detects negations in test reviews were SVM does not.

5.7 Implementation and deployment

5.7.1 Exporting models

To use this models in a real world implementation, we need to export them into usable saved model which are called "pretrained models" these pretrained models can be loaded later to use in other applications figure 5.23 shows the export process, we note also that we need to export the preprocessor.

```

▶ #Saving the trained model
model_location = f"/content/drive/My Drive/SENTIMENT_ANALYSIS_PREPROCESSED_DATASETS/{KERAS_MODEL}"
model.save(model_location)

[ ] #Saving preprocessor
model_location = f"/content/drive/My Drive/SENTIMENT_ANALYSIS_PREPROCESSED_DATASETS/{TOKENIZER_MODEL}"
pickle.dump(tokenizer, open(model_location, "wb"), protocol=0)

```

Figure 5.23: Exporting models.

5.7.2 Sentiment analysis web interface

5.7.2.1 User interface preview

The following 5.24 is the user interface our web application built for providing a simple implementation of our machine learning pretrained models,For the case of predicting sentiments of product reviews, this web project is deployed at Heroku ⁶ and can be tested online, it can be reached with this link ⁷.

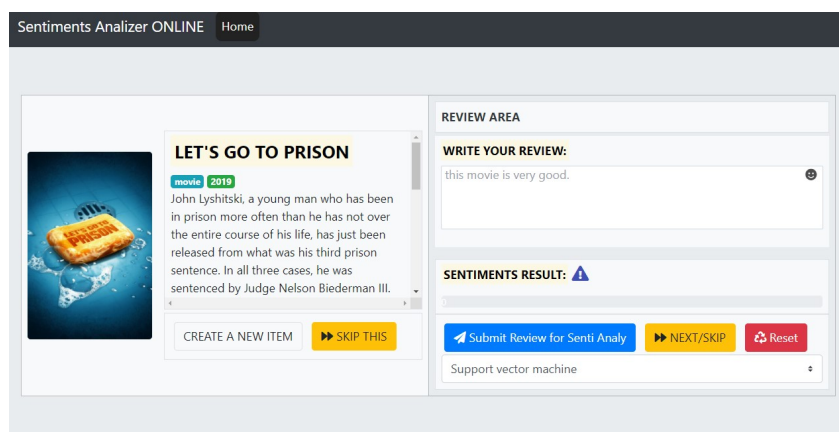


Figure 5.24: Sentiment analysis web interface.

⁶Heroku is a platform as a service (PaaS) to enable developers build applications in the cloud

⁷<http://emotions-analysis-app.herokuapp.com/>

5.7.2.2 User interface Detailed Explanation

In this section we will explain the components of our previous interface, as we have it displayed and dissected in the bellow figure 5.25.

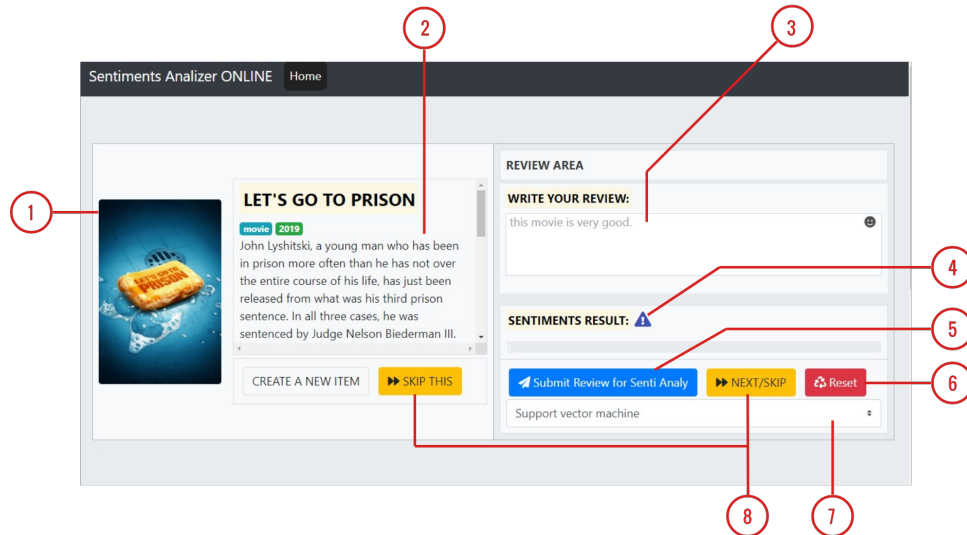


Figure 5.25: Sentiment analysis web interface components.

1. This represent the product image, in our case its a movie cover.
2. This is a description of our product.
3. This is a text area where we can provide feedback or a review for the presented product.
4. The part the provides the sentiment analysis result, it can give "Negative" or "Positive" result.
5. Button to submitting the review for analysis.
6. Button to reset all the fields.
7. A drop-down list to select a classifier, in our case we have SVM or LSTM.
8. Buttons to skip or move to next product for review.

5.7.2.3 A Simple Usage Scenario

As a simple usage scenario of our web application, we choose to provide certain reviews for a specific movies then we will submit and see the returned results. In figure 5.27 we have provided the following review *"this was a very good movie, and i have enjoyed watching it."* and was classified by LSTM as POSTIVE with 98% score, on the other hand in figure 5.26 we

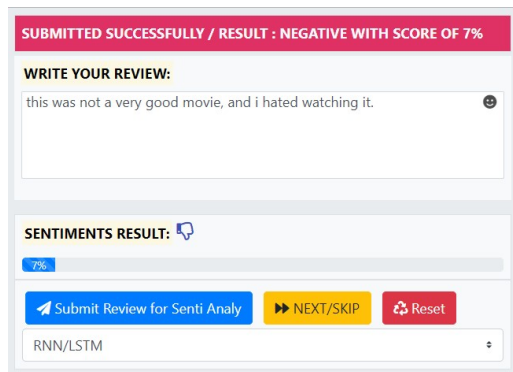


Figure 5.26: Negative review

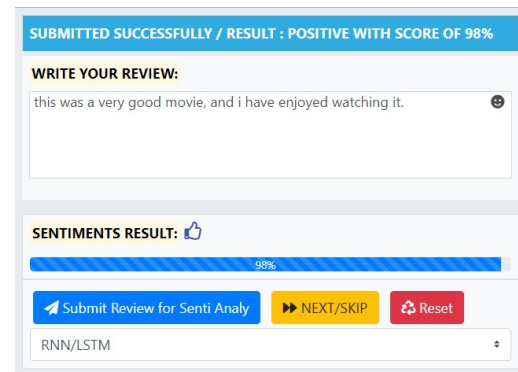


Figure 5.27: Positive review

submitted the following review "*this was not a very good movie, and i hated watching it.*" and was classified correctly as NEGATIVE with 7% score.

5.8 Conclusion

Through out the sections of this chapter, We have tried to Implement Sentiment Analysis using Machine Learning Svm and Deep learning LSTM whereas we have shown the technologies used and their implementation by step, and their deployment in a real world web application.

Chapter

6

Conclusion and Future work

“Success in creating AI would be the biggest event in human history.
Unfortunately, it might also be the last, unless we learn how to avoid the risks.”

Stephen Hawking

Sentiment analysis or what so called opinion mining is a very important field in AI and specifically in NLP field, thanks to its importance for real world businesses and politics and other fields, in this work we have tried to tackle this problem with high technologies in machine learning and deep learning as they are most effective which provides high performance and accurate results.

Since new technologies and new research is in NLP is emerging and being developed, the need to extend our sentiment analysis system is a must, to add more work in the future we suggest some points.

- Extending the models to cover arabic and algerian dialect languages.
- Implementing new and state of the art language models such as GPT-2, Transformers.
- Trying to collect more data and train our model again.
- Modify our system to be able to predict neutral .
- Extend our system be able to detect sentiments in articles and large documents .
- Trying to build more general purpose system to analyze sentiments for any language context.

Bibliography

- [1] arXiv.org. Deep learning for sentiment analysis : A survey. <https://arxiv.org/ftp/arxiv/papers/1801/1801.07883.pdf>, 2017. Accessed: 2020-06-01.
- [2] arXiv.org. Natural language processing: State of the art, current trends and challenges. <https://arxiv.org/pdf/1708.05148v1.pdf>, 2017. Accessed: 2020-06-01.
- [3] M. Avinash and E. Sivasankar. A study of feature extraction techniques for sentiment analysis. *Advances in Intelligent Systems and Computing*, 814:475–486, 2019.
- [4] J. Brownlee. *Master Machine Learning Algorithms*. Australia, 2017.
- [5] E. Definition. Emotions dictionary definition. <https://www.merriam-webster.com/dictionary/emotion>, 2020. Accessed: 2020-06-01.
- [6] E. O. Definition. Emotions dictionary definition. <https://dictionary.cambridge.org/us/dictionary/english/opinion>, 2020. Accessed: 2020-06-01.
- [7] O. Definition. Emotions dictionary definition. <https://www.merriam-webster.com/dictionary/opinion>, 2020. Accessed: 2020-06-01.
- [8] S. Definition. Sentiment dictionary definition. <https://www.merriam-webster.com/dictionary/sentiment>, 2020. Accessed: 2020-06-01.
- [9] T. Dholpuria, Y. Rana, and C. Agrawal. A sentiment analysis approach through deep learning for a movie review. pages 173–181, 11 2018.
- [10] K. Ganesan and C. Zhai. Opinion-based entity ranking. *Information Retrieval*, 15(2):116–150, Apr 2012.
- [11] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
- [12] Z. Hu, J. Hu, W. Ding, and X. Zheng. Review sentiment analysis based on deep learning. pages 87–94, 10 2015.

- [13] D. Khurana, A. Koli, K. Khatter, and S. Singh. Natural language processing : State of the art , current trends and challenges natural language processing : State of the art , current trends and challenges department of computer science and engineering manav rachna international university , faridabad-. *arXiv preprint arXiv*, (August 2017), 2018.
- [14] lexalytics.com. Top applications of sentiment analysis and text analytics. <https://www.lexalytics.com/applications>, 2020. Accessed: 2020-06-01.
- [15] medium. Classification algorithms types. <https://medium.com/@Mandysidana/machine-learning-types-of-classification-9497bd4f2e14>, 2020. Accessed: 2020-01-01.
- [16] monkeylearn. An introduction to support vector machines (svm). <https://monkeylearn.com/blog/introduction-to-support-vector-machines-svm/>, 2020. Accessed: 2020-06-01.
- [17] U. of Cambridge Computer Laboratory. Natural language processing. <https://www.cl.cam.ac.uk/teaching/2002/NatLangProc/>, 2020. Accessed: 2020-06-01.
- [18] D. S. R. Pooja Dinkar Shinde. A comparative study of sentiment analysis techniques. *International Journal of Innovations and Advancement in Computer Science IJIACS*, 7:771–777, mar 2018.
- [19] U. C. J. Praveen Kumar. A comparative study on sentiment analysis and opinion mining. *International Journal of Engineering and Technology*, 8(2):938–943, Apr 2016.
- [20] L. M. Rojas-Barahona. Deep learning for sentiment analysis. *Language and Linguistics Compass*, 10(12):701–719, 2016.
- [21] Z. Singla, S. Randhawa, and S. Jain. Sentiment analysis of customer product reviews using machine learning. pages 1–5, 06 2017.
- [22] J. Souza, A. Oliveira, G. Coelho de Andrade, and M. Alexandra. *A Deep Learning Approach for Sentiment Analysis Applied to Hotel’s Reviews*, pages 48–56. 01 2018.
- [23] towardsdatascience. Decision trees in machine learning. <https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052>, 2017. Accessed: 2020-06-01.

-
- [24] wearesocial. Digital 2020 internet stats. <https://wearesocial.com/blog/2020/01/digital-2020-3-8-billion-people-use-social-media>, 2020. Accessed: 2020-06-01.
- [25] www.cs.uic.edu. Sentiment analysis and opinion mining. <https://www.cs.uic.edu/~liub/FBS/>, 2012. Accessed: 2020-06-01.
- [26] A. Yousefpour, R. Ibrahim, H. N. A. Hamed, and M. s. Hajmohammadi. A comparative study on sentiment analysis. *Advances in Environmental Biology*, 8:53–68, 08 2014.
- [27] L. Zhang and B. Liu. *Sentiment Analysis and Opinion Mining*, pages 1152–1161. Springer US, Boston, MA, 2017.

Erratum

- we apologize for some errors in chapters
- We thank everyone, and apologize for any errors in any chapters. You report any errors to me, this is my address : nacersalaheddine05@gmail.com