

SENTIMENT ANALYSIS FOR PRODUCT RATING

A MINI PROJECT REPORT

Submitted by

R.SARAVANAN

Reg No: 814419104021

S.SATHISH KUMAR

Reg No: 814419104022

in partial fulfilment for the award of the degree

of

**BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING
SUDHARSAN ENGINEERING COLLEGE**

SATHIYAMANGALAM

ANNA UNIVERSITY: CHENNAI 600 025

2022

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report “ **SENTIMENT ANALYSIS FOR PRODUCT RATING** ” is the bonafide work of “ **R.SARAVANAN (814419104021), S.SATHISH KUMAR (814419104022)** ” who carried out the project work under my supervision.

SIGNATURE

Dr.P.SUJATHA M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

Associate Professor
Computer Science & Engineering
Sudharsan Engineering College
Sathiyamangalam – 622 501
Pudukkottai District.

SIGNATURE

Mrs.R.KAMALITTA M.E.,

SUPERVISOR

Assistant Professor
Computer Science & Engineering
Sudharsan Engineering College
Sathiyamangalam – 622 501
Pudukkottai District.

Submitted for the Project Viva Voce held on -----

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

First and Foremost, I thank **GOD** and **My Parents** for their blessings on Me to complete this project successfully.

At this moment of accomplishment , first of all I pay homage to **Dr.S.KARTHIKEYAN M.E.,Ph.D., Principal, Sudharsan Engineering College , Sathiyamangalam**, for giving me an opportunity to do this project.

I express my whole hearted thanks to **Dr.P.Sujatha M.E.,PhD., Associate Professor Head of the Department CSE, Sudharsan Engineering College, Sathiyamangalam**, for guided me in an inspiring manner to complete my project successfully.

I express my thanks to my project supervisor **Mrs.R.KAMALITTA M.E., Assistant Professor, Department of CSE, Sudharsan Engineering Colleges, and other staff members of CSE Department** for their kind cooperation.

Last but not the least, I thank all **My Family Members** and **My Friends** For helping me to finish this project work successfully.

ABSTRACT

Rating of a product has become an important factor to determine the quality of a product in recent days. Product Rating system is a system that detects hidden sentiments in reviews and rates the product accordingly. The system uses sentiment analysis methodology in order to achieve desired functionality. The reviews provided by the users about a product on an e-commerce website is analyzed and review of that product is generated based on the review. This system completely eradicates the trouble of giving rating as well as writing review and helps to generate accurate rating based on user reviews. The output of the project is a web app where users can write a review of a product. The review is the passed to natural language processor that analyzes the review and generate rating based on the review. The rating is out of 5. The overall rating of the product is the average of all the individual ratings.

Key Words: Sentiment Analysis, Neural Networks, Product Reviews, Rating, Users, E-commerce.

LIST OF CONTENTS

ABSTRACT	i
LIST OF CONTENTS	ii
1. INTRODUCTION	1
1.1 Background	1
1.2 Problem Statement	1
1.3 Objectives	1
1.4 System Features	2
1.5 System Specifications	2
1.5.1 Hard Specification	2
1.5.2 Software Specification	2
1.6 Challenges	3
2. LITERATURE REVIEW	3
2.1 History	3
2.2 Deep Learning for NLP	3
3. METHODOLOGY	4
3.1 Word Vector	4
3.2 Recurrent Neural Network	7
3.3 Hyperparameter Tuning	9
3.4 Implementation Details	9
3.5 Data Cleaning and Preprocessing	10
3.6 Model Visualization	11
4. SYSTEM DEVELOPMENT	12

5. SYSTEM DESIGN AND ARCHITECTURE	14
5.1 Flowchart	14
5.2 UML Diagrams	15
5.2.1 Use Case Diagram	15
5.2.2 Context Diagram	16
5.2.3 Sequence Diagram	18
5.2.4 Class Diagram	19
5.3 Block Diagram	20
6. PROGRAM	21
7. RESULT	24
8. REFERENCES	28

1. INTRODUCTION

1.1 Background

Product Rating system is a system that detects hidden sentiments in comments and rates the product accordingly. The system uses sentiment analysis methodology in order to achieve desired functionality. This project is an E-Commerce web application where the registered user will view the product and product features and will comment about the product. System will analyze the comments of various users and will rank product. We use a database of sentimentbased keywords along with positivity or negativity weight in database and then based on these sentiment keywords mined in user comment is ranked. Comment will be analyzed by comparing the comment with the keywords stored in database. The System takes comments of various users, based on the comment, system will specify whether the product is good, bad, or worst. Once user login the system he can view the product and product features. After viewing product user can comment about the product. User can also view comment of another users. The role of the admin is to add product to the system and to add keywords in database. User can easily find out correct product for his usage. This application also works as an advertisement which makes many people aware about the product. This system is also useful for the users who need review about a product.

1.2 Problem Statement

The ever-increasing use of e-commerce has produced vast consumer generated contents such as reviews. It is becoming imperative for organizations as well as consumers to collect these data for product analysis or rating according to user needs.

1.3 Objectives

- ✧ To implement an algorithm for automatic classification of text as positive or negative.
- ✧ To provide people the facility of giving only reviews of a product and avoid the double work of rating as well as review.

- ✦ To rate the products based on the weight age of the keywords in database, so that the result is appropriate. Reviews of user will contribute to generate overall rating of the product.

1.4 System Features

The main feature of our web application is to determine the rating of products in an ecommerce website by analyzing the user reviews. Our system is capable of determining new reviews taking reference to previously trained reviews.

1.5 System Specifications

1.5.1 Hardware Specifications

Processor	:	Dual core
Processor speed	:	2.5 Ghz
RAM	:	4 GB
Hard disk drive	:	500 GB
CD ROM Drive	:	Sony
Monitor	:	18.5 inches
Keyboard	:	Logitech

1.5.2 Software Specifications

Operating system	:	Windows 7 or above
Technology used	:	HTML/CSS
Browser	:	Google chrome
IDE	:	ANACONDA Jupyter Note Book
Database	:	MySQL

1.6 Challenges

Some of the major challenges in sentiment analysis

- ✦ The comments given by user for a product is considered positive at one situation and negative at another situation.
- ✦ Some people don't express opinions in the same way.
- ✦ Most reviews will have both positive and negative comments, which somewhat manageable by analyzing sentences one at a time.
- ✦ Sometimes people may give fake comments about the product, which gives the bad review about the product.
- ✦ The sentiment analysis problem can be sometimes managed by manual methods.

2. LITERATURE REVIEW

2.1 History

Sentiment analysis is a rapidly emerging domain in the area of research in the field of Natural Language Processing (NLP). It has gained much attention in recent years. Sentiment classification is used to verify or analyze the comments given by the user to extract the opinion from it. Sentiment analysis is a machine learning approach in which machines classify and analyze the human's sentiments, emotions, opinions etc. about the products which are expressed in the form of text, star rating, thumbs up and thumbs down. The data used in this study is online product reviews collected from the sample website that we have created. Words such as adjectives and adverbs are able to convey opposite sentiment with the help of negative prefixes. Negation phrase identification algorithm is used to find such words. The performance is evaluated through evaluation measures. At last, we also give insight into our future work on sentiment analysis.

2.2 Deep Learning for NLP

Natural Language Processing is all about creating systems that process or "understand" language in order to perform certain tasks. These tasks could include:

- ‡ **Question Answering** - The main job of technologies like Siri, Alexa, and Cortana
- ‡ **Sentiment Analysis** - Determining the emotional tone behind a piece of text
- ‡ **Image to Text Mappings** - Generating a caption for an input image
- ‡ **Machine Translation** - Translating a paragraph of text to another language **Speech Recognition** - Having computers recognize spoken words

In the pre-deep learning era, NLP was a thriving field that saw lots of different advancements. However, in all of the successes in the aforementioned tasks, one needed to do a lot of feature engineering and thus had to have a lot of domain knowledge in linguistics. Entire 4year degrees are devoted to this field of study, as practitioners needed to be comfortable with terms like phonemes and morphemes. In the past few years, deep learning has seen incredible progress and has largely removed the requirement of strong domain knowledge. As a result of the lower barrier to entry, applications to NLP tasks have been one of the biggest areas of deep learning research.

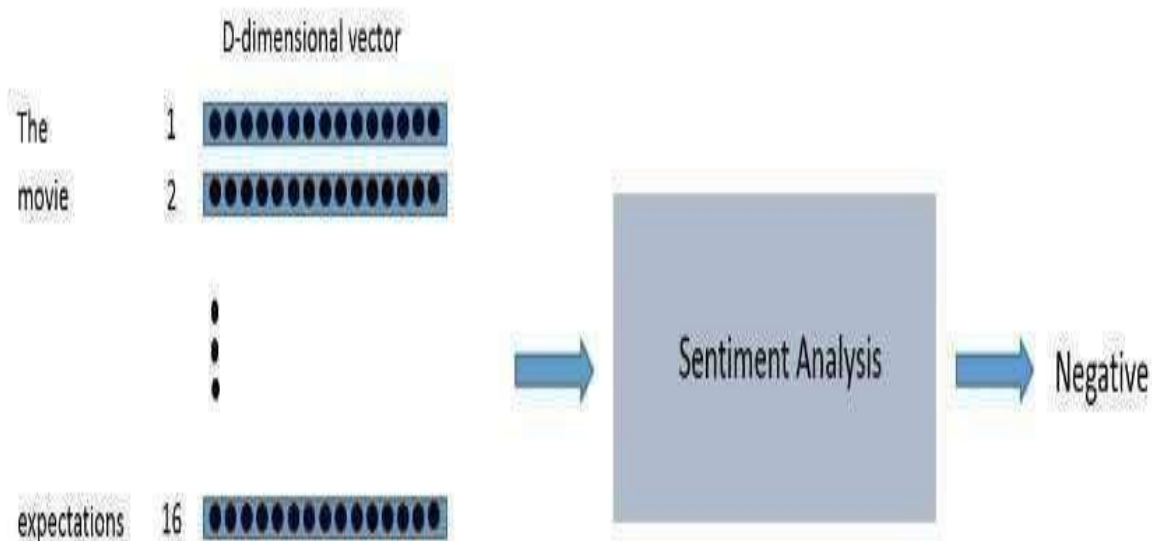
3. METHODOLOGY

3.1 Word Vector

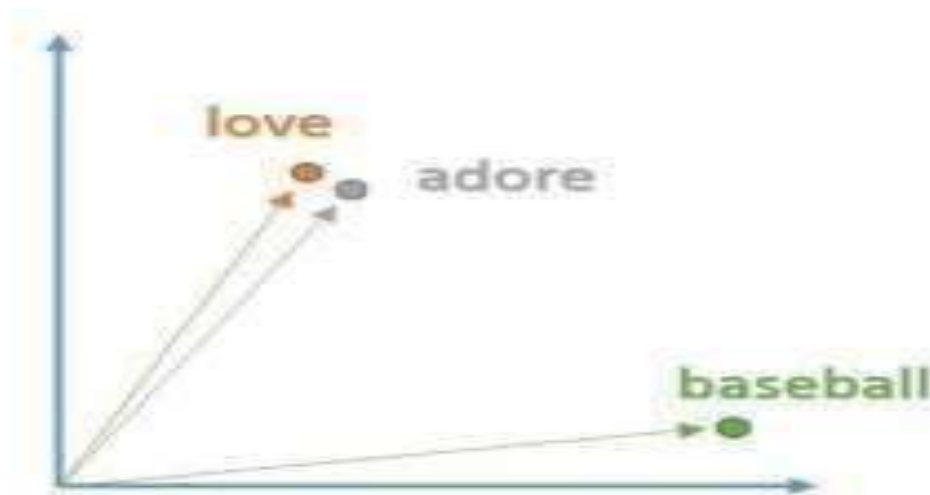
In order to understand how deep learning can be applied, think about all the different forms of data that are used as inputs into machine learning or deep learning models. Convolutional neural networks use arrays of pixel values, logistic regression uses quantifiable features, and reinforcement learning models use reward signals. The common theme is that the inputs need to be scalar values, or matrices of scalar values.

There is no way for us to do common operations like dot products or backpropagation on a single string. Instead of having a string input, we will need to convert each word in the sentence to a vector.

You can think of the input to the sentiment analysis module as being a $16 \times D$ dimensional

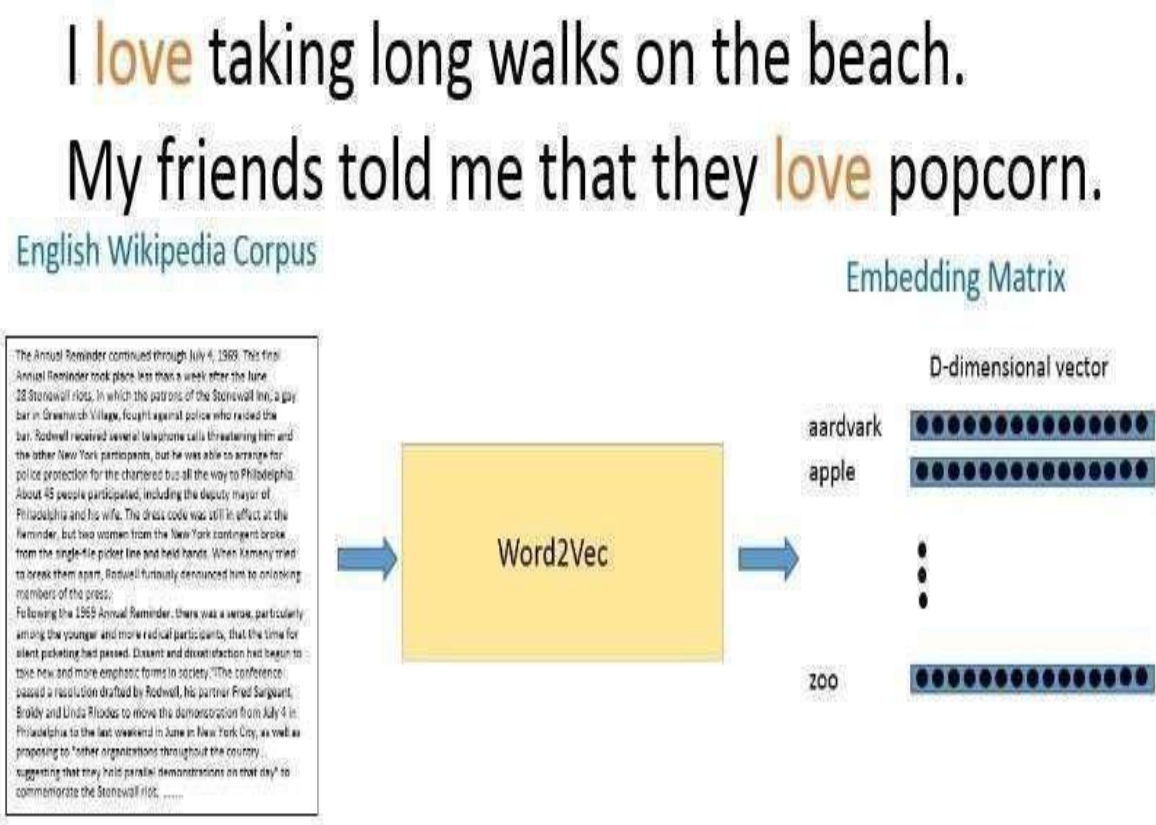


matrix. We want these vectors to be created in such a way that they somehow represent the word and its context, meaning, and semantics. For example, we'd like the vectors for the words "love" and "adore" to reside in relatively the same area in the vector space since they both have similar definitions and are both used in similar contexts. The vector representation of a word is also known as a word embedding.



Word2Vec: In order to create these word embeddings, we'll use a model that's commonly referred to as "Word2Vec". Without going into too much detail, the model creates word vectors by looking at the context with which words appear in sentences. Words with similar contexts will be placed close together in the vector space. In natural language, the context of words can be very important when trying to determine their meanings. Taking our previous example of

the words "adore" and "love", consider the types of sentences we'd find these words in. From the context of the sentences, we can see that both words are generally used in sentences with positive connotations and generally precede nouns or noun phrases. This is an indication that both words have something in common and can possibly be synonyms. Context



It is also very important when considering grammatical structure in sentences. Most sentences will follow traditional paradigms of having verbs follow nouns, adjectives precede nouns, and so on. For this reason, the model is more likely to position nouns in the same general area as other nouns. The model takes in a large dataset of sentences (English Wikipedia for example) and outputs vectors for each unique word in the corpus. The output of a Word2Vec model is called an embedding matrix.

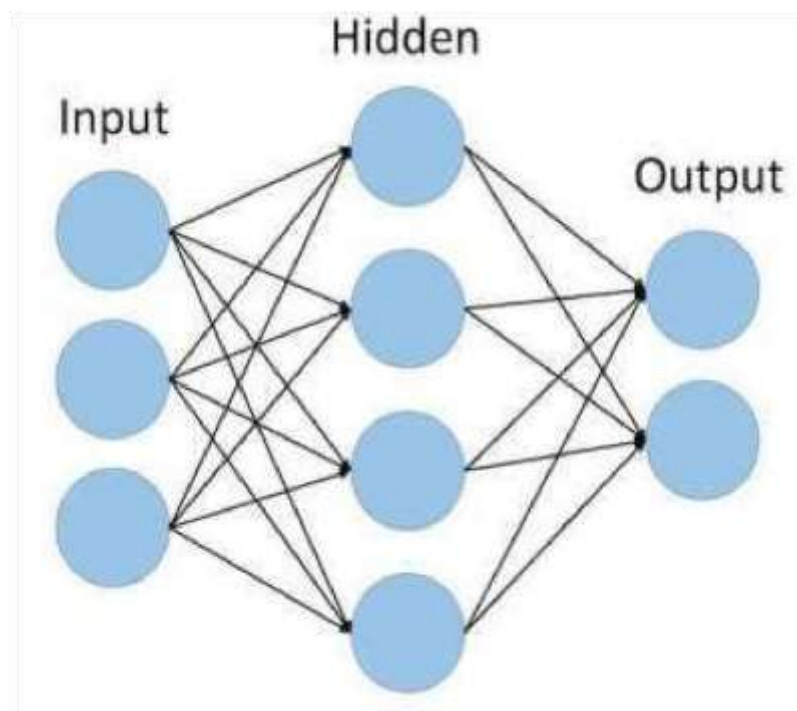
This embedding matrix will contain vectors for every distinct word in the training corpus. Traditionally, embedding matrices can contain over 3 million word vectors.

The Word2Vec model is trained by taking each sentence in the dataset, sliding a window of fixed size over it, and trying to predict the center word of the window, given the other words. Using a loss function and optimization procedure, the model generates vectors for each unique word.

3.2 Recurrent Neural Networks (RNNs)

Now that we have our word vectors as input, let's look at the actual network architecture we're going to be building. The unique aspect of NLP data is that there is a temporal aspect to it. Each word in a sentence depends greatly on what came before and comes after it. In order to account for this dependency, we use a recurrent neural network.

The recurrent neural network structure is a little different from the traditional feedforward NN you may be accustomed to seeing. The feedforward network consists of input nodes, hidden units, and output nodes.



The main difference between feedforward neural networks and recurrent ones is the temporal aspect of the latter. In RNNs, each word in an input sequence will be associated with

The movie was ... expectations

x_0 x_1 x_2 ... x_{15}
 $t = 0$ $t = 1$ $t = 2$ $t = 15$

a specific time

step. In effect, the number of time steps will be equal to the max sequence length.

Associated with each time step is also a new component called a hidden state vector h_t .

From a high level, this vector seeks to encapsulate and summarize all of the information that was seen in the previous time steps. Just like x_t is a vector that encapsulates all the information of a specific word, h_t is a vector that summarizes information from previous time steps.

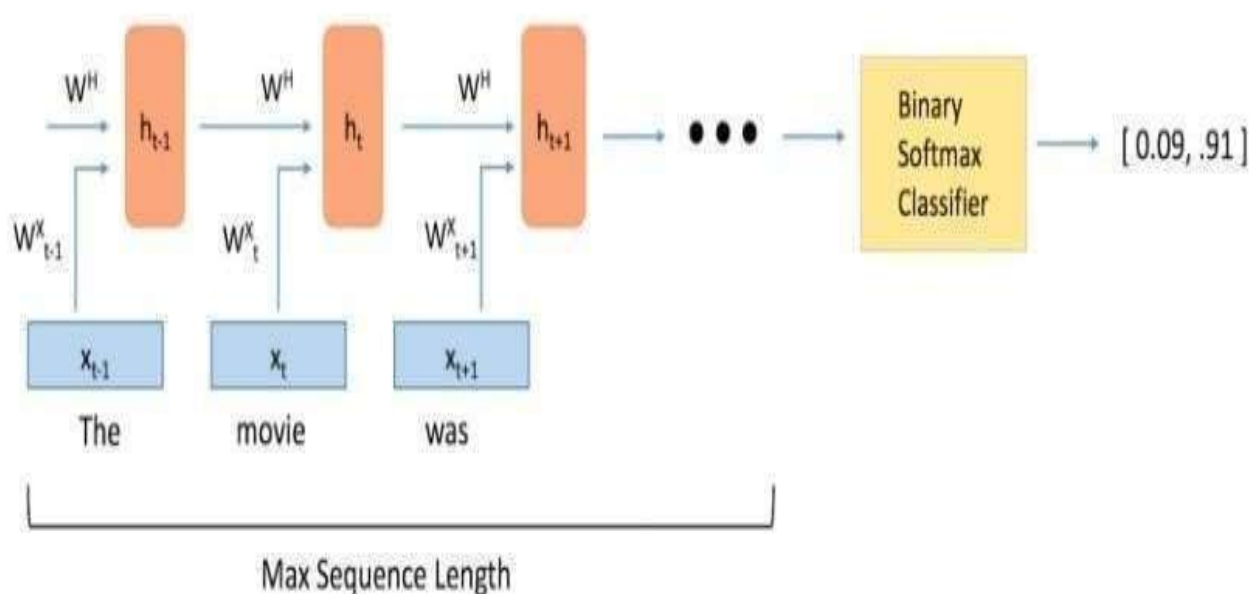
$$h_t = \sigma(W^H h_{t-1} + W^X x_t)$$

The hidden state is a function of both the current word vector and the hidden state vector at the previous time step. The sigma indicates that the sum of the two terms will be put through an activation function (normally a sigmoid or RELU).

The 2 W terms in the above formulation represent weight matrices. If you take a close look at the superscripts, you'll see that there's a weight matrix W^X which we're going to multiply with our input, and there's a recurrent weight matrix W^H which is multiplied with the hidden state vector at the previous time step. W^H is a matrix that stays the same across all time steps, and the weight matrix W^X is different for each input.

The weight matrices are updated through an optimization process called backpropagation through time.

The hidden state vector at the final time step is fed into a binary SoftMax classifier where it is multiplied by another weight matrix and put through a SoftMax function that outputs values between 0 and 1, effectively giving us the probabilities of positive and negative sentiment.



3.3 Hyperparameter Tuning

Choosing the right values for your hyperparameters is a crucial part of training deep neural networks effectively. We found that our training loss curves can vary with our choice of optimizer (Adam, Adadelta, SGD, etc), learning rate, and network architecture. With RNNs and LSTMs in particular, some other important factors include the number of LSTM units and the size of the word vectors.

Learning Rate: RNNs are infamous for being difficult to train because of the large number of time steps they have. Learning rate becomes extremely important since we don't want our weight values to fluctuate wildly as a result of a large learning rate, nor do we want a slow training process due to a low learning rate. The default value of 0.001 is a good place to start. You should increase this value if the training loss is changing very slowly, and decrease if the loss is unstable.

Optimizer: There isn't a consensus choice among researchers, but Adam has been widely popular due to having the adaptive learning rate property (Keep in mind that optimal learning rates can differ with the choice of optimizer).

Number of LSTM units: This value is largely dependent on the average length of your input texts. While a greater number of units provides more expressibility for the model and allows the model to store more information for longer texts, the network will take longer to train and will be computationally expensive.

Word Vector Size: Dimensions for word vectors generally range from 50 to 300. A larger size means that the vector is able to encapsulate more information about the word, but you should also expect a more computationally expensive model.

3.4 Implementation Details

The dataset applicable for our project were generated from the data set of following projects done in Kaggle:

- 400K text reviews and ratings for training and testing

- 3.6M text reviews and ratings for training

'__label__2 Great CD: My lovely Pat has one of the GREAT voices of her generation. I have listened to this CD for YEARS and I still LOVE IT. When I\'m in a good mood it makes

me feel better. A bad mood just evaporates like sugar in the rain. This CD just oozes LIFE. Vocals are jusat STUUNNING and lyrics just kill. One of life\'s hidden gems. This is a desert isle CD in my book. Why she never made it big is just beyond me. Everytime I play this, no matter black, white, young, old, male, female EVERYBODY says one thing "Who was that singing ?"\n

3.5 Data Cleaning and Preprocessing

At first, we have splitted our dataset into train sentences and labels.

Also, we converted to lowercase.

And finally we removed some words in sentences like “http”, www., “.com”,etc.

Train sentences: 'great cd: my lovely pat has one of the great voices of her generation. i have listened to this cd for years and i still love it. when i\'m in a good mood it makes me feel better. a bad mood just evaporates like sugar in the rain. this cd just oozes life. vocals are jusat stuunning and lyrics just kill. one of life\'s hidden gems. this is a desert isle cd in my book. why she never made it big is just beyond me. everytime i play this, no matter black, white, young, old, male, female everybody says one thing "who was that singing?"'

Label: 1

Texts_to_sequences: Transform each text in corpus in a sequence of integers

Pad_sequences: This function transforms a list of num_samples sequences (lists of integers) into a matrix of shape (num_samples, num_timesteps). num_timesteps is either the maxlenargument if provided, or the length of the longest sequence otherwise.Sequences that are shorter than num_timesteps are padded with value at the end.

Sequences longer than num_timesteps are truncated so that they fit the desired length. The position where padding or truncation happens is determined by the arguments padding and truncating, respectively.Pre-padding is the default.

3.6 Model visualization:

Layer (type)	Output Shape	Param #
=====		
embedding_1 (Embedding)	(None, None, 100)	1000000

lstm_1 (LSTM)	(None, None, 256)	365568

lstm_2 (LSTM)	(None, 512)	1574912

dense_1 (Dense)	(None, 500)	256500

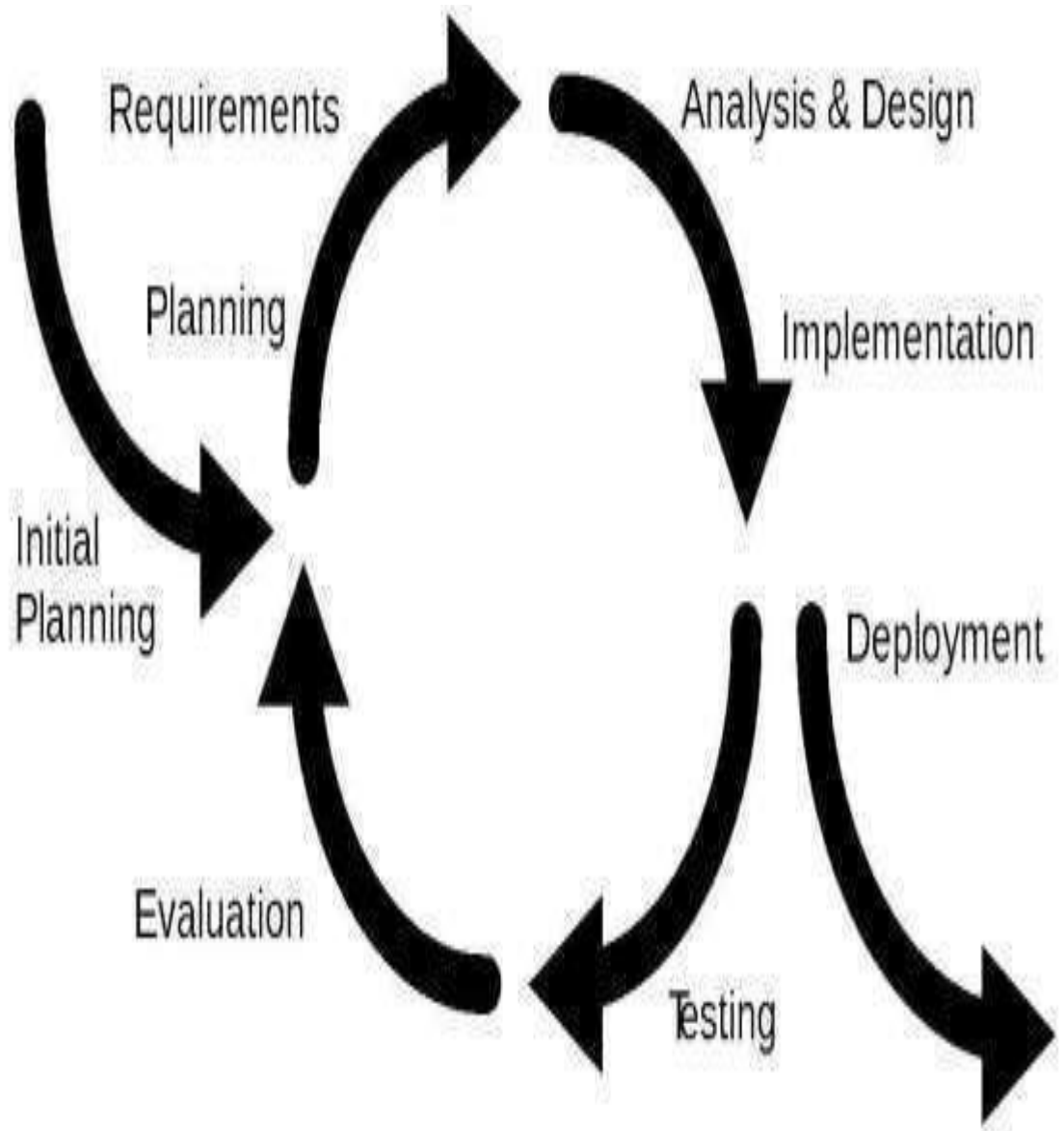
dropout_1 (Dropout)	(None, 500)	0

dense_2 (Dense)	(None, 100)	50100

dropout_2 (Dropout)	(None, 100)	0

dense_3 (Dense)	(None, 1)	101
=====		
Total params: 3,247,181		
Trainable params: 3,247,181		
Non-trainable params: 0		

4. System Development



For the development of our system, we chose the iterative and incremental model of Software Development. The software development lifecycle follows a pattern of planning, developing, testing and rebuilding over and over to evolve the system through several stable intermediates until a final finished product is achieved. The development process we used was also focused on developing rapid prototypes over the entire cycle to verify the system outputs to the actual software requirements.

The following were the main phases that we followed for our overall system design procedure:

- **Initial Planning:** In the foremost stage, we analyzed the feasibility of the project based on several factors including requirements of our course of study, our domain expertise and knowledge, literature reviews and scope analysis, availability of datasets, practicality of concept and validated it for approval by the proper guidance of our project supervisor.
- **System plan:** Once the project approval was received, we developed a concrete plan for development of the system. To begin with, we started analyzing the use cases and several scenarios possible in the system and developed every possible alternative to the main success scenario that could likely occur in the system.
- **Analysis and Design:** We started with the designing of system models and designs for proper visualization and understanding of tasks to be completed in each iterative cycle.
- **Implementation:** Having made the system models, we develop the codes and equivalent program sections required for the implementation of different parts of the system. The developed system code was also compiled to create a prototype of the system to depict the current output.
- **Testing:** The developed prototype was then tested after implementation to check whether the goal of the iteration was achieved or not.
- **Evaluation:** The system was revised again and again with modifications at the time of the testing phase until the system gradually evolved into a system with desired specifications.

5. SYSTEM DESIGN AND ARCHITECTURE

5.1 Flowchart

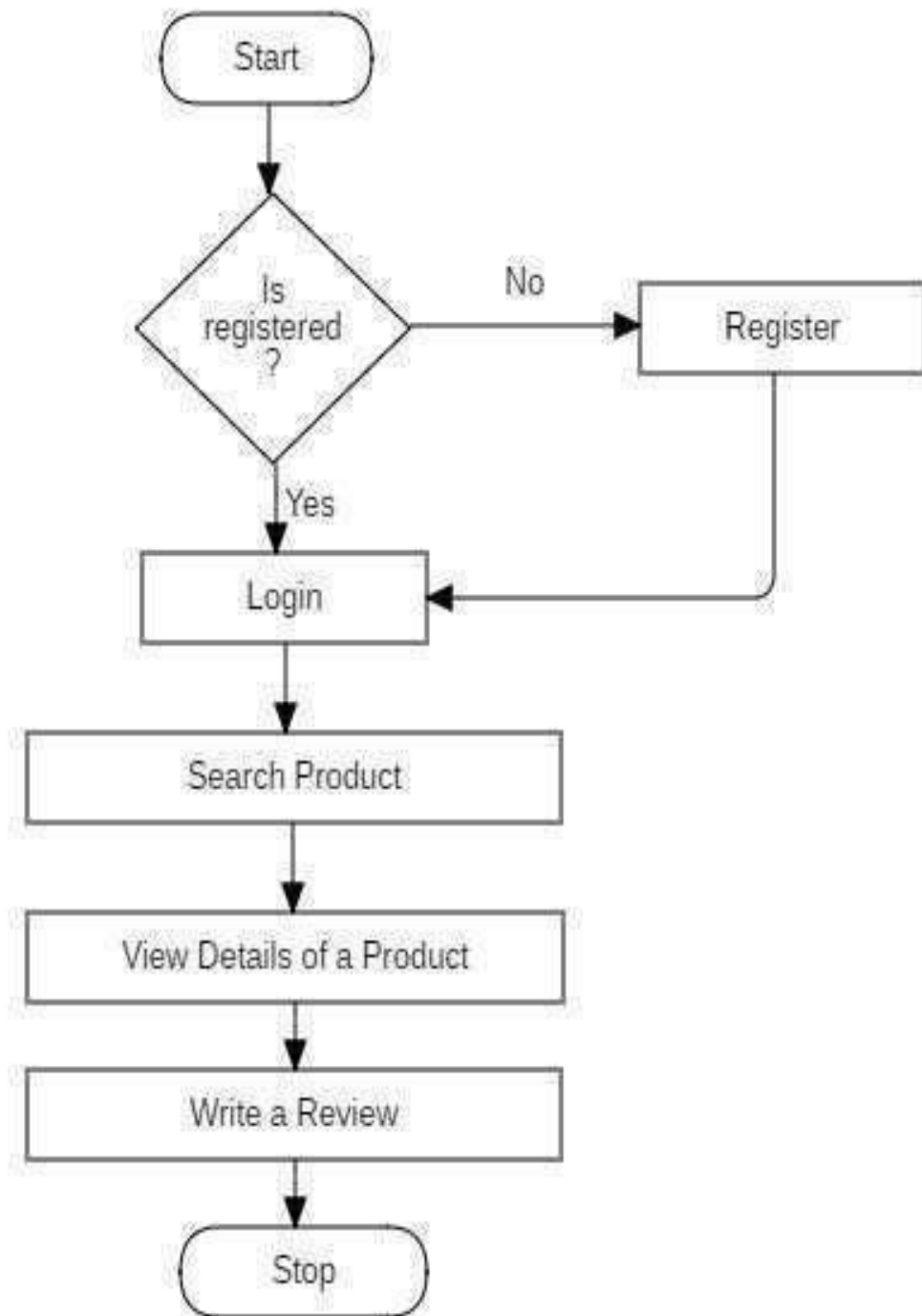


Fig 1: Typical flow of the Product Rating syste

5.2 UML Diagrams

5.2.1 Use Case Diagram

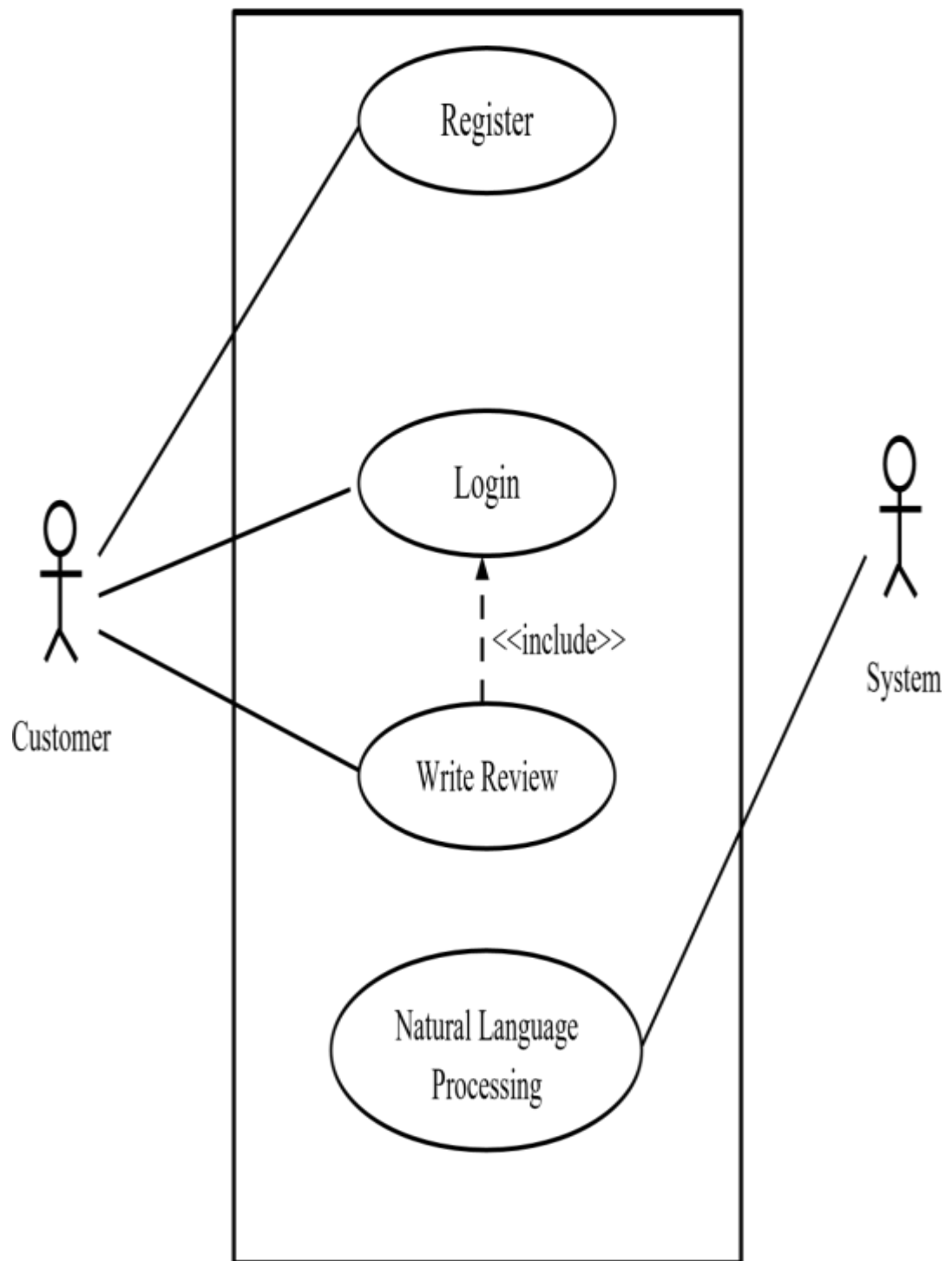


Fig 2: Use case Diagram for Product Rating System

5.2.2 Context Diagram

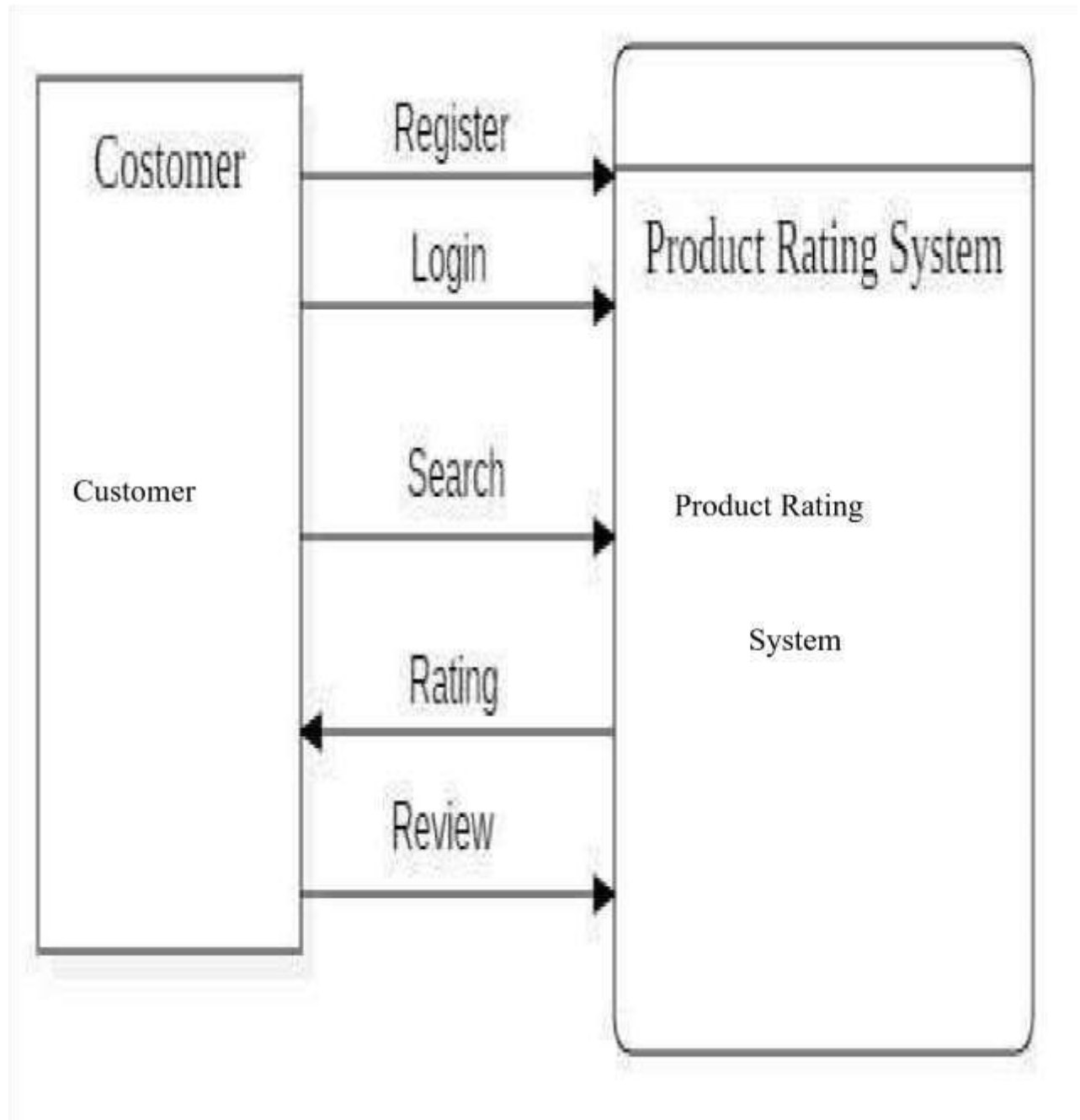


Fig 3: Level 0 Data Flow Diagram (DFD) for Product Rating System

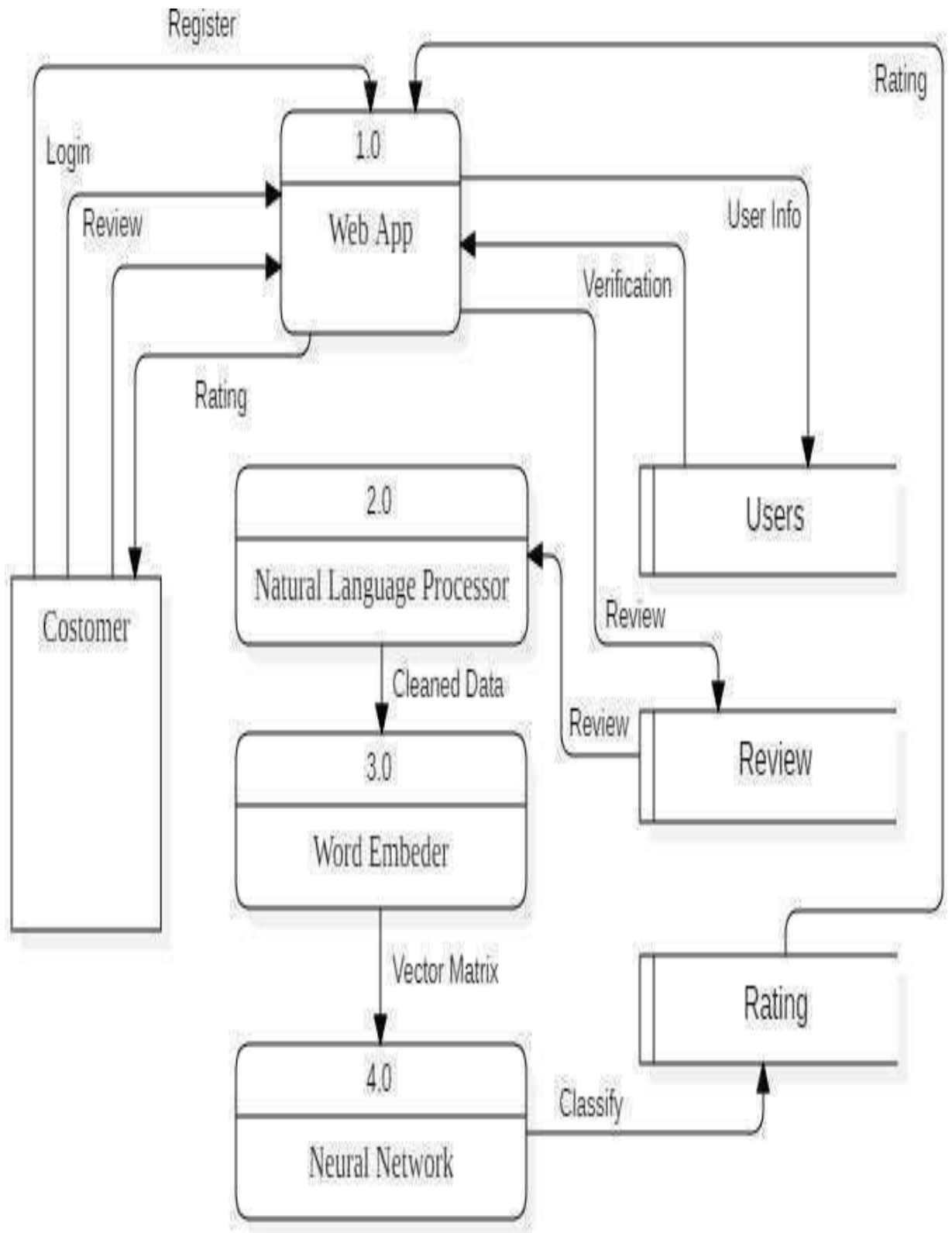


Fig 4: Level 1 Data Flow Diagram (DFD) for Product Rating System

5.2.3 Sequence Diagram

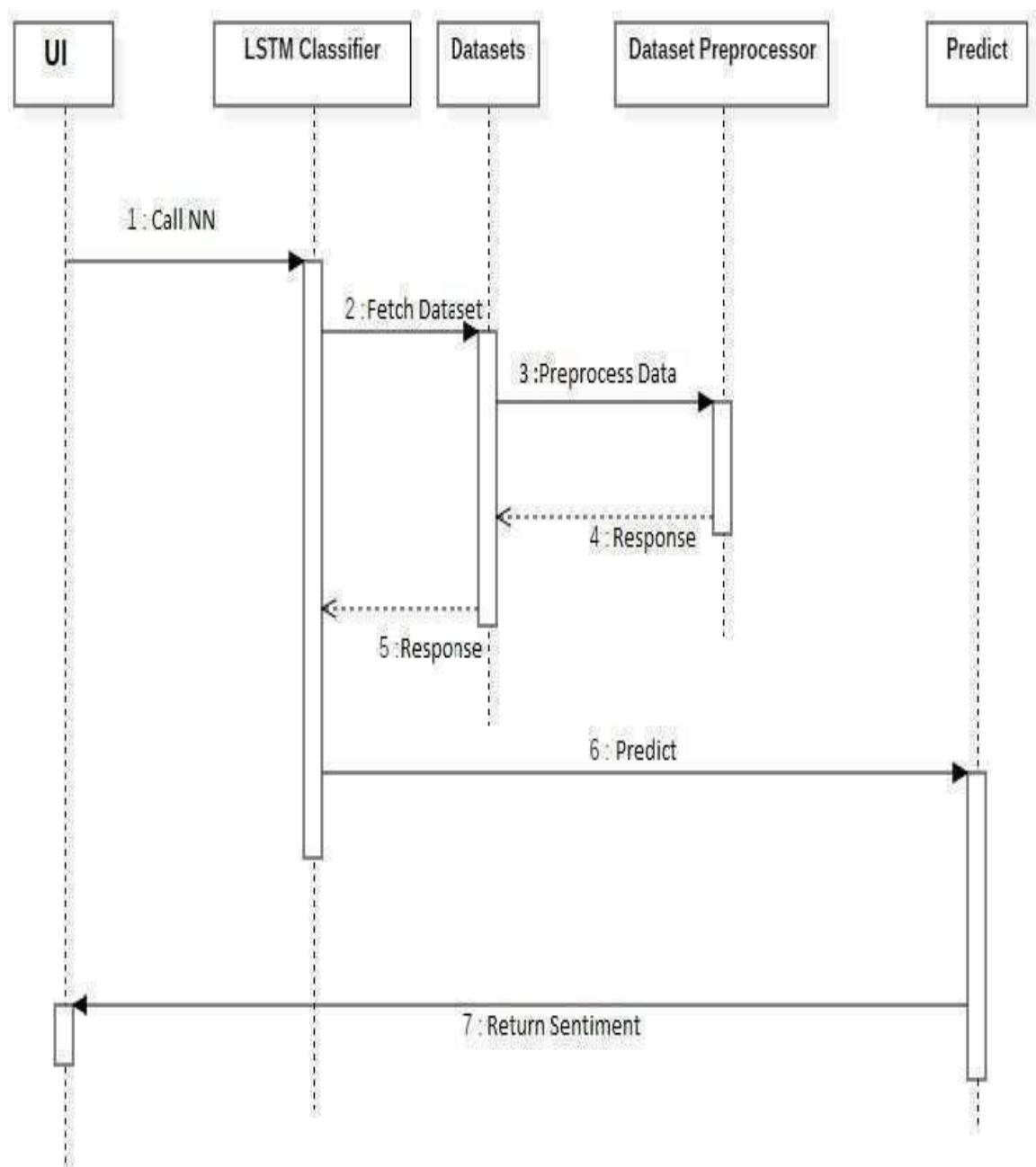


Fig 5: Sequence Diagram for Product Rating System

5.2.4 Class Diagram

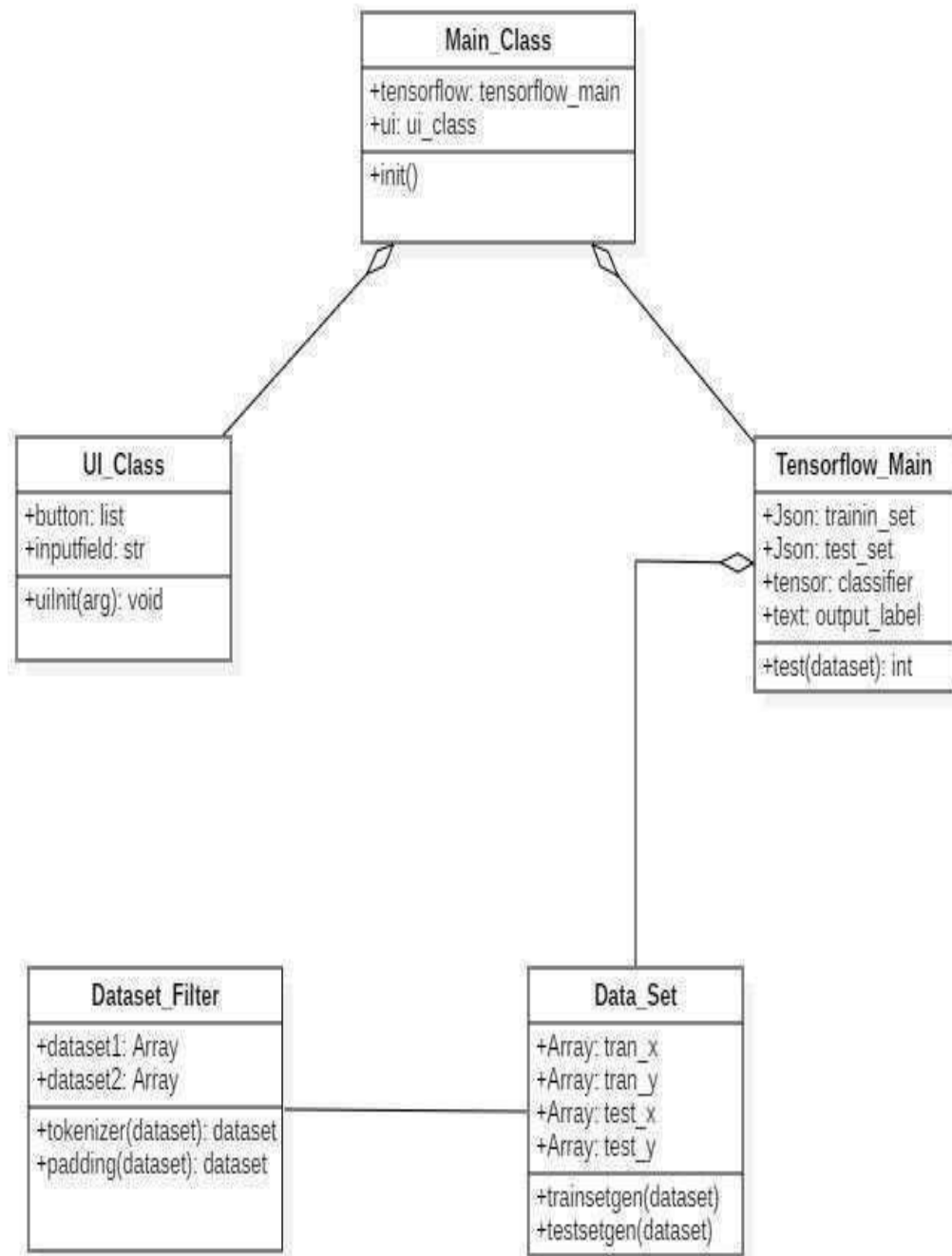


Fig 6: Class Diagram for Product Rating System

5.3 Block Diagram

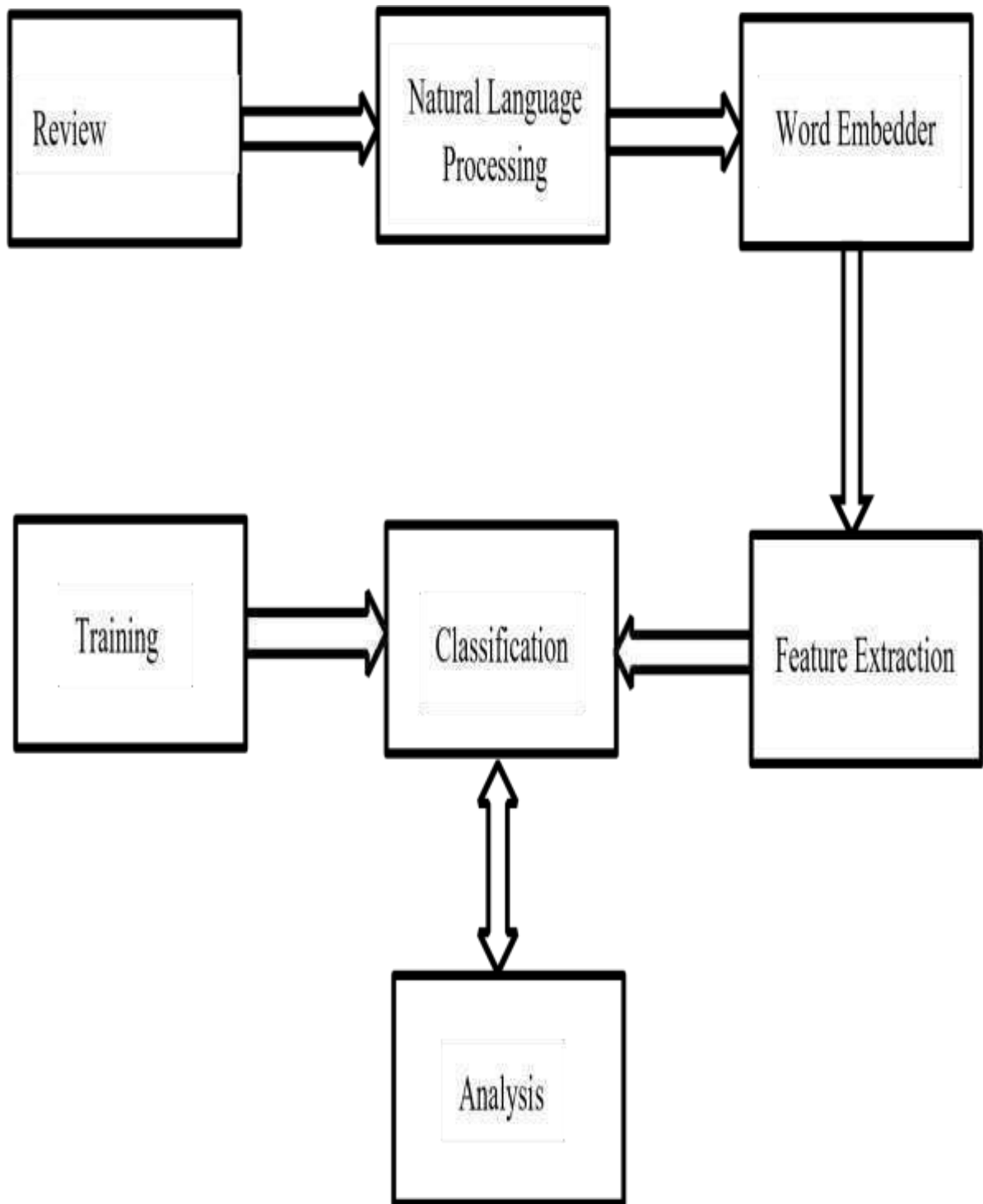


Fig 7: Block Diagram for Product Rating System

6..PROGRAM

```
#!/usr/bin/env python # coding: utf-8
```

```
import numpy as np # linear algebra import pandas as pd # data
processing, CSV file I/O (e.g. pd.read_csv) import os import bz2
import re from tqdm import tqdm import tensorflow as tf from
sklearn.utils import shuffle from matplotlib import pyplot as plt from
keras.preprocessing.text import Tokenizer from
keras.preprocessing.sequence import pad_sequences from
keras.models import Sequential from keras.layers import
Embedding,LSTM,Dropout,Dense from keras.callbacks import
EarlyStopping, ModelCheckpoint from nltk.corpus import stopwords
from sklearn.model_selection import train_test_split from
sklearn.preprocessing import LabelBinarizer from keras.models
import load_model from keras import backend as K
```

```
# In[2]: Load Model
```

```
model = load_model('LSTMmodel.h5') print("model loaded")
```

```
# In[3]: Train Model
```

```
train_file = bz2.BZ2File('test.ft.txt.bz2')
```

```
# In[4]:
```

```
train_file_lines = train_file.readlines() train_file_lines = [x.decode('utf-8') for x in train_file_lines]
train_labels = [0 if x.split(' ')[0] == '__label__1' else 1 for x in train_file_lines] train_sentences =
[x.split(' ', 1)[1][:1].lower() for x in train_file_lines] for i in range(len(train_sentences)):
train_sentences[i] = re.sub('\d','0',train_sentences[i])
```

```
for i in range(len(train_sentences)): if 'www.' in train_sentences[i] or 'http:' in
train_sentences[i] or 'https:' in train_sentences[i] or '.com' in train_sentences[i]:
train_sentences[i] = re.sub(r"([^\ ]+(?<=\.[a-z]{3}))", "<url>", train_sentences[i])
X_train,X_test,y_train,y_test=train_test_split(train_sentences,train_labels,train_size=0.80,test_s
ize=0.20,random_state=42) tokenizer = Tokenizer(num_words=10000)
tokenizer.fit_on_texts(X_train)
```

```
#In[5]: Find the rating
```

```
def rate(p):
return (p*5)
```

```
# In[6]:
```

```
#!C:\Users\xitis\AppData\Local\Programs\Python\Python37-32\python.exe import sys from  
flask import Flask, render_template, url_for,json, request, jsonify, redirect import  
MySQLdb
```

```
# In[7]:
```

```
app = Flask(__name__)
```

```
conn = MySQLdb.connect(host= "localhost",          user="root",  
passwd="pukar11",          db="rating") c = conn.cursor()
```

```
# In[8]:
```

```
@app.route("/") def  
hello():  
    return render_template('index.html')
```

```
@app.route("/jacket", methods=['GET']) def  
jacket():  
    conn = MySQLdb.connect(host= "localhost",          user="root",  
passwd="pukar11",          db="rating") c = conn.cursor()
```

```
#Fetch the reviews and their corresponding ratings for a product c.execute("SELECT *  
FROM jacket ORDER BY timeadded DESC")
```

```
data = c.fetchall()
```

```
#Fetch the overall rating of a product c.execute("SELECT * FROM product")
```

```
stars = c.fetchone()
```

```
return render_template('jacket.html', data=data, stars = stars)
```

```
# In[9]:
```

```
@app.route("/postreview", methods=['POST']) def  
postreview():  
    #Get the data posted from the form message  
= request.get_json(force=True) review =  
message['review'] uname = message['uname']
```

```
#assign the review text to a variable a=[review]
```

```

#predict the outcome
pred=model.predict(pad_sequences(tokenizer.texts_to_sequences(a),maxlen=100))
value=rate(pred.item(0,0))

#Save the review, username and predicted value to the database    conn =
MySQLdb.connect(host= "localhost",          user="root",          passwd="pukar11",
db="rating")    c = conn.cursor()

c.execute(      "''''INSERT INTO jacket (uname, review, rate)      VALUES (%s, %s,
%s)''''",(uname, review , value ))

conn.commit()    #Calculate average of the ratings including the recent value
c.execute("''''SELECT AVG(rate) FROM jacket''''")    avg = c.fetchone()

#Update the overall rating of a product in database

c.execute("''''UPDATE product SET rating = %s WHERE id = %s''''",(avg,1))

conn.commit()

return

if __name__ == "__main__"
:
app.run()

```

7. RESULT

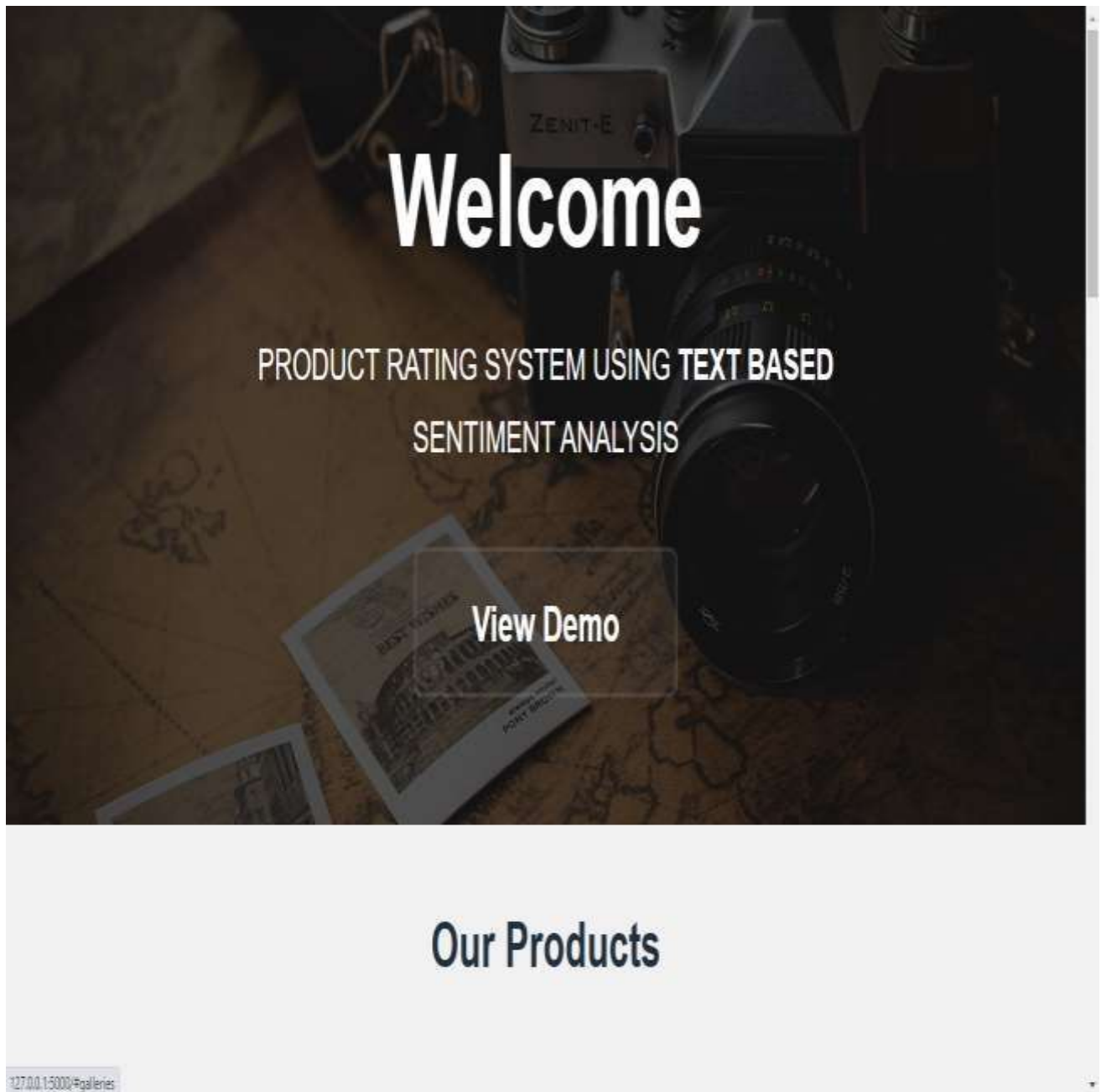




Fig 8 : Welcome demo of intro


Our Products




Men's Denim Jacket
Rs. 4,500 | 3.5 Stars




Men's Denim Jacket
Rs. 4,500 | 3.5 Stars




Men's Denim Jacket
Rs. 4,500 | 3.5 Stars




Men's Denim Jacket
Rs. 4,500 | 3.5 Stars




Men's Denim Jacket
Rs. 4,500 | 3.5 Stars



Men's Denim Jacket
Rs. 4,500 | 3.5 Stars



Men's Denim Jacket
Rs. 4,500 | 3.5 Stars



Men's Denim Jacket
Rs. 4,500 | 3.5 Stars

127.0.0.1:5000/jacket

Fig 9 : Price of a product with stars

Product Rating System



Men's Denim Jacket

Rs. 4,500

3.41 Stars

Your Name

Your Review

Write a Review

Tell us how you feel about the product.

Submit

The product rating is generated based on the review you provide

Previous Reviews

Dependra 4.41

This is awesome one

Pukar 1.55

I did not like it at all


Rajesh 4.26

This jacket colour is good and I liked its look.

© Project Team | Pukar Ghosh | Rajesh Pandey | Dependra Ramte | Prathiv Pawar

Fig 10 : Rating of a product with three reviews

Product Rating System



Men's Denim Jacket

Rs. 4,500

2.79 Stars

Your Name

Your Review

Write a Review

Tell us how you feel about the product.

Submit

The product rating is generated based on the review you provide

Previous Reviews

Prativa 0.95

Worst jacket I have ever seen.

Dependra 4.41

This is awesome one.

Pukar 1.33

I did not like it at all

Rajesh 4.26

This jacket colour is good and I liked its look.

© Project Team : Pukar Gillwala | Rajesh Parshay | Dependra Kanchal | Prativa Poudyal

Fig 11 : Rating of a product changes with arrival of new reviews.

8. REFERENCES

1. Popesu Ana-Maria, Oren Etzioni, "Extracting Product Features and Opinions from Reviews", ACL, pp. 339-346, October 2005.
2. Chhaya Chauhan, Smriti Sehgal, "Sentiment analysis on product reviews", Computing Communication and Automation (ICCCA) 2017 International Conference on, pp. 26-31, 2017.
3. K L Santhosh Kumar, Jayanti Desai, Jharna Majumdar, "Opinion mining and sentiment analysis on online customer review", Computational Intelligence and Computing Research (ICCIC) 2016 IEEE International Conference on, pp. 1-4, 2016.
4. Bird, Klein, and Edward Loper. Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. San Diego, CA: O'Reilly Media, June 2009.