UNIVERSITY OF CALCUTTA

Asutosh College

(Department of Computer Science)

Major Project Report On

MOVIE RECOMMENDATION SYSTEM USING CONTENT BASED FILTERING AND SENTIMENT ANALYSIS

By

Abhishek Bhakta

Reg. No: 223-1122-0156-13 Roll: CIS/17/04 No:**001**

Arup Das

Reg. No :544-1122-0626-14 Roll: CIS/17/04 No:**005**

Manu Gond

Reg. No : 611-1121-0028-14 Roll: CIS/17/04 No :**013**

Under the supervision of

Prof. Gautam Mahapatra
Associate Professor

Prof. Atrayee Chatterjee
Faculty Member

Preface

Computer Science is an application oriented subject. Major Project is a mandatory part of Masters in Computer Science course. Every project comes with a time bound but most of the project requires more time than the scheduled time. So there is a trade-off between time and completeness of the project. Also there are some resource limitations as all the required resources for such project are generally not readily available in an academic environment. A project requires guidance from faculties and since their availability is not always ensured, so there is a limitation of guidance. This project deals with sentiment analysis on short text type data and implementing a recommendation system with help of sentiment analysis. The sentiment analysis is extremely useful in social media monitoring as it allows us to gain an overview of the wider public opinion behind certain topics. The applications of sentiment analysis are broad and powerful. The ability to extract insights from social data is a practice that is being widely adopted by organizations across the world. Recommendation systems are used to give the users individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting objects in a larger space of possible options. Thus creating more richer user experience of product exposure from a source.We have tried our level best to add completeness to this project by guidance of our mentors. Project work helps us to gain real world experiences which are useful for future in professional fields.

Contents

1. Introduction

The opinions of others have a significant influence in our daily decision-making process. These decisions range from buying a product such as a smart phone to making investments to choosing a school—all decisions that affect various aspects of our daily life. Before the Internet, people would seek opinions on products and services from sources such as friends, relatives, or consumer reports.

Similarly, organizations use surveys, opinion polls, and social media as a mechanism to obtain feedback on their products and services. Sentiment analysis or opinion mining is the computational study of opinions, sentiments, and emotions expressed in text. The use of sentiment analysis is becoming more widely leveraged because the information it yields can result in the monetization of products and services. One place where user opinions become vital is when recommending something to user, we can use the sentiment analysis on social media reviews to get overall user ratings of certain item, which is movie in this case. Then based on recommended movies suggested to user, this generated ratings can be used to filter out the irrelevant movies which might not match user interests. This approach of using Conetnt Based filtering for movie recommendation and fusing it with sentiment analysis overcomes the disadvantage of Collabrative based recommendation which is cold start.

Using this technique we can recommend movies to user without having a large user base to use collaborative filtering. Purposed system consists of two parts, first part deals with analyzing twitter sentiment to generate the overall rating for a topic which is movie in this case. And second part uses content based recommendation and ratings from first part to generate final recommendations. However generating database of ratings for each movie is computationally costly task, and only be done in production environment. We have only generated a tool for doing that. For recommendation results of this project, a preloaded dataset having all ratings has been used rather than creating a huge dataset for movies rating from start.

1.1 Broad Topic

Deep Learning, Recurrent Neural Network, Recommendation System, Cloud Computing, Mobile Computing.

1.2 Objective

The main objective of our project is to build an **android application** that can show **movie recommended to user** based on previous interaction. Also to demonstrate that model can be used to generate **ratings database**, a **twitter sentiment analysis** option to analyze tweets for a particular topic. Thus this system can be scaled up to work on e-commerce data, and when user base is increased it can migrate to a hybrid recommendation system.

1.3 Literature Review

The core topic about this project is sentiment analysis, which has been described in detail under **section [2]**. To create a model capable of classifying sentiments on short text type data deep learning approach has been used. The basic feed forward neural network has been discussed in **section [3.1]**. The recurrent neural network and Long-Short term network has been discussed in **section [3.2-3.3]** which has been used to implement the actual system for sentiment analysis. The learning algorithm Adam model training is explained in **section [3.4]**, and implementation of sentiment analysis model in **section [4]**, which covers from data cleaning to the prediction and overall performance.

The second part of project is composed of Recommendation system. Types of recommendation system has been described in **section [5.3-5.3.3]**, with implementation of our purposed model in **section [6]** using the twitter sentiment analysis and content based filtering.

Since the whole project requires two different platforms, one for requesting the results and one for computing the results. Database schema for this project has been mentioned in **section [8]**. The first part which is platform for computation on complex data has been implemented using Google Cloud Compute Engine and has been described in **section [9.2]**. The second part, which is client-based application, has been implemented on Android OS for mobile computing devices has been discussed in **section [9.5]**. Conclusions and future scopes has been discussed in **section [11]** and the code for implementation has been added under **section [14]**.

2. Sentiment Analysis

2.1 What is sentiment analysis?

It is a process of computationally identifying and categorizing opinions from apiece of text, and determine whether thewriter's attitude towards a particular topic or the product. Sentiment Analysis is a branch of computer science, and overlaps heavily with Machine Learning, and Computational Linguistics In the last decade, sentiment analysis , also known as **opinion mining**, has attracted an increasing interest. It is a hard challenge for language technologies, and achieving good results is much more difficult than some people think. All in all, **sentiment analysis** boils down to one thing: It's the process of analyzing online pieces of writing to determine the emotional tone they carry. In simple words, **sentiment analysis** is used to find the author's attitude towards something.

2.2 Need of Sentiment Analysis

Sentiment analysis is extremely useful in social media monitoring as it allows us to gain an overview of the wider public opinion behind certain topics.

Another of the **benefits of sentiment analysis** is its ability to provide a helping hand. **Sentiment analysis** helps you keep track of the moods of any number of customers for your team. At a glance, you can see which chats are going smoothly, and which need further attention.

There are no limitations, in that many systems are very effective, and while in some domains there is less success, with the assistance of domain specific lexicons and perhaps some feature engineering. However, the specificity of polarity/prediction of fine-grained sentiment is generally not something that is a priority in many academic settings at present, and this many become more prominent due to industry demands, and will likely be a key limitation but at present this can be handled with significant preprocessing and feature engineering. Ultimately, the limitations are really dependent on the restraints you place on the degree the input can be modified for your sentiment analysis algorithm and the accuracy/specificity of the output.

3. Tools: Sentiment Analysis

Classification is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known For example would be assigning a given email into “spam” or “non-spam” classes An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. The term classifier sometimes also refers to the mathematical function, implemented by a classification algorithm that maps input data to a category.

To learn about classification using the Artificial Neural Networks we need to understand the basics as discussed in this section.

3.1 Feed Forward Neural Networks

An artificial neural network is a computational model based on the structure and functionality of a biological neuron. It consists of processing element, a number of inputs and weighted edges connecting each input to the processing element. The each processing element is called an artificial neuron [3].

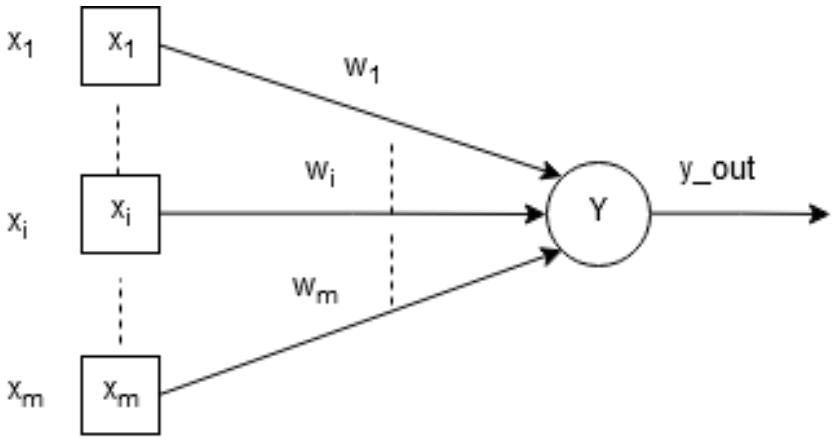


Fig 2.1: Basic overview of an artificial neuron

Artificial neuron. An artificial neuron may contain m number of input units X_1, X_2, X_3,....., X_m. and y_out is the output signal of the processing unit Y.

Notational Convention

X_i **:** The i^{th} input unit.
Y **:** The output unit, In case there are more than one output units, the j^{th} output unit is denoted as Y_j.

x_i **:** Signal to the input unit X_j. This signal is transmitted to the output unit Y scaled by the weight w_i.

w_i **:** The weight associated with the interconnection between input unit X_i and the output unit Y. In this case there are more than one output units, w_{ij} denotes the weight between input unit X_i and the j_{th} output unit Y_j.

y_in **:** The total (or net) input to the output unit Y. It is the algebraic sum of all weighted input to Y.

y_out : Signal transmitted by the output unit Y. It is known as the activation of Y.

The net input y_out to the processing element Y is obtained as

$$\boldsymbol{y\_in = x_1 w_1 + x_2 w_2 + \ldots\ldots + x_m w_m = \sum_{i=1}^{m} x_i w_i} \quad \text{----------(1)}$$

If there are more than one output units, then the net input to the jth output unit Y_j, denoted as y_in_j, can be derived from equation 1 as follows:

$$\boldsymbol{y\_in_j = x_1 w_{1j} + x_2 w_{2j} + \ldots\ldots + x_m w_{mj} = \sum_{i=1}^{m} x_i w_{ij}} \quad \text{----------(2)}$$

The weight w_i associated with the input X_i, may be positive or negative. A positive weight w_i means the corresponding input X_i has an excitatory effect on Y. If, however w_i is negative then the input X_i is said to have an in inhibitory effect on Y. The output signal transmitted by Y is a function of the net input y_in, Hence,

$$\boldsymbol{y_{out} = f(y_{in})} \quad \text{---------- (3)}$$

In its simplest form f(.) is a step function. A binary step function for the output unit is defined as

$$\boldsymbol{y_{out} = f(y_{in}) = f(x) = \begin{cases} 1, & if\ y\_in > 0 \\ x, & if\ y_{in} \leq 0 \end{cases}} \quad \text{----------(4)}$$

When a non-zero threshold θ is used then the above equation takes form

$$\boldsymbol{y_{out} = f(y_{in}) = f(x) = \begin{cases} 1, & if\ y\_in > \theta \\ x, & if\ y_{in} \leq \theta \end{cases}} \quad \text{----------(5)}$$

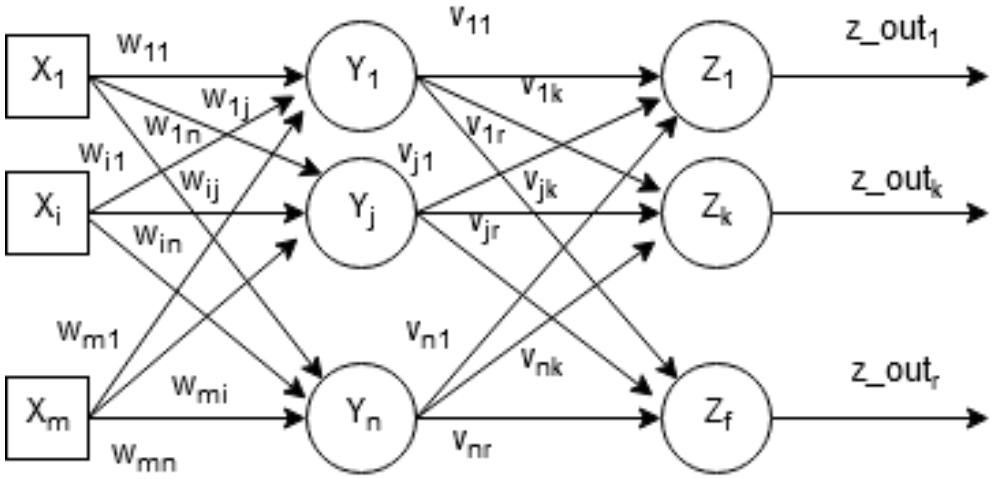


Fig 2.2: Basic overview of a feed forward neural network

In a multilayer feed forward neural network has one or more hidden layer between input and output layer. The expression s for the net inputs to the hidden layer units and the output units are obtained as

$$\boldsymbol{y\_in = X \times W} \qquad \text{-------(6)}$$

$$\boldsymbol{Z\_in = Y\_out \times V} \qquad \text{-------(7)}$$

Where,

$X = [x_1, x_2, \ldots, x_m]$, X is input vector,

$Y\_in = [y\_in_1, y\_in_2, \ldots, y\_in_n]$, is the net input vector to the hidden layer,

$Z\_in = [z\_in_1, z\_in_2, \ldots, z\_in_r]$, is the net input vector to the output layer,

$Y\_out = [y\_out_1, y\_out_2, \ldots, y\_out_n]$, is the output vector from the hidden layer,

W and V are the weight metrices for the interconnections between the input layer, hidden layer, and output layer respectively.

$$W = \begin{bmatrix} w_{11}, & w_{12} \ldots\ldots\ldots\ldots, w_{1n} \\ w_{21}, & w_{22} \ldots\ldots\ldots\ldots, w_{2n} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ w_{m1}, w_{m2} \ldots\ldots\ldots\ldots & , w_{mn} \end{bmatrix}$$

$$V = \begin{bmatrix} v_{11}, & v_{12} \ldots\ldots\ldots\ldots, v_{1r} \\ v_{21}, & v_{22} \ldots\ldots\ldots\ldots, v_{2r} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ v_{n1}, & v_{n2} \ldots\ldots\ldots\ldots & , v_{nr} \end{bmatrix}$$

3.2 Recurrent Neural Networks

A feed forward neural network is useful if the data input is not a sequential data means not the interconnected data.

For example: Image, Attributes of a chemical etc.

This has drawbacks

- A series of dependent data can't be fed into these networks since it only works on current input.
- No memory elements: since it only works on current input there is a memory element to store the current state's information to be used by upcoming inputs.

Due to limitation of feed forward neural network which don't have additional memory element to store current state we need RNNs.

A **recurrent neural network** (**RNN**) is a class of artificial neural network where connections between nodes form a directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs [6].

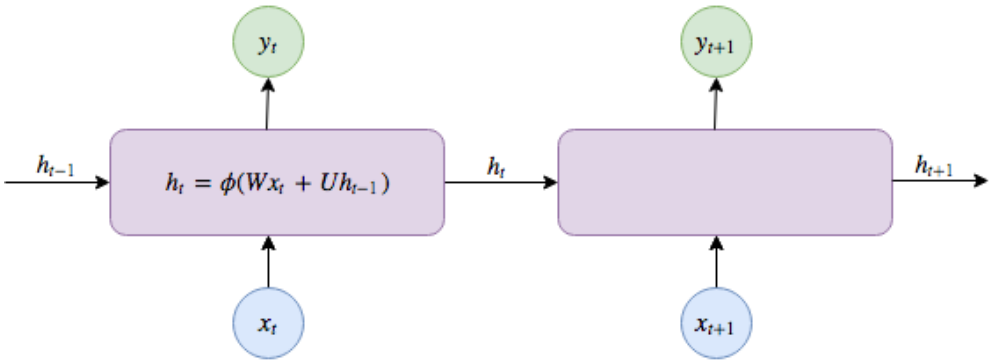


Fig 3.1: Basic overview of a Recurrent Neural Network

Recurrent neural networks add additional memory element and current output is based on previous hidden state, which is retrieved be interconnected units.

$$h_t = \phi(Wx_t + Uh_{t-1}) \quad \text{-------(8)}$$

$$y_t = Vh_t \quad \text{----------(9)}$$

Where,

h_t : Hidden state at time t

x_t : Input vector at time t

y_t : Output vector at time t

W : Weight associated with input x_t at time t

Uh_{t-1} : Weight associated with hidden state h_{t-1} at time t-1

V : Weight associated with output unit

The RNN thus introduces two major advantages:

1. *Regardless of the sequence length, the learned model always has the same input size, because it is specified in terms of transition from one state to another state, rather than specified in terms of variable-length history of states*.
2. *It is possible to use same transition function f with same parameters at every time step.*

These two factors make it possible to learn a single model f that operates on all time steps and all sequence lengths, rather than needing to learn a separate model for all possible time steps. Learning a single shared model allows generalization to sequence lengths that did not appear in the training set, and enables the model to be estimated with far fewer training examples than would be required without parameter sharing.

3.3 Recurrent Neural Networks with LSTM (Long Short Term Memory)

Long Short Term Memory networks – usually just called "LSTMs" – are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter & Schmidhuber (1997), and were refined and popularized by many people in following work. They work tremendously well on a large variety of problems, and are now widely used [6].

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.

Each LSTM cell perform operation using 4 gates namely:-

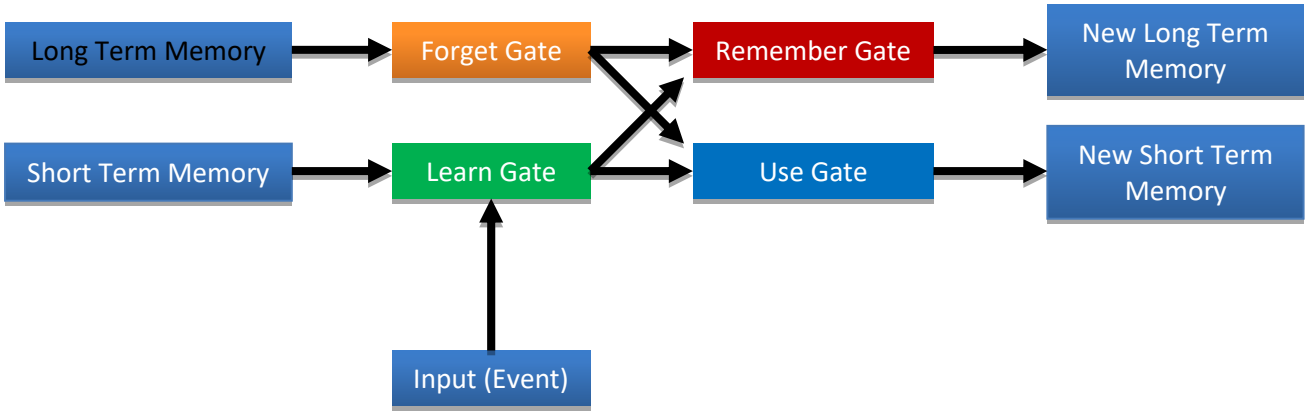


Fig 3.2: Logical view of LSTM view

1.Forget Gate : Uses previous long term memory to remove additional information.

2. Learn Gate: Uses current input and previous short term memory to create new information to be used by remember gate and use gate.

3. Remember Gate: It uses output of forget gate and learn gate to produce new long term memory.

4. Uses Gate: Uses output of learn gate and forget gate to create new short term memory which is also the current output

Input calculation of LSTM:

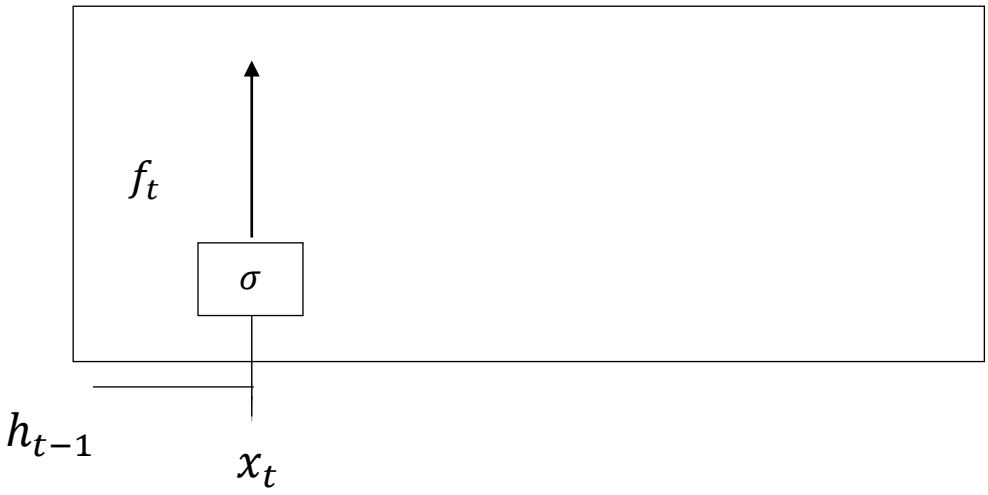


Fig 3.3: Output calculation of forget factor

$$f_t = \sigma(W_f[x_t] + U_f[h_{t-1}] + b_f) \quad \text{---------(10)}$$

This decision is made by a sigmoid layer called the "forget gate layer." It looks at h_{t-1} and x_t, and outputs a number between 0 and 1 for each number in the cell state C_{t-1}. A 1 represents "completely keep this" while a 0 represents "completely get rid of this."

To calculatef_t:

1. Multiply current input event $x_t$with weight matrix W_f
2. Multiply previous short term memory h_{t-1} with weight matrix U_f associated with short term memory.
3. Add both results with bias b_f and pass it into the sigmoid activation function to keep result within 0 and 1.

Dimension of x_t can be (n × 1), in that case W_f will be of size (1 x n), so that when we multiply it will produce a single value since (1 × n) × (n × 1) =(1 × 1).

In case of U_f it will be a single value, so that after multiplication of it with h_{t-1} we can add it with result of $W_f * U_t$and add bias b_f into it.

1.

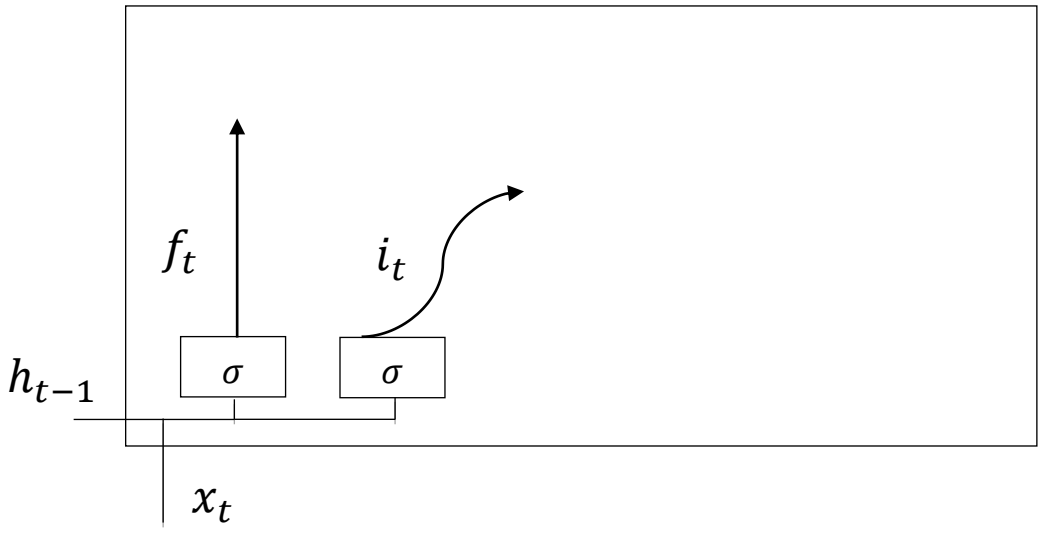


Fig 3.4: Output calculation for i_t

$$i_t = \sigma(W_i[x_t] + U_i[h_{t-1}] + b_i) \quad \text{---------(11)}$$

To calculate i_t:

1. Multiply current input event $x_t$with weight matrix W_i
2. Multiply previous short term memory h_{t-1} with weight matrix U_i associated with short term memory.
3. Add both results with bias b_i and pass it into the sigmoid activation function to keep result within 0 and 1.

Dimension of x_t can be (n × 1), in that case W_i will be of size (1 x n), so that when we multiply it will produce a single value since (1 × n) × (n × 1) =(1 × 1).

In case of U_i it will be a single value, so that after multiplication of it with h_{t-1} we can add it with result of $W_i * U_i$ and add bias b_i into it.

2.

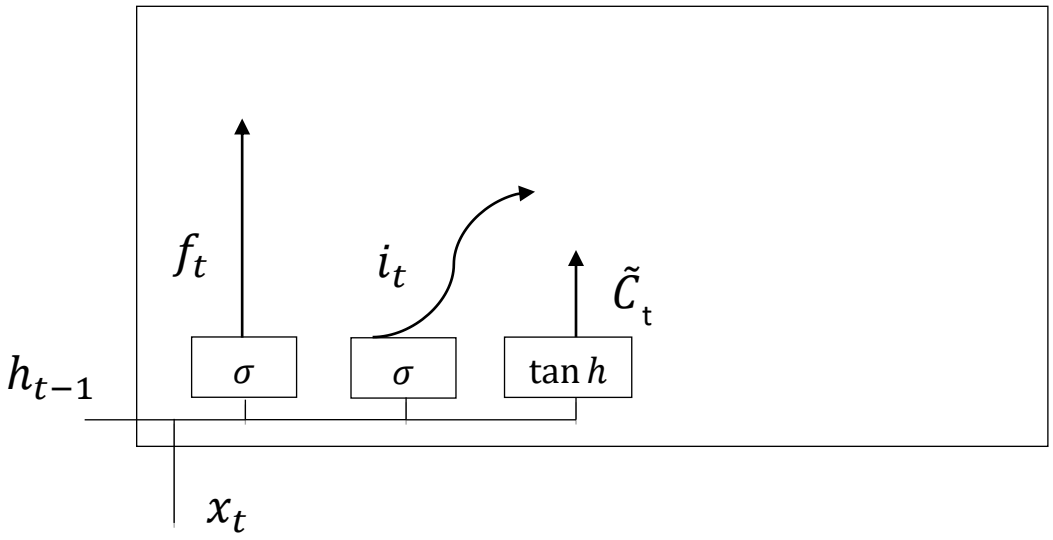


Fig 3.5: Output calculation for $\tilde{C}_t$

$$\tilde{C}_t = \tanh(W_c[x_t] + U_c[h_{t-1}] + b_c) \quad \text{----------(12)}$$

To calculate$\tilde{C}_t$:

1. Multiply current input event $x_t$with weight matrix W_c
2. Multiply previous short term memory h_{t-1} with weight matrix U_c associated with short term memory.
3. Add both results with bias b_c and pass it into the sigmoid activation function to keep result within 0 and 1.
4. Whole calculated values is passed into a $tanh$ activation function to keep the value within -1 to 1.

Dimension of x_t can be (n × 1), in that case W_c will be of size (1 x n), so that when we multiply it will produce a single value since (1 × n) × (n × 1) =(1 × 1).

In case of U_c it will be a single value, so that after multiplication of it with h_{t-1} we can add it with result of $W_c * U_c$ and add bias b_i into it.

3.

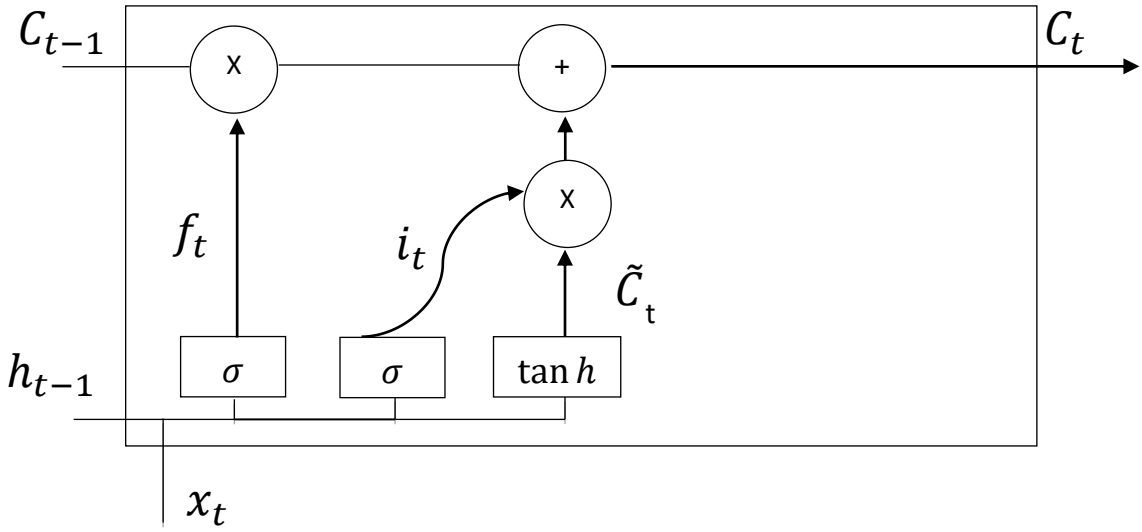


Fig 3.6: Output Caclulation for C_t

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad \text{----------(13)}$$

Using the forget factor f_t , previous long term memory C_{t-1}, i_t, and $\tilde{C}_t$ the new long term memory C_t will be calculated.

To calculate C_t:

1. Multiply f_t with C_{t-1}.
2. Multiply i_t with $\tilde{C}_t$.
3. Add both multiplication results to obtain C_t

This calculated new long term memory will serve as input for next time stamp state “t”. Since there is no input for first cell, the initiated value for C_0 is always 0.

5.

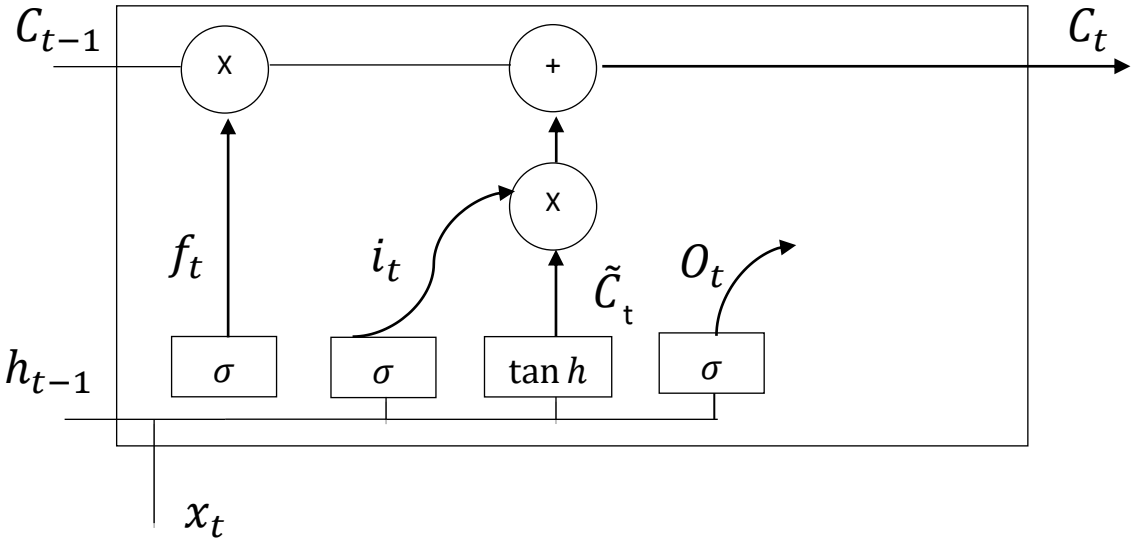


Fig 3.7: Output calculation for O_t

$$O_t = \sigma(W_o[x_t] + U_o[h_{t-1}] + b_o) \quad \text{----------(14)}$$

To calculate O_t:

1. Multiply current input event $x_t$with weight matrix W_o
2. Multiply previous short term memory h_{t-1} with weight matrix U_o associated with short term memory.
3. Add both results with bias b_o and pass it into the sigmoid activation function to keep result within 0 and 1.
4. Whole calculated values is passed into a $sigmoid$ activation function to keep the value within 0 to 1.

Dimension of x_t can be (n × 1), in that case W_o will be of size (1 x n), so that when we multiply it will produce a single value since (1 × n) × (n × 1) =(1 × 1).

In case of U_o it will be a single value, so that after multiplication of it with h_{t-1} we can add it with result of $W_o * U_o$ and add bias b_o into it

6.

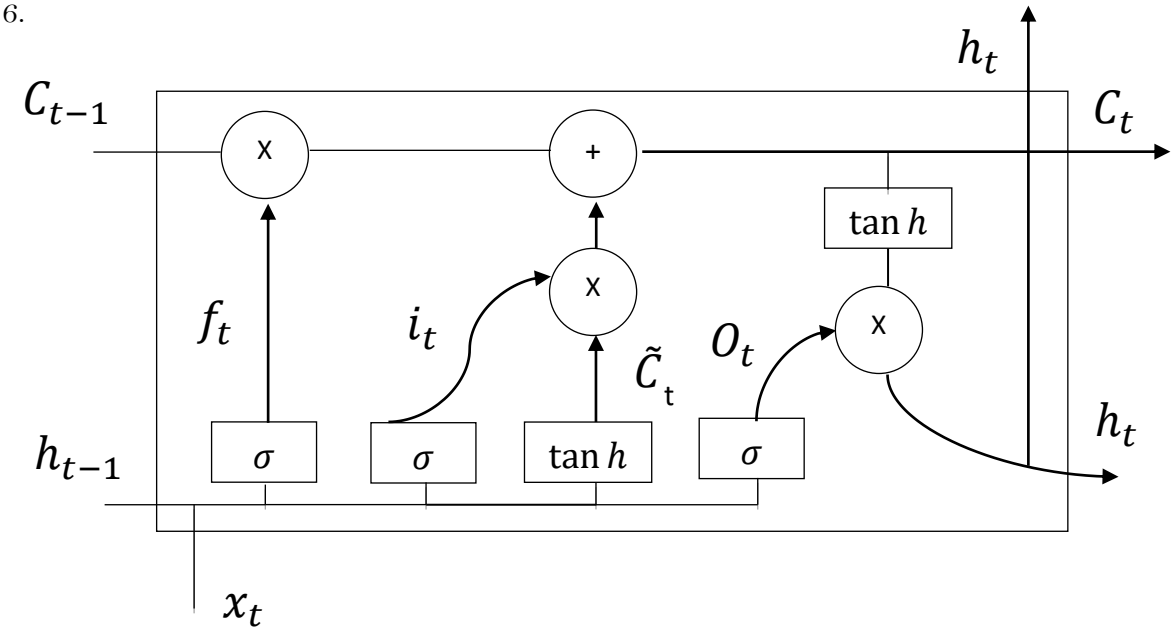


Fig 3.8: Output calculation for h_t

$$h_t = O_t * \tanh(C_t) \qquad \text{----------(15)}$$

To calculate the next short term memory (h_t) which is also the current output the values of O_t and the calculated next long term memory (C_t).

To calculate h_t:

1. Pass the C_t into a $tanh$ activation function to keep the values within -1 to 1.
2. Multiply the value with O_t to get the new short term memory (h_t).

3.4 Adam Optimization Algorithm

Stochastic gradient-based optimization is of core practical importance in many fields of science and engineering. Many problems in these fields can be cast as the optimization of some scalar parameterized objective function requiring maximization or minimization with respect to its parameters. If the function is differentiable w.r.t. its parameters, gradient descent is a relatively efficient optimization method, since the computation of first-order partial derivatives w.r.t. all the parameters is of the same computational complexity as just evaluating the function. Often, objective functions are stochastic. For example, many objective functions are composed of a sum of sub functions evaluated at different subsamples of data; in this case optimization can be made more efficient by taking gradient stepsw.r.t. Individualsub functions, i.e. stochastic gradient descent (SGD) or ascent. SGD proved itself as an efficient and effective optimization method [1].

Adam is a method for efficient stochastic optimization that only requires first-order gradients with little memory requirement. The method computes individual adaptive learning

rates for different parameters from estimates of first and second moments of the gradients; the nameAdamis derived from adaptive moment estimation. Adam can be looked at as a combination of RMSprop and Stochastic Gradient Descent with momentum. It uses the squared gradients to scale the learning rate like RMSprop and it takes advantage of momentum by using moving average of the gradient instead of gradient itself like SGD with momentum. [1]

Algorithm: g_t^2 indicates the square $g_t \odot g_t$.

Default values for

$\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\in = 10^{-8}$

All operations on vectors are element-wise. Withβ_1^tandβ_2^twe denote$\beta_1$and$\beta_2$to the power t.

Require:$\propto$: Stepsize

Require:$\beta_1, \beta_2 \in [0,1)$: Exponential decay rates for the moment estimates

Require:$f(\Theta)$: Stochastic objective function with parameters θ

Require:θ_0: Initial parameter vector

$m_0 \leftarrow 0$(Initialize 1st moment vector)
$v_0 \leftarrow 0$(Initialize 2nd moment vector)
$t \leftarrow 0$(Initialize timestep)
while$\theta_t$not converged **do**

$$t \leftarrow t + 1$$

$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$(Get gradients w.r.t. stochastic objective at timestep t)
$m_t \leftarrow \beta_1 . m_{t-1} + (1 - \beta_1) . g_t$(Update biased first moment estimate)
$v_t \leftarrow \beta_2 . v_{t-1} + (1 - \beta_2) . g_t^2$(Update biased second raw moment estimate)
$\widehat{m_t} \leftarrow m_t / (1 - \beta_1^t)$(Compute bias-corrected first moment estimate)
$\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$(Compute bias-corrected second raw moment estimate)
$\theta_t \leftarrow \theta_{t-1} - \propto . \hat{m}_t / (\sqrt{\hat{v}_t} + \in)$(Update parameters)
end while

return θ_t(Resulting parameters)

Properties of Adam

1. Actual step size taken by the Adam in each iteration is approximately bounded the step size hyper-parameter. This property add intuitive understanding to previous unintuitive learning rate hyper-parameter.

2. Step size of Adam update rule is invariant to the magnitude of the gradient, which helps a lot when going through areas with tiny gradients (such as saddle points or ravines). In these areas SGD struggles to quickly navigate through them.

3. Adam was designed to combine the advantages of Adagrad, which works well with sparse gradients, and RMSprop, which works well in on-line settings. Having both of these enables us to use Adam for broader range of tasks. Adam can also be looked at as the combination of RMSprop and SGD with momentum.

Comparison of Adam's Performance in this case with SGD:

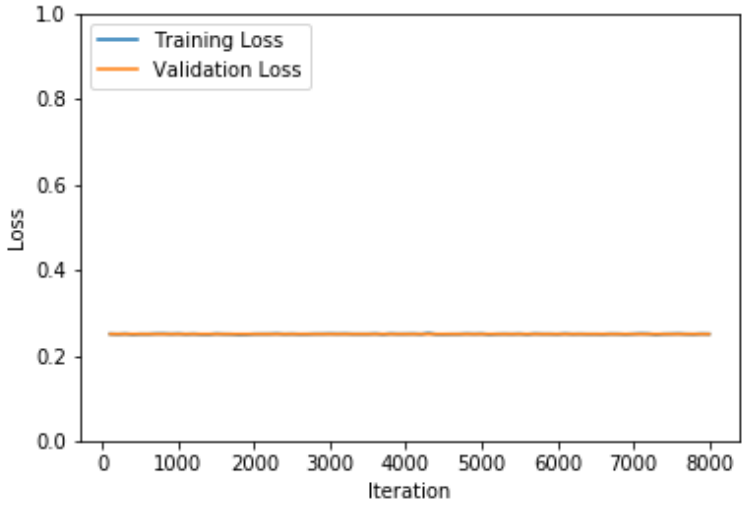


Fig 3.9: Training and Validation Loss of Stochastic gradient descent (SGD)

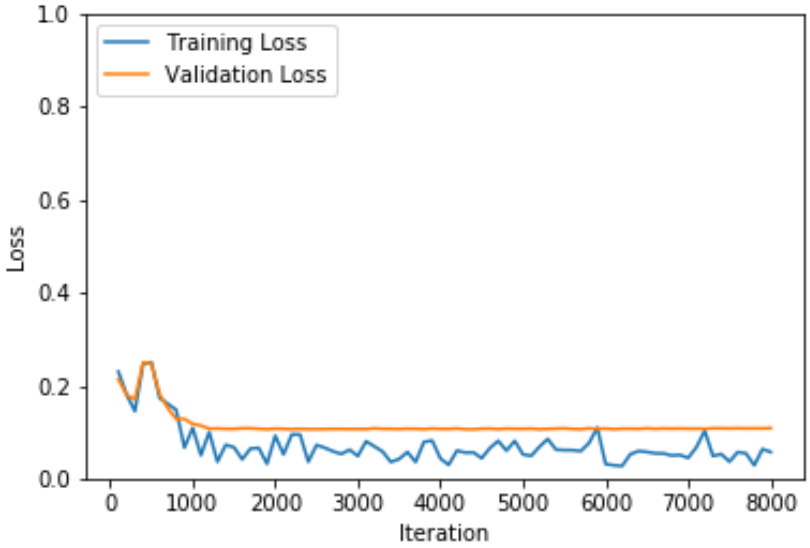


Fig 3.10: Training and Validation Loss of Adam

Hence it is concluded that Adam is better choice compared to SDG in this sentiment analysis problem.

4. Implementation: Sentiment Analysis Model

The deep learning model to be designed will use Supervised learning to train the model. This form of learning assumes the availability of a labeled (i.e., ground-truthed) set of training data made up of N input—output examples:

$$\boldsymbol{T = \{(x_i, d_i)\}_{i=1}^{N}} \quad \textbf{------(16)}$$

where $\mathbf{x_i}$ = input vector of i^{th} example

$\mathbf{d_i}$ = desired (target) response of i^{th} example, assumed to be scalar for convenience of presentation

N = sample size

Given the training sample T,the requirement is to compute the free parameters of the neural network so that the actual output y_i of the neural network due to x_i is close enough to d_i for all i in a statistical sense. For example, we may use the mean-square error

$$\boldsymbol{E(n) = \frac{1}{N}\sum_{i=1}^{N}(d_i - y_i)^2} \quad \textbf{------(17)}$$

as the index of performance to be minimized.

The training data contains the reviews in text format.

4.1 Data Encoding

Training data contains reviews in text format with their corresponding class as Positive or Negative.

For example:

Data: “this film is mediocre at best . angie harmon is as funny as a bag of hammers. her bitchy demeanor from law and order carries over in a failed attempt at comedy. charlie sheen is the only one to come out unscathed in this horrible anti comedy . the only positive thing to come out of this mess is charlie and denise s marriage . hopefully that effort produces better results .”

Label : “negative”

Here are the processing steps to clean the data:

- Get rid of periods and extraneous punctuation.
- Reviews are delimited with newline characters \n. To deal with those split the text into each review using \n as the delimiter.
- Combined all the reviews back together into one big string.
- If review length is zero or near zero then remove the review.

The embedding lookup requires that we pass in integers to network. The easiest way to do this is to create dictionaries that map the words in the vocabulary to integers. Then we can convert each of reviews into integers so they can be passed into the network.To feed the data into model first we need to encode the individual words into tokens of integers.

For example a text **"This was the worst movie i have ever seen"**can be encoded into token of integers as **"[10, 25, 1, 47, 55, 9, 5, 19, 77]"**.

Encoding the text into tokens will include following steps:

1. Create a mapping of each unique word into an integer as a dictionary.
2. For each word in the review replace the text with corresponding word to integer mapping.

For labels Positive can be denoted as "1" and negative as "0".

Padding Sequence: The length of sentence might vary, hence a fixed padding sequence length will be used.

If encoded review is less than padding sequence length then pad 0s at front of data. For example if padding sequence length = **15**, and encoded data is = **"[10, 25, 1, 47, 55, 9, 5, 19, 77]"**.

Then final encoded data after padding will be = **"[0, 0, 0, 0, 0, 0, 10, 25, 1, 47, 55, 9, 5, 19, 77]"**

If encoded review is greater than padding sequence length then we consider first tokens up to the padding sequence length. For example if padding sequence length = **5** , and the encoded data is = **"[10, 25, 1, 47, 55, 9, 5, 19, 77]"**.

Then final encoded data after padding will be = **"[10, 25, 1, 47, 55]"**

The following is Python code to clean, encode and create the train, test and validation set of data.

```
import numpy as np
import os
from string import punctuation
from collections import Counter
import torch
from torch.utils.data import TensorDataset,DataLoader
import torch.nn as nn

defcreateData(positiveDataPath,negativeDataPath):
    reviews=[]
    labels=[]
    all_files_neg = os.listdir(negativeDataPath)
    all_files_pos = os.listdir(positiveDataPath)
for file_location in all_files_pos:
```

```
# For all positive files, map the word ‘positive’ and for all negative map
‘negative’
withopen(positiveDataPath+file_location,'r',encoding="utf8")as f:
            reviews.append(f.read())
            labels.append("positive")

for file_location in all_files_neg:
withopen(negativeDataPath+file_location,'r',encoding="utf8")as f:
            reviews.append(f.read())
            labels.append("negative")
    myDict={}
# create a dict for each review with its index mapping to shuffle them
for i inrange(len(reviews)):
       myDict[i]=i
    reviewArray=np.arange(len(reviews))
    np.random.shuffle(reviewArray)
    reviewString=""
    labelString=""
for i inlist(reviewArray):
        reviewString=reviewString+reviews[i]+"\n"
        labelString=labelString+labels[myDict[i]]+"\n"

# final data and its corresponding labels are returned
return reviewString,labelString

‘’’Function to pad additional zeros or trim down the review according to
sequence length’’’
deffeaturesPadding(reviews,sequenceLength):
  features=np.zeros((len(reviews),sequenceLength),dtype=int)
for i,row inenumerate(reviews):
    features[i,-len(row):]=np.array(row)[:sequenceLength]
return features


defgetFeaturesLabels(reviews,labels,seq_length):
  # For all reviews convert them to lower case and remove punctuations
  reviews=reviews.lower()
  all_text=''.join([c for c in reviews if c notin punctuation])
  reviews_split=all_text.split('\n')
  all_text=' '.join(reviews_split)
  words=all_text.split()
  counts=Counter(words)

  # For all reviews read all unique words and map each to a integer by
creating a collection of unique words

  vocab=sorted(counts,key=counts.get,reverse=True)
  vocab_to_int={word:ii for ii, word inenumerate(vocab,1)}
  reviews_ints=[]
```

```
for review in reviews_split:
    reviews_ints.append([vocab_to_int[word] for word in review.split()])
  labels_split=labels.split('\n')
  encoded_labels=np.array([1if label=='positive'else0for label in
labels_split])
  reviews_lens=Counter([len(x) for x in reviews_ints])
  non_zero_idx=[ii for ii,review inenumerate(reviews_ints) iflen(review)!=0]
  reviews_ints=[reviews_ints[ii] for ii in non_zero_idx]
  encoded_labels=np.array([encoded_labels[ii] for ii in non_zero_idx])
  features=featuresPadding(reviews_ints,seq_length)
return features,encoded_labels,vocab_to_int


'''Function to split the data into train, test and validation sets'''

defcreateTrainTestValidateData(split_frac,features,encoded_labels):
  split_idx = int(len(features)*split_frac)
  train_x, remaining_x = features[:split_idx], features[split_idx:]
  train_y, remaining_y = encoded_labels[:split_idx],
encoded_labels[split_idx:]
  test_idx = int(len(remaining_x)*0.5)
  val_x, test_x = remaining_x[:test_idx], remaining_x[test_idx:]
  val_y, test_y = remaining_y[:test_idx], remaining_y[test_idx:]
return train_x,train_y,test_x,test_y,val_x,val_y
```

4.2 Model Overview

Once data has been encoded and padded. The model needs to be created. One way of feeding the data is to one hot encode the inputs as vectors, but this method is only feasible there are less number of inputs.

We need to add an embedding layer because there are more than 100000words in our vocabulary. It is massively inefficient to one-hot encode that many classes. So, instead of one-hot encoding, we can have an embedding layer and use that layer as a lookup table. We could train an embedding layer using Word2Vec, then load it here. But, it's fine to just make a new layer, using it for only dimensionality reduction, and let the network learn the weights.

So if the embedding layer has 100000 × 400 dimension then it will reduce the input data dimension to 400.

Final Model's Architecture is as follows:

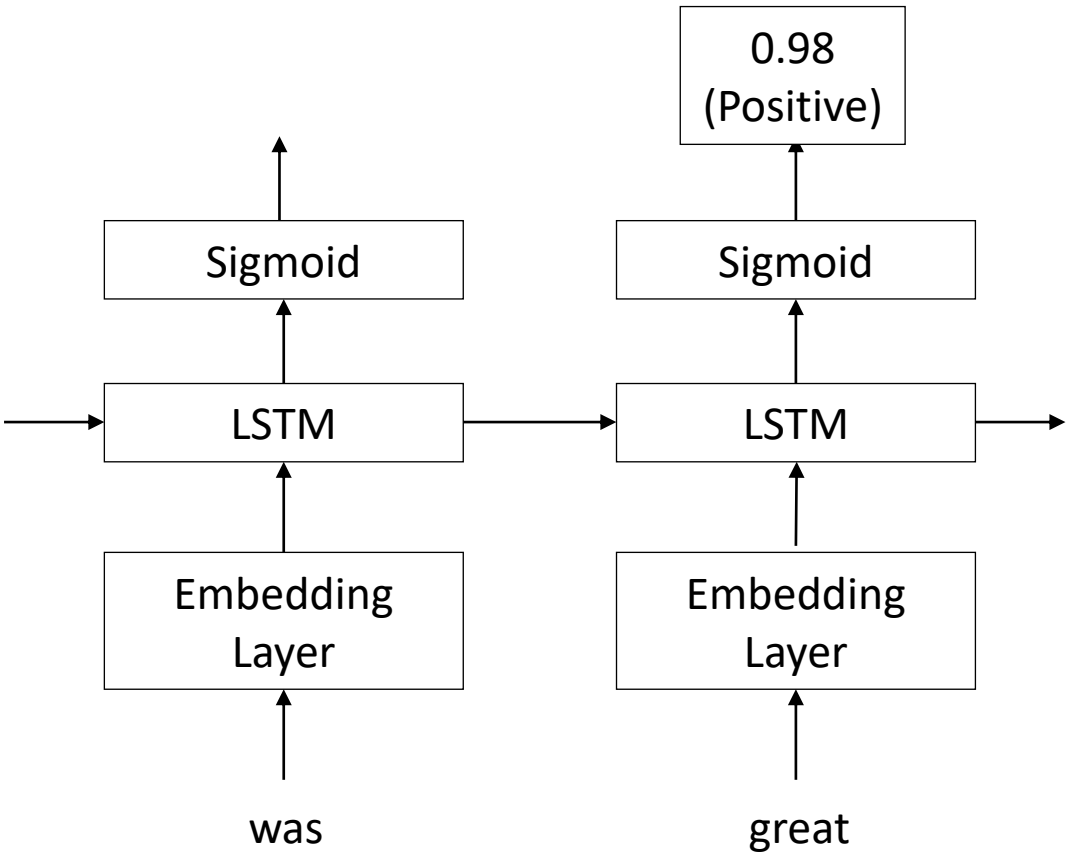


Fig 4.1: A logical view of model's architecture.

The description of model is as follows:

1. An embedding layer that converts our word tokens (integers) into embedding of a specific size
2. An LSTM layer defined by a hidden state size and number of layers
3. A fully-connected output layer that maps the LSTM layer outputs to a desired output size
4. A sigmoid activation layer, which turns all, outputs into a value 0-1; return **only the last sigmoid output** as the output of this network.

Here the hyper parameters are:

Embedding dimension	:	(Vocabulary size+1) × 600
Hidden Dimension	:	512
Number of Stacked LSTMs	:	3
Learning Rate (α)	:	0.003

Output size : $1\begin{cases} Positive\ \ if\ output > 0.5 \\ Negative\ if\ output < 0.5 \end{cases}$

4.3 Learning Algorithm

Adam algorithm that is used to work with gradients of batches, is initiated with default parameters of the algorithm [1].

Default parameters are:

$\beta_1 = 0.9$, $\beta_2 = 0.999$, $\in = 10^{-8}$

Following is implementation of Adam algorithm to work with gradients using Python:

```
from torch.optim import Optimizer
import  math
classAdamOptimizerAlgorithm(Optimizer):

def__init__(self, params, lr=1e-3, betas=(0.9, 0.999), eps=1e-8):
        defaults = dict(lr=lr, betas=betas, eps=eps)
super(AdamOptimizerAlgorithm, self).__init__(params, defaults)

defstep(self):
for group inself.param_groups:
for p in group['params']:
if p.grad isNone:
continue
                grad = p.grad.data
                state = self.state[p]

# State initialization in case of initial state
iflen(state) == 0:
                    state['step'] = 0
# Exponential moving average of gradient values
                    state['exp_avg'] = torch.zeros_like(p.data)
# Exponential moving average of squared gradient values
                    state['exp_avg_sq'] = torch.zeros_like(p.data)


                exp_avg, exp_avg_sq = state['exp_avg'], state['exp_avg_sq']

                beta1, beta2 = group['betas']

                state['step'] += 1

#following is just a formula implementaion of the algorithm
                exp_avg.mul_(beta1).add_(1 - beta1, grad)
```

```
                exp_avg_sq.mul_(beta2).addcmul_(1 - beta2, grad, grad)

                denom = exp_avg_sq.sqrt().add_(group['eps'])

                b_1 = 1 - beta1 ** state['step']
                b_2 = 1 - beta2 ** state['step']
                step_size = group['lr'] * math.sqrt(b_2) / b_1

                p.data.addcdiv_(-step_size, exp_avg, denom)  # -
step_size*(exp_avg/denom) operation done in single step
```

4.4 Training

For training we need to split the data into 3 parts: Training data, Validation data, and Testing data.

Usually the 80% of data is used for training and rest 20% is split in half for validation and testing.

Following Python code snippet is used to split data into corresponding sets:

```
'''Function to create the train, test and validation sets, by splitting the original data by given percentage into the train sets, and remaining data into half for validation and test sets '''

defcreateTrainTestValidateData(split_frac,features,encoded_labels):
  split_idx = int(len(features)*split_frac)
  train_x, remaining_x = features[:split_idx], features[split_idx:]
  train_y, remaining_y = encoded_labels[:split_idx],
encoded_labels[split_idx:]
  test_idx = int(len(remaining_x)*0.5)
  val_x, test_x = remaining_x[:test_idx], remaining_x[test_idx:]
  val_y, test_y = remaining_y[:test_idx], remaining_y[test_idx:]
return train_x,train_y,test_x,test_y,val_x,val_y
```

Creating Batches:

Since Adam algorithm works on batches of data, It is efficient to get the batches within size such that approx. average gradient is effective when calculating the updated weights.

For example:

Batch size = 100

Total training data = 20000

No of iterations per training epoch = $\frac{Total\ training\ dat}{Batch\ size} = \frac{20000}{100}$ = 200 iterations

Training Process:

1. Make forward pass with training data.
2. Calculate the loss using following formula of **Mean Square Error**

$$\boldsymbol{MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2} \quad \textbf{------(18)}$$

3. Backward pass to update the weights using Adam Algorithm
4. Repeat till end of epochs

Avoiding Overfitting:

Overfitting is a modeling error which occurs when a function is too closely fit to a limited set of data points. Overfitting the model generally takes the form of making an overly complex model to explain idiosyncrasies in the data under study.

In reality, the data often studied has some degree of error or random noise within it. Thus attempting to make the model conform too closely to slightly inaccurate data can infect the model with substantial errors and reduce its predictive power.[5]

Methods used to avoid overfitting are as follows:

1. **Regularization using dropout:** Deep neural nets with a large number of parameters are very powerful machine learning systems. However, over fitting is a serious problem in such networks. Large networks are also slow to use, making it difficult to deal with over fitting by combining the predictions of many different large neural nets at test time. Dropout is a technique for addressing this problem. The key idea is to randomly drop units (along with their connections) from the neural network during training. This prevents units from co-adapting too much. During training, dropout samples from an exponential number of different "thinned" networks. At test time, it is easy to approximate the effect of averaging the predictions of all these thinned networks by simply using a single unthinned network that has smaller weights. This significantly reduces over fitting and gives major improvements over other regularization methods. We show that dropout improves the performance of neural networks on supervised learning tasks in vision, speech recognition, document classification and computational biology, obtaining state-of-the-art results on many benchmark data sets.

2. **Early Stopping:** As model proceeds in training, it learns better weights over time. But after some epochs, when error is low, the model starts to memorize the training data, thus performs poorly in classification after the training. To overcome that the **Validation data** is used to perform early stopping in training.

After each fixed amount of intervals, the validation data is feed into the network and it is only used to check the accuracy, if network starts to perform poorly on validation data, then training is stopped. Following figure shows a graphical view of over fitting, when the

validation loss starts increasing, this indicates that model is making more errors in validation data.

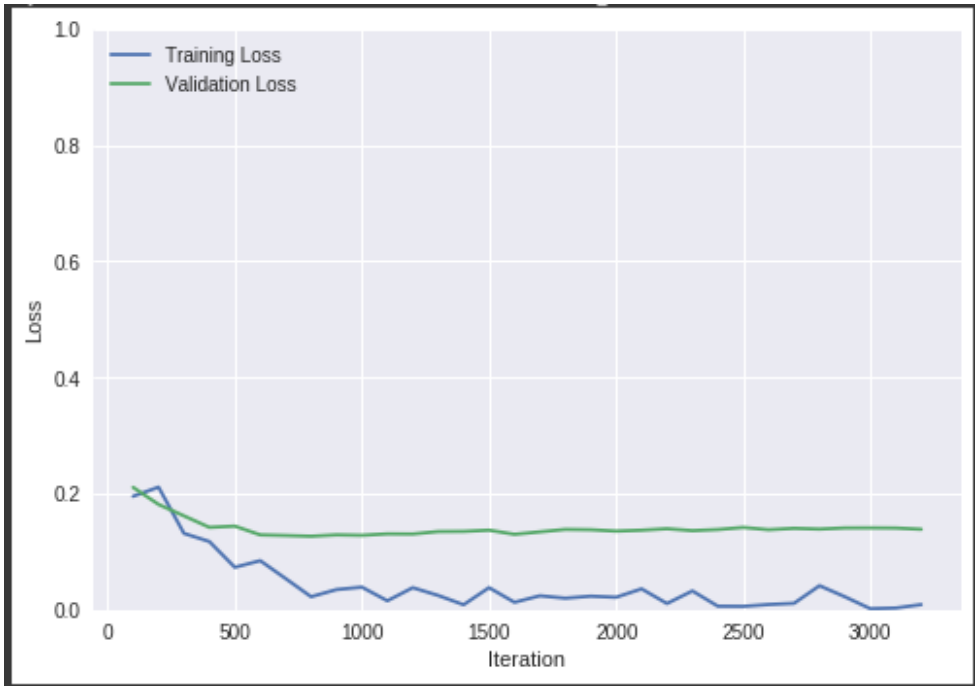


Fig 4.2: Comparison of Validation and Training loss.

3. **Reducing Learning Rate Over Time:** As we move closer to global minimum, error becomes less, thus keeping the learning rate high might result in overshooting from global minimum. Thus after few epochs it is necessary to lower the learning rate and train for more epochs.

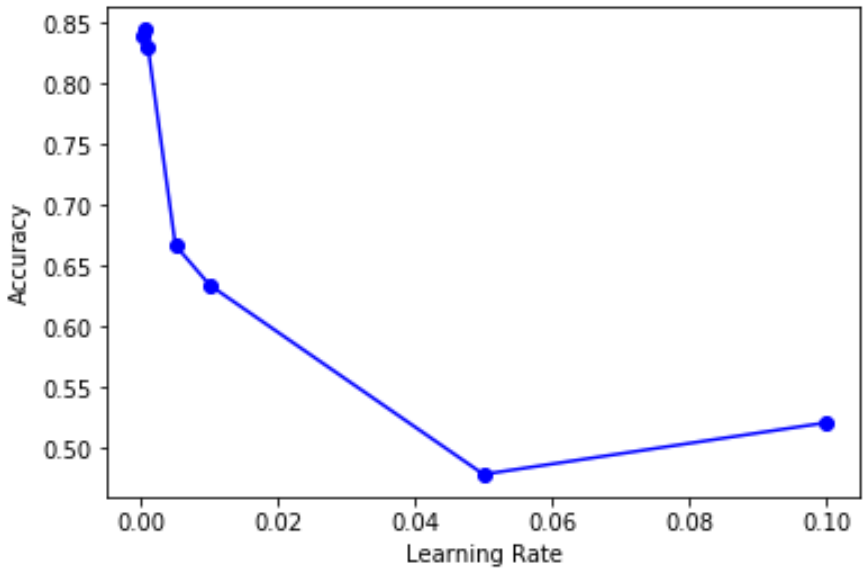


Fig 4.3: Comparison of different learning rates in training.

The training log:

```
Epoch: 1       Iteration: 100  Training Loss: 0.2418555      Validation
Loss: 0.2359691
Validation loss decreased  hence saving checkpoint successfully
Epoch: 1       Iteration: 200  Training Loss: 0.1766867      Validation
Loss: 0.1915403
Validation loss decreased  hence saving checkpoint successfully
Epoch: 2       Iteration: 300  Training Loss: 0.1953648      Validation
Loss: 0.1558252
Validation loss decreased  hence saving checkpoint successfully
Epoch: 2       Iteration: 400  Training Loss: 0.1183145      Validation
Loss: 0.1264227
Validation loss decreased  hence saving checkpoint successfully
Epoch: 3       Iteration: 500  Training Loss: 0.0701459      Validation
Loss: 0.1303956
Epoch: 3       Iteration: 600  Training Loss: 0.0913621      Validation
Loss: 0.1347342
Reducing the learning rate!
Epoch: 4       Iteration: 700  Training Loss: 0.0187498      Validation
Loss: 0.1279074
Epoch: 4       Iteration: 800  Training Loss: 0.0281214      Validation
Loss: 0.1257939
Validation loss decreased  hence saving checkpoint successfully
Epoch: 5       Iteration: 900  Training Loss: 0.0450199      Validation
Loss: 0.1280346
Epoch: 5       Iteration: 1000 Training Loss: 0.0244467      Validation
Loss: 0.1282429
Epoch: 6       Iteration: 1100 Training Loss: 0.0295494      Validation
Loss: 0.1305099
Epoch: 6       Iteration: 1200 Training Loss: 0.0104207      Validation
Loss: 0.1324596
Epoch: 7       Iteration: 1300 Training Loss: 0.0230359      Validation
Loss: 0.1337625
Epoch: 7       Iteration: 1400 Training Loss: 0.0175603      Validation
Loss: 0.1374276
Epoch: 8       Iteration: 1500 Training Loss: 0.0315925      Validation
Loss: 0.1379397
Epoch: 8       Iteration: 1600 Training Loss: 0.0194792      Validation
Loss: 0.1375286
Epoch: 9       Iteration: 1700 Training Loss: 0.0100298      Validation
Loss: 0.1388796
Epoch: 9       Iteration: 1800 Training Loss: 0.0298400      Validation
Loss: 0.1387108
Epoch: 10      Iteration: 1900 Training Loss: 0.0102925      Validation
Loss: 0.1383559
Epoch: 10      Iteration: 2000 Training Loss: 0.0104309      Validation
Loss: 0.1380666
Epoch: 11      Iteration: 2100 Training Loss: 0.0103464      Validation
Loss: 0.1417695
Epoch: 11      Iteration: 2200 Training Loss: 0.0000727      Validation
Loss: 0.1373071
Epoch: 12      Iteration: 2300 Training Loss: 0.0000936      Validation
Loss: 0.1415522
Epoch: 12      Iteration: 2400 Training Loss: 0.0001245      Validation
Loss: 0.1405720
Epoch: 13      Iteration: 2500 Training Loss: 0.0025269      Validation
Loss: 0.1410832
Epoch: 13      Iteration: 2600 Training Loss: 0.0003073      Validation
Loss: 0.1444580
```

```
Epoch: 14      Iteration: 2700 Training Loss: 0.0144143      Validation
Loss: 0.1422861
Epoch: 14      Iteration: 2800 Training Loss: 0.0057060      Validation
Loss: 0.1455268
Epoch: 15      Iteration: 2900 Training Loss: 0.0296322      Validation
Loss: 0.1441364
Epoch: 15      Iteration: 3000 Training Loss: 0.0001388      Validation
Loss: 0.1419338
Epoch: 16      Iteration: 3100 Training Loss: 0.0244618      Validation
Loss: 0.1424383
Epoch: 16      Iteration: 3200 Training Loss: 0.0118945      Validation
Loss: 0.1441191
Training Time: 41.66050688823064 minutes
```

Used Environment for training:

```
CPU: Intel(R) Xeon(R) CPU @ 2.20GHz
GPU: Nvidia Tesla K80 12 GB
RAM: 12 GB
Storage: 358 GB
```

4.5 Making Predictions

Making predictions on the trained model is obtained using following steps:

1. Use the dictionary to integer mapping to convert the input sentence into tokens.
2. If word is not in dictionary then drop that word.
3. Pad the tokenized words to generate input stream.
4. Forward pass the input stream to get the sigmoid output from last layer.
5. If output value < 0.5 then sentence is **Negative** otherwise **Positive**.

4.6 Testing with custom inputs

Sample tokenized text:

```
sentence="Avengers Endgame was most awaited movie of this year, and it performed as expected. 3 Hours of pure fun"
test_ints=tokenize_sentence(sentence,vocab_to_int)
print(test_ints)
```

```
[[18250, 25950, 13, 87, 13065, 17, 4, 10, 335, 2, 8, 2537, 14, 853, 438, 612, 4, 1023, 249]]
```

Fig 4.4: Tokenization of a given review

Making prediction:

```
result=predict(model,sentence,vocab_to_int,seq_length)
print(result)
```

```
0.9697  Positive sentence!
```

Fig 4.5: Prediction result of a given review

Accuracy of model on test data:

```
checkAccuracy(model,test_loader)
```

```
Accuracy: 0.8488
```

Fig 4.6: Overall Accuracy on test data

Following is the code snippet used to calculate accuracy:

```
defcheckAccuracy(model,test_loader):
#move model to GPU
  model.cuda()

  test_losses=[]
  correct_prediction=0

#set initial state to zero
  hidden_state=model.initialize_hidden_state(batch_size)

# set the model into evaluation mode, so that we don't need to calculate
# the gradients
  model.eval()

for inputs,labels in test_loader:

# move the data and labels into GPU
    inputs,labels=inputs.cuda().long(),labels.cuda().long()

    output,hidden_state=model(inputs,hidden_state)
# labels are initially long or int, we need to conver them into float
# so that loss cab be calculated in float value
    test_loss=criterion(output.squeeze(),labels.float())
    test_losses.append(test_loss.item())

# get the prediction either 1 or 0
    prediction=torch.round(output.squeeze())

    correct_tensor=prediction.eq(labels.float().view_as(prediction))
    correct=np.squeeze(correct_tensor.cpu().numpy())
    correct_prediction+=np.sum(correct)

#print(np.mean(test_losses))
print("Accuracy: {:.4f}".format(correct_prediction/len(test_loader.dataset)))
```

5. Tools: Recommendation System

5.1 Introduction

Recommender systems are defined as recommendation inputs given by the people, which the system then aggregates and directs to appropriate recipients. It can be further defined as a system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting objects in a larger space of possible options. Recommender system will become an integral part of the Media and Entertainment (M&E) industry in the near future.

Many e-commerce and retail companies are leveraging the power of data and boosting sales by implementing recommender systems on their websites. In short, these systems aim to predict users' interests and recommend items that quite likely are interesting for them. Data required for recommender systems stems from explicit user ratings after watching a movie or listening to a song, from implicit search engine queries and purchase histories, or from other knowledge about the users/items themselves [7].

5.2 Need of Recommendation System

Companies using recommender systems focus on increasing sales as a result of very personalized offers and an enhanced customer experience.

Recommendations typically speed up searches and make it easier for users to access content they are interested in, and surprise them with offers they would have never searched for.

What is more, companies are able to gain and retain customers by sending out emails with links to new offers that meet the recipients' interests or suggestions of films and TV shows that suit their profiles.

The user starts to feel known and understood and is more likely to buy additional products or consume more content. By knowing what a user wants, the company gains competitive advantage and the threat of losing a customer to a competitor decreases.

Providing that benefit to users by including recommendations in systems and products is appealing. Furthermore, it allows companies to position ahead of their competitors and eventually increase their earnings.

5.3 Types of Recommendation System

Recommender systems function with two kinds of information:

- *Characteristic information*. This is information about items (keywords, categories, etc.) and users (preferences, profiles, etc.).
- *User-item interactions.* This is information such as ratings, number of purchases, likes, etc.

Based on these there are 3 major classification of Recommendation Systems namely:

I) Content Based Filtering Recommendation System
II) Collaborative Filtering System
III) Hybrid System

5.3.1 Content Based Filtering Recommendation System

These systems make recommendations using a user's item and profile features. They hypothesize that if a user was interested in an item in the past, they will once again be interested in it in the future. Similar items are usually grouped based on their features. User profiles are constructed using historical interactions or by explicitly asking users about their interests. There are other systems, not considered purely content-based, which utilize user personal and social data.

One issue that arises is making obvious recommendations because of excessive specialization (user A is only interested in categories B, C, and D, and the system is not able to recommend items outside those categories, even though they could be interesting to them).

Another common problem is that new users lack a defined profile unless they are explicitly asked for information. Nevertheless, it is relatively simple to add new items to the system. We just need to ensure that we assign them a group according to their features.

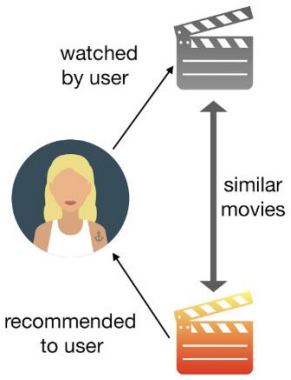
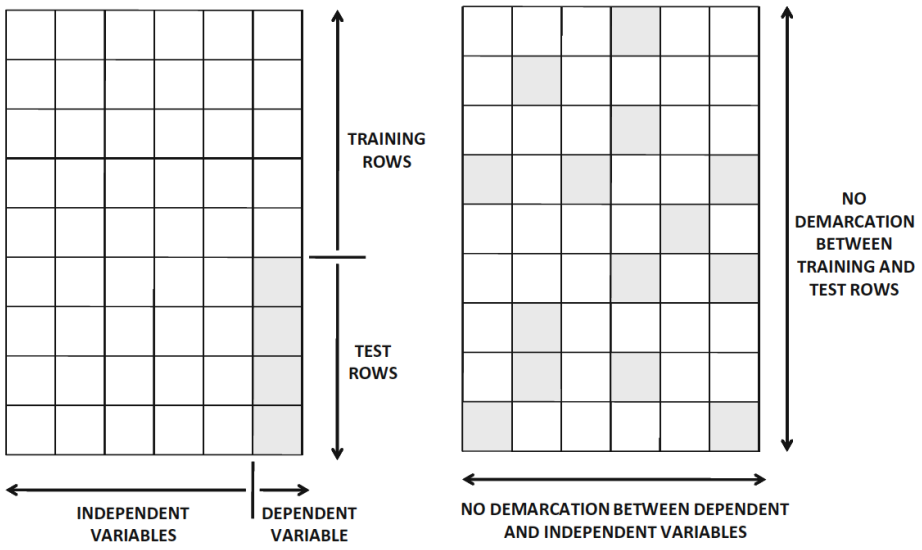


Fig 5.1: A logical view of content based filtering recommendation system

5.3.2 Collaborative Filtering Recommendation System

These kinds of systems utilize user interactions to filter for items of interest. We can visualize the set of interactions with a matrix, where each entry (i,j)(i, j)(i,j) represents the interaction between user iii and item jjj. An interesting way of looking at collaborative filtering is to think of it as a generalization of classification and regression. While in these cases we aim to predict a variable that directly depends on other variables (features), in collaborative filtering there is no such distinction of feature variables and class variables.

Visualizing the problem as a matrix, we don't look to predict the values of a unique column, but rather to predict the value of any given entry.



(a) Classification (b) Collaborative filtering

In short, collaborative filtering systems are based on the assumption that if a user likes item A and another user likes the same item A as well as another item, item B, the first user could also be interested in the second item. Hence, they aim to predict new interactions based on historical ones.

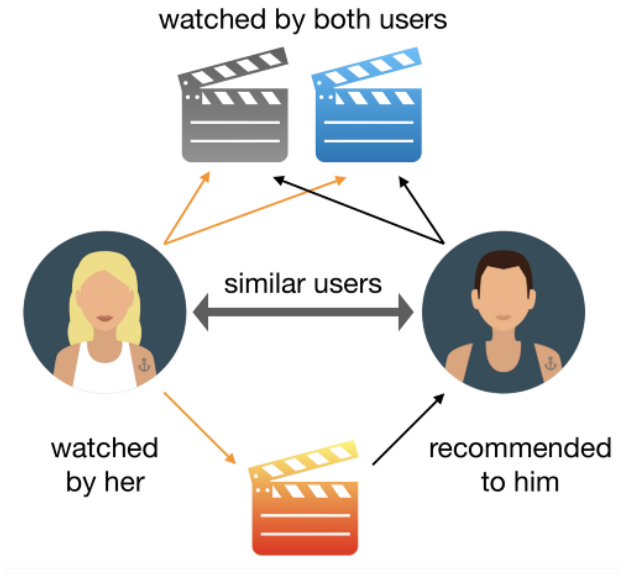


Fig 5.2: A logical view of collaborative filtering recommendation system

5.3.3 Hybrid Recommendation System

Hybrid Recommendation systems are combining collaborative and content-based recommendation can be more effective. Hybrid approaches can be implemented by making content-based and collaborative-based predictions separately and then combining them.

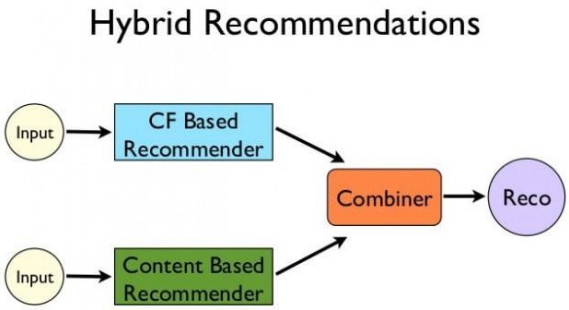


Fig 5.3: Logical view of hybrid recommendation system

6.Implementation:RecommendationSystem with Sentiment Analysis

Recommendation systems are important intelligent systems that play a vital role in providing selective information to users. Traditional approaches in recommendation systems include collaborative filtering and content-based filtering. However, these approaches have certain limitations like the necessity of prior user history and habits for performing the task of recommendation. In order to reduce the effect of such dependencies this recommendation system combines the content-based filtering with sentiment analysis of movie tweets. The movie tweets will be collected from Twitter understand the user response of the movie and create a rating database for movies.

The recommendation system implemented here used the **Content-based recommendation system with Sentiment analysis**. This due to the following problems that is faced in**Collaborative-filtering recommendation systems**:-

1. Cold start: we should have enough information (user-item interactions) for the system to work. If we setup a new e-commerce site, we cannot give recommendations until users have interacted with a significant number of items.

2. Adding new users/items to the system: whether it is a new user or item, we have no prior information about them since they don't have existing interactions.

6.1 Model Overview

The purposed model has been shown in following Fig(1). Model requires two different types of opeartion.

1. Processing the reviews to extract the rating for each movie in database.
2. Using Content-Based filtering to extract socre for each recommended movie depending on catgories.

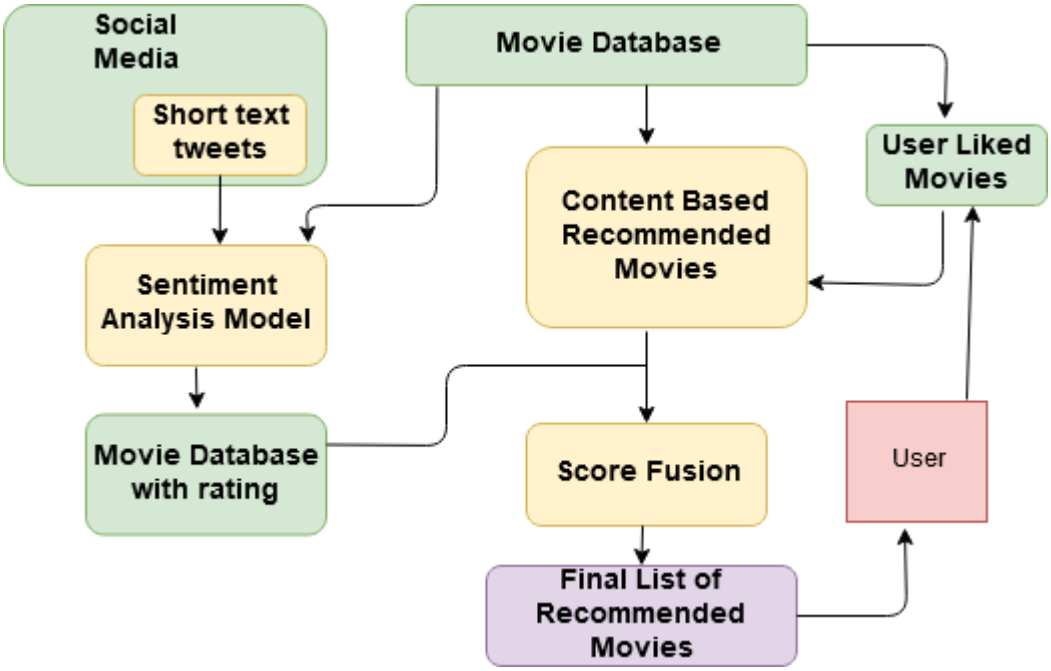


Fig 6.1: Purposed recommendation system model overview

After both scores is collected, the scores are fused. For a input movie with rating score of s_i , the recommendation score for a movie with rating score s_j is defined as follows

$$G(i,j) = D - |s_i - s_j| \quad \text{------(19)}$$

Where $G(i,j)$ is the recommendation score for movie j.

D is constant value for maximum value of rating scale, here D=10 since rating scale is form 1-10.

After calculating $G(i,j)$ for each movie in database, the scores are sorted and movies with higher score are recommended to user.

6.2 Implementation of Recommendation Model

6.2.1 Calculating Content based score of each movie

Term Frequency(TF):TF is simply the frequency of a word in a document.

Inverse Document Frequency (IDF): IDF is the inverse of the document frequency among the whole corpus of documents.

TF-IDF is used mainly because of two reasons: Suppose we search for "**the results of latest European Socccer games**" on Google. It is certain that "**the**" will occur more frequently than "**soccer games**" but the relative importance of **soccer games**is higher than the search query point of view. In such cases, TF-IDF weighting negates the effect of high frequency words in determining the importance of an item (document).

$$tfidf_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right) \quad \text{----------(20)}$$

Where,

$tf_{i,j}$ = Total number of occurrences of ***i*** in ***j***

N = Number of documents

df_i= Total number of documents containing ***i***

After calculating TF-IDF scores, to determine which items are closer to each other, rather closer to the user profile is done using the **Vector Space Model** which computes the proximity based on the angle between the vectors. In this model, each item is stored as a vector of its attributes (which are also vectors) in an **n-dimensional space** and the angles between the vectors are calculated to **determine the similarity between the vectors**. Next, the user profile vectors are also created based on his actions on previous attributes of items and the similarity between an item and a user is also determined in a similar way.

Following Fig(2) show a vector space model. Sentence 2 is more likely to be using Term 2 than using Term 1. Vice-versa for Sentence 1. The method of calculating this relative measure is calculated by taking the cosine of the angle between the sentences and the terms. The ultimate reason behind using cosine is that the **value of cosine will increase with decreasing value of the angle** between which signifies more similarity. The vectors are length normalized after which they become vectors of length 1 and then the cosine calculation is simply the sum-product of vectors.

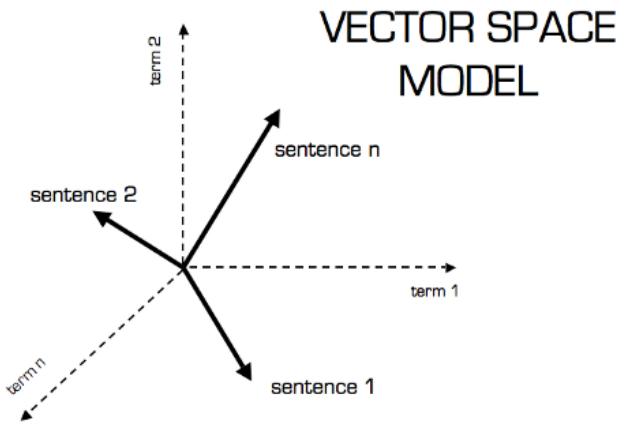


Fig 6.2: Vector Space Model of items

Cosine Distance Formula:

$$Cosine\ Similarity\ (A,B) = \frac{A.B}{||A||\times||B||} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} {A_i}^2} \times \sqrt{\sum_{i=1}^{n} {B_i}^2}} \quad \text{----------(21)}$$

Values range between -1 and 1, where -1 is perfectly dissimilar and 1 is perfectly similar.

Since TF-IDF score is already calculated, there is no need to implement this formula in this case, calculating dot product will directly give the cosine distance value.

6.2.2 Calculating the rating for each movie using reviews and generating final list of recommended movies

Collecting Tweets via Web Scraping:

For each movie a twitter review of $\boldsymbol{n}$ tweets are collected directly from Twitter API call. Hence for a movie database containing $\boldsymbol{m}$movies, will require to process $\boldsymbol{m} \times \boldsymbol{n}$number of tweets by passing them into the sentiment prediction model.

Cleaning of Tweets:

For each tweet it is necessary that it holds only proper texts that can be analyzer by the sentiment prediction model. Tweets are cleaned to free them from following:

I) Web Links (example: www.netflix.com)
II) Emoji Characters (example: 👌, 0x0001F44C)
III) Images

After cleaning the tweets, duplicate and short tweets are dropped.

Processing rating for each movie:

Following Fig(3) shows steps performed to get the rating of movie.

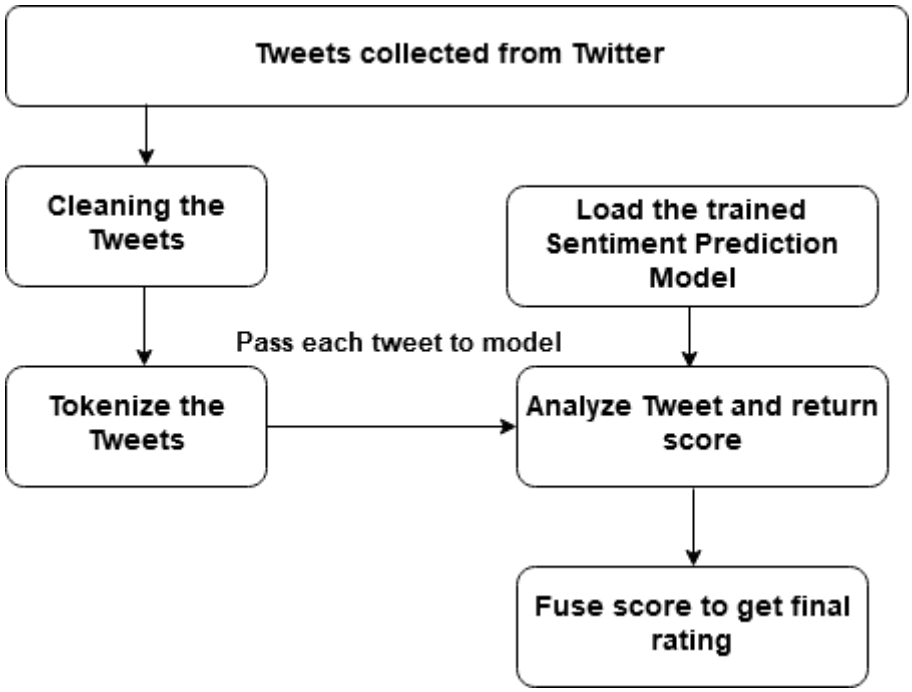


Fig 6.3: Block diagram to show flow of processing of tweets to get rating.

For each movie the percentage of positive tweets are considered as rating which can be scaled down to scale of (1-10).

After both scores (rating of selected movie and rating of all other movies) is collected, the scores are fused. For a input movie with rating score of $\boldsymbol{s_i}$ **,** the recommendation score for a movie with rating score$\boldsymbol{s_j}$is defined as follows

$$\boldsymbol{G(i,j) = D - |s_i - s_j|} \quad \text{------(22)}$$

Where $\boldsymbol{G(i,j)}$is the recommendation score for movie $\boldsymbol{j}$**.** Sorting the final recommendation score will give the most relevant movies that can be sent as recommendation.

7. Software System Structure

This part describes the overall system structure that is going to be implemented and also the life cycle model which is followed for designing this software.

7.1 Life Cycle Model

The life cycle of a software represents the series of identifiable stages through which it evolves during its life time. With this knowledge of a software life cycle it can further define Software development life cycle(SDLC) model(also called software life cycle model and software development process model. A life cycle model graphically represents the different phases through which a software evolves. It is usually accompanied by a textual description of the different activities that need to be carried out during each phase.

7.1.1 Need of Software Life Cycle Model

The development team must identify a suitable life cycle model for the particular project and then adhere to it. Without using of a particular life cycle model the development of a software product would not be in a systematic and disciplined manner. When a software product is being developed by a team there must be a clear understanding among team members about when and what to do. Otherwise it would lead to chaos and project failure. A software life cycle model defines entry and exit criteria for every phase. A phase can start only if its phase-entry criteria have been satisfied. So without software life cycle model the entry and exit criteria for a phase cannot be recognized. Without software life cycle models it becomes difficult for software project managers to monitor the progress of the project.

7.1.2 Different Software Life Cycle Models

Many life cycle models have been proposed so far. Each of them has some advantages as well as some disadvantages. A few important and commonly used life cycle models are as follows:

1. Classical Waterfall Model.

2. Iterative Waterfall Model.

3. Prototyping Model.

4. Evolutionary Model.

5. Spiral Model.

In this project for Software structure design Evolutionaly Model will be used.

7.2 Evolutionary Model

Evolutionary model is a combination of Iterative and Incremental model of software development life cycle. Delivering your system in a big bang release, delivering it in incremental process over time is the action done in this model. Some initial requirements and architecture envisioning need to be done.

It is better for software products that have their feature sets redefined during development because of user feedback and other factors. The Evolutionary development model divides the development cycle into smaller, incremental waterfall models in which users are able to get access to the product at the end of each cycle.

Feedback is provided by the users on the product for the planning stage of the next cycle and the development team responds, often by changing the product, plan or process. Therefore, the software product evolves with time.

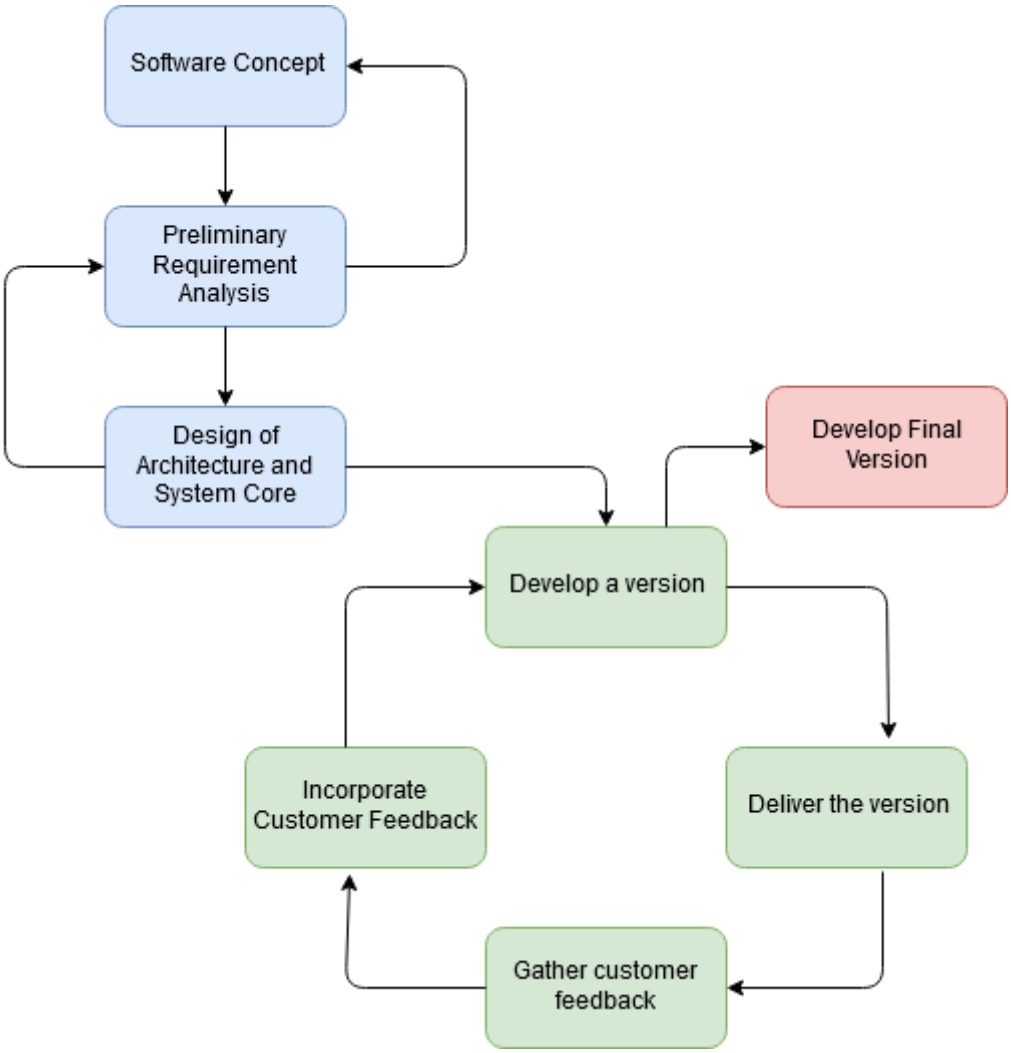


Fig 7.1: Evolutionary Model

Evolutionary model suggests breaking down of work into smaller chunks, prioritizing them and then delivering those chunks to the customer one by one. The number of chunks is huge and is the number of deliveries made to the customer. The main advantage is that the customer's confidence increases as he constantly gets quantifiable goods or services from the beginning of the project to verify and validate his requirements. The model allows for changing requirements as well as all work in broken down into maintainable work chunks.

7.2.1 Need for Evolutionary Model

I) In evolutionary model, a user gets a chance to experiment partially developed system.

II) It reduces the error because the core modules get tested thoroughly.

III) It is used in large projects where you can easily find modules for incremental implementation. Evolutionary model is commonly used when the customer wants to start using the core features instead of waiting for the full software.

IV) Evolutionary model is also used in object oriented software development because the system can be easily portioned into units in terms of objects.

7.3 Software Requirements Specification (SRS)

The SRS document usually contains all the user requirements in a structured though an informal form. The SRS document should describe the system to be developed as a black box, and should specify only the externally visible behavior of the system. For this reason, the SRS document is also called the black-box specification of the software being developed. An SRS document should clearly document the following aspects of a software:

1. Functional Requirements:

- The whole set up should be connected to a network
- To store the user authentication details and activities database is required
- Server system should be powerful enough to complete the execution successfully.

2. Non-Functional Requirements:

- Software maintenance that is update to new version of the software must be made.
- For server process the hardware should be powerful enough to peform heavy computation for the requests maded through the network. In case of any failure system should show the proper logs to debug the problem.

7.4 Context Diagram

The context diagram is the most abstract data flow representation of a system. It represents the entire system as a single bubble. This bubble is labeled according to the main function of

the system. The various external entities with which the system interacts and the data flow occurring between the system and the external entities are also represented. The data input to the system and the data output from the system are represented as incoming and outgoing arrows. These data flow arrows should be annotated with the corresponding data names.

The name context diagram is well justified because it represents the context in which the system is to exist, i.e. the external entities who would interact with the system and the specific data items they would be supplying the system and the data items they would be receiving from the system. The context diagram is also called as the level 0 DFD.

The bubble in the context diagram is annotated with the name of the software system being developed.This is expected since the purpose of the context diagram is to capture the context of the system rather than its functionality.

The figure below shows context diagram designed for the software system.

- The class USER DATA and RECOMMENDED MOVIES are external entities.
- The entire system RECOMMENDATION SYSTEM is represented in single bubble.
- The data flow is shown using the pointed arrows.

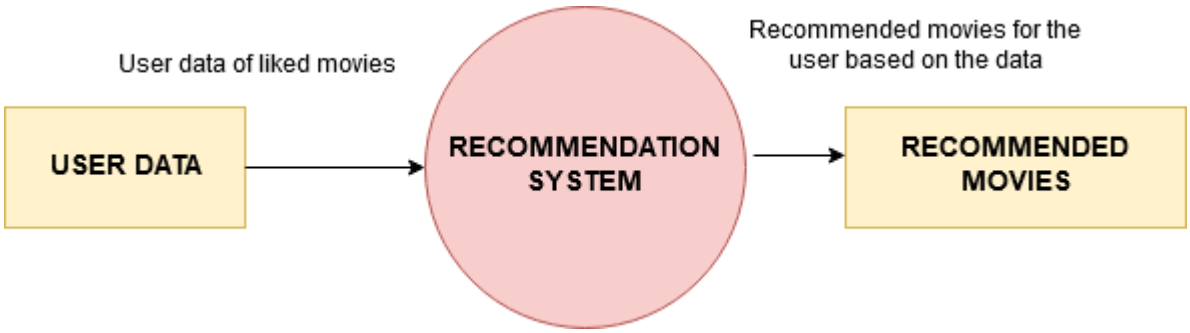


Fig 7.2: Context Diagram of Recommendation System

7.5 Data Flow Diagram (DFD)

Data flow diagram is a hierarchical graphical representation of data flow in an information system. It is capable of depicting incoming data flow, outgoing data flow and stored data. The DFD does not mention anything about how data flows through the system.

7.5.1 Types of DFD

Data Flow Diagrams are of two types:

1. Logical DFD: This type of DFD concentrates on the system process and flow of data in the system.

2. Physical DFD: This type of DFD concentrates on the mapping of data flow onto respective physical system.

7.5.2 DFD Components

DFD can represent Source, destination, storage and flow of data using the following set of components

1. Entities: Entities are source and destination of information data. Entities are represented by rectangles with their respective names.

2. Process: Activities and action taken on the data are represented by Circle or Round-edged rectangles.

3. Data Storage: There are two variants of data storage - it can either be represented as a rectangle with absence of both smaller sides or as an open-sided rectangle with only one side missing.

4. Data Flow: Movement of data is shown by pointed arrows. Data movement is shown from the base of arrow as its source towards head of the arrow as destination.

7.5.3 Importance of DFD

- The DFD technique is so popular is probably because of the fact that DFD is simple to understand and use.

- Starting with a set of high-level functions that a system performs, a DFD model hierarchically represents various sub-functions. In fact, any hierarchical model is simple to understand. Human mind is such that it can easily understand any hierarchical model of a system because in a hierarchical model, starting with a very simple and abstract model of a system, different details of the system are slowly introduced through different hierarchies.

- The data flow diagramming technique also follows a very simple set of intuitive concepts and rules.

- DFD is an elegant modeling technique that turns out to be useful not only to represent the results of structured analysis of a software problem, but also for several other applications such as showing the flow of documents or items in an organization.

7.6 Level 1 DFD

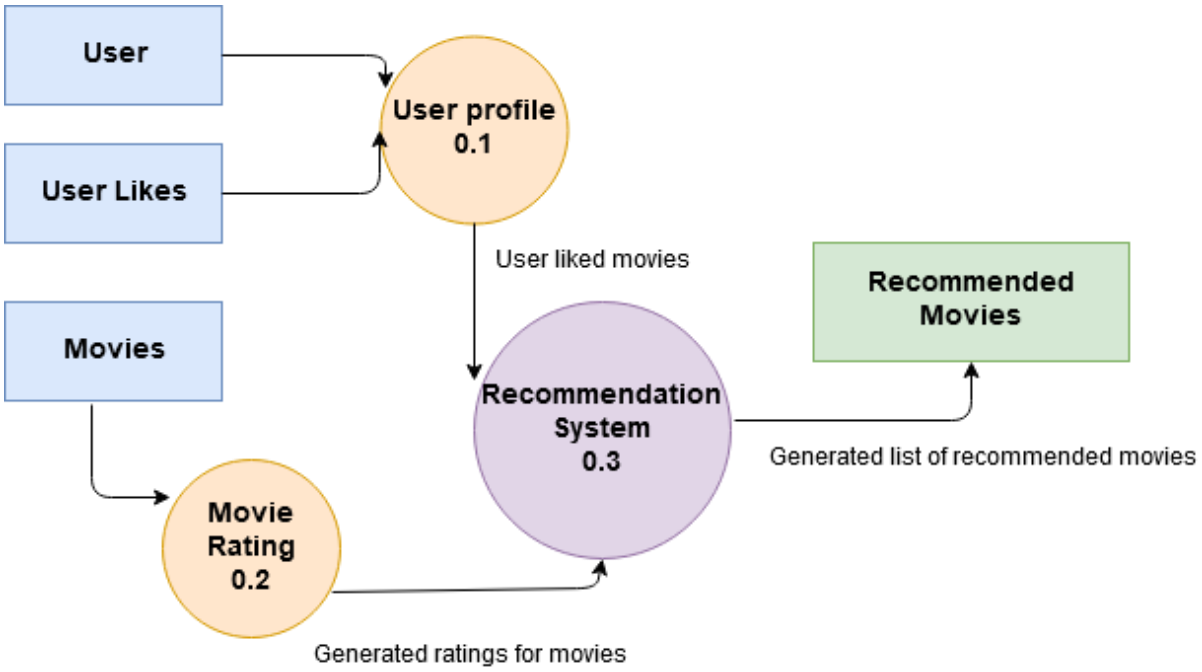


Fig 7.3: Level 1 DFD

The above figure is demonstrated below:

- User profile is generated based on user likes.
- All movies are processed to generated the ratings for movies.
- Recommendation system fuses the user likes with movie data to generate the recommended movies.

7.7 Level 2 DFD

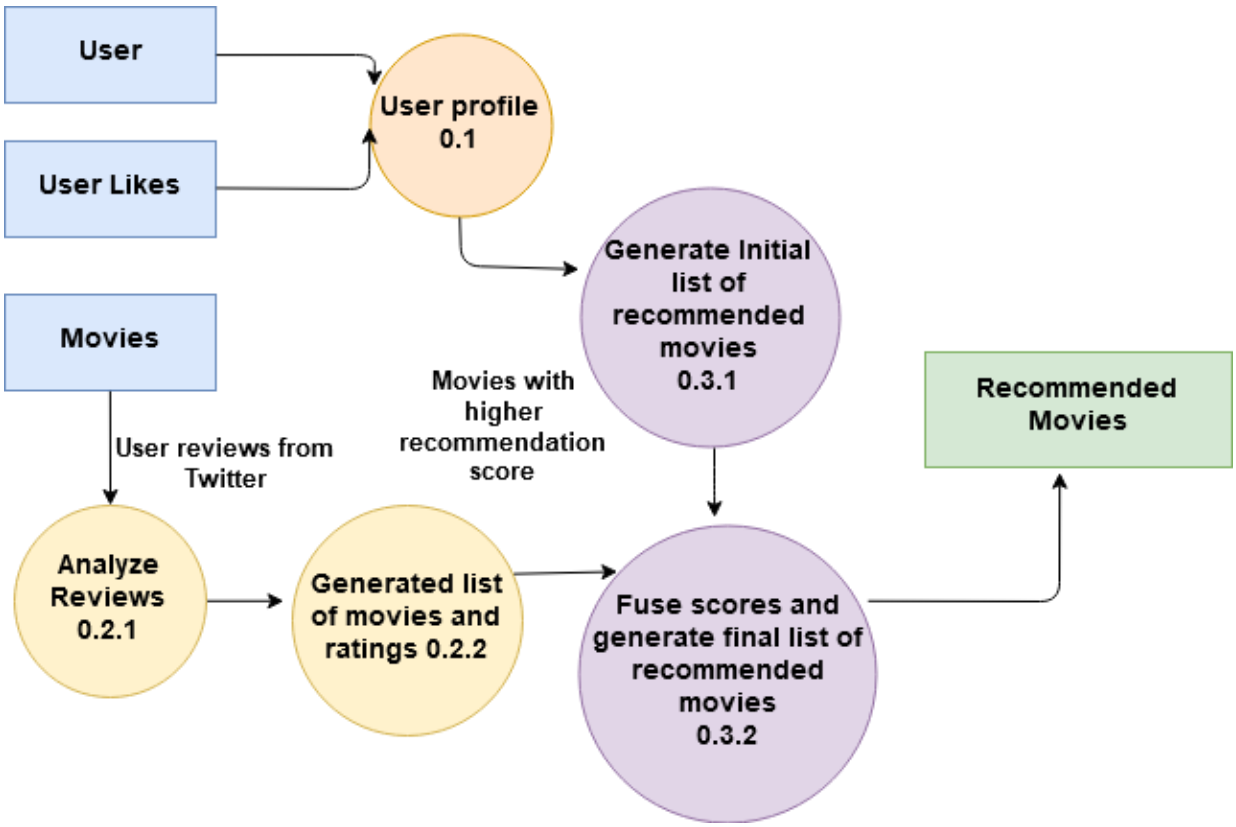


Fig 7.4: Level 2 DFD

The above figure is discussed below:

- The reviews are collected form tweets.
- Each tweet is analyzed for a particular movie to generate a rating for that movie, this is done for all movies to generate a rating database.
- Based on user likes an initial recommended movies list is generated.
- Both generated list of movies and movies rating scores are fused and movies with highest score are treated as recommended movies.

7.8 Level 3 DFD

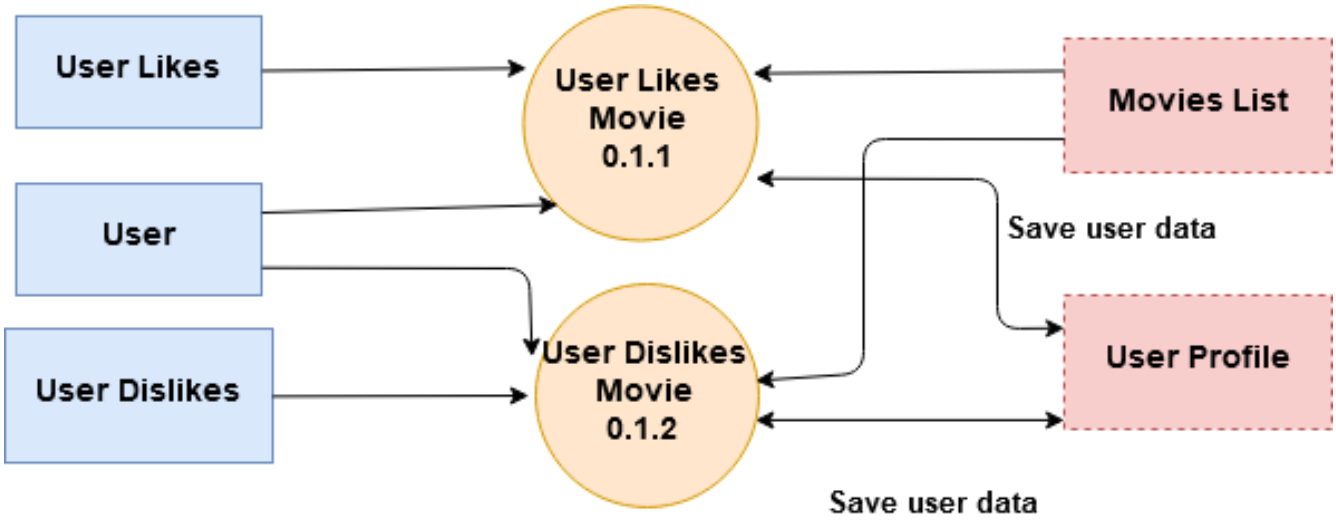


Fig 7.5: Level 3 DFD

The above figure is described below:

- The user likes any movie from movie list, response is recorded and user profile is updated.
- The user dislikes any movie from list, response is recorded and user profile is updated.
- The final user profile will server as input for recommendation system.

8. Database Design

Here the database design for the whole system is depicted using Entity Relationship (E-R) diagram. Before going further, first we need to know what is an E-R diagram and why it is so important in designing a database.

An ”Entity Relationship Diagram (ERD)” shows the relationships of entity sets stored in a database. An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases.

• For database designing Top Down approach is known as E-R diagram.

• Here we are focusing on top down approach of designing database. It is a graphical technique, which is used to convert the requirement of the system to graphical representation. So that it can become well understandable.

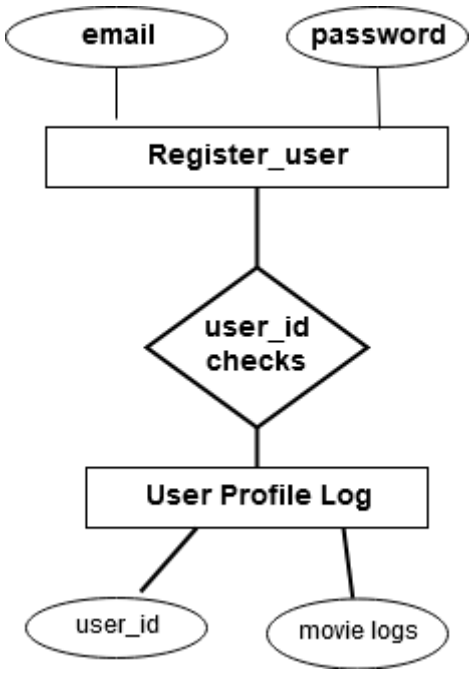


Fig 8.1: E-R diagram of the registered user schema with user log schema

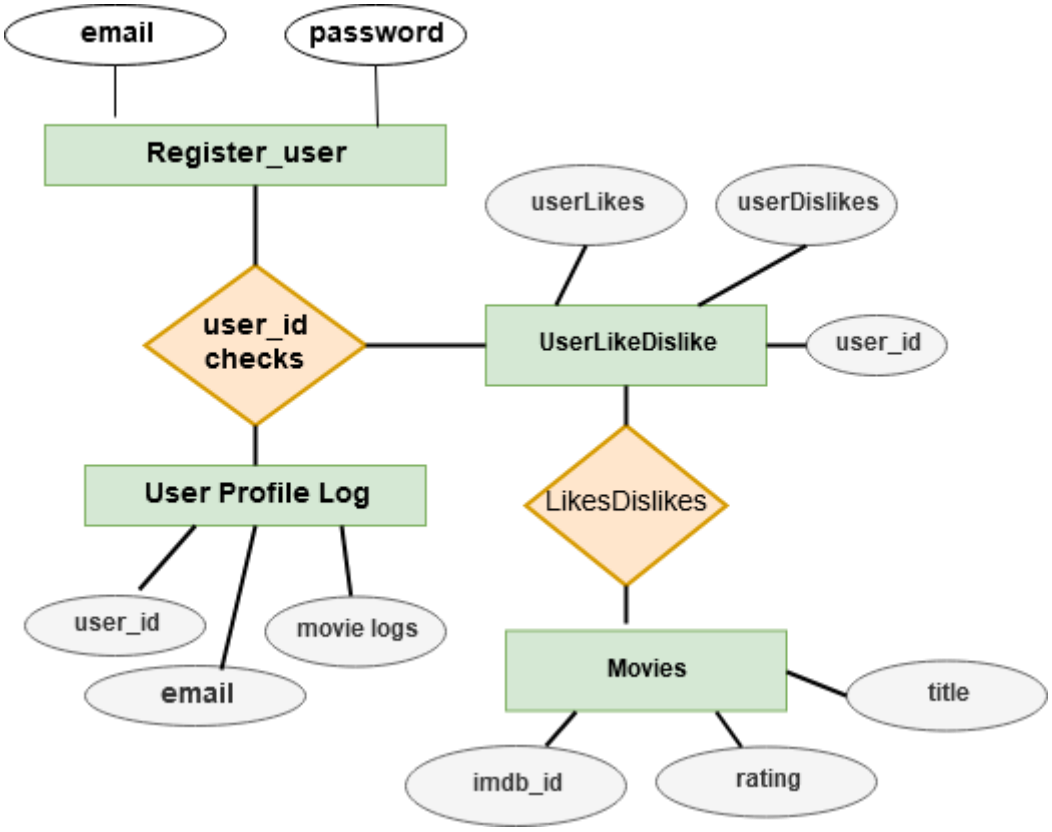


Fig 8.2: ER Diagram of the whole system

8.1 Database schema

This part describes the underlying database schema of Recommendation System

The database schemas are:

1. register_user.

2. user_profile_log

3. userLikeDislike

4. movies

8.1.1 register_user

This schema contains the email and password mappings to user_id of a user.

email: It is primary key that mapps to a password.

password: User generated password

8.1.2 user_profile_log

This schema contains the userID and userLogs mappings.

user_id: Primary key to uniquely identify user.

email: Email corresponding to unique user_id.

movie_logs: List of movies user has interacted.

8.1.3 userLikeDislike

This schema contains the following attributes:

user_id: Foreign key from user_profile_log.

user_likes: Liked movies by user.

user_dislikes: Disliked movies by user.

8.1.4 Movies

This schema contains the following attributes:

imdb_id: Primary key to uniquely identify movie.

rating: Rating on scale 1-10 of the movie.

title: Title of movie.

9. Implementation: Final System

After Sentiment Analysis model and Recommendation system has been implemented. The Final system that will be used as backend to perform the necessary computation needs to be deployed on a computation platform from where functions can be called to get results.

9.1 Introduction to Google Cloud

Google provides a cloud computing platform to train, deploy and manage the Deep Neural Networks and other applications. Some of the basic services provided by Google Cloud has been listed in below Fig(1)

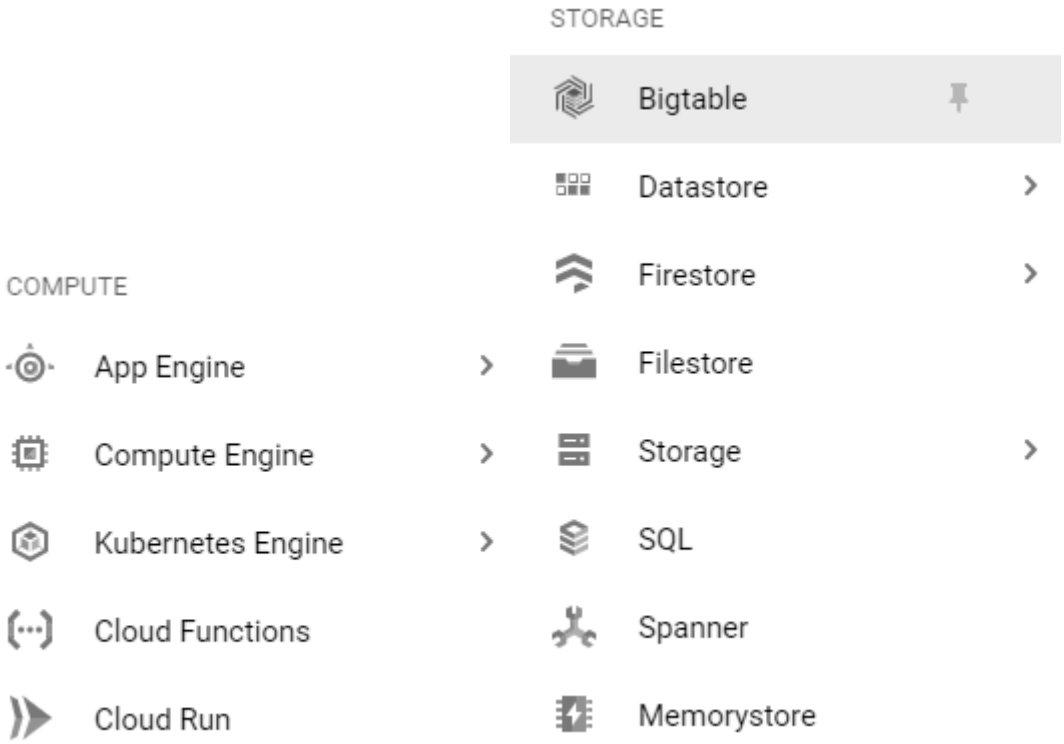


Fig 9.1: Basic services provided by Google Cloud

Other similar services are Amazon AWS, Digital Ocean, Microsoft Azure, but since Firebase(User here for user authentication) is also managed by Google and it supports better computation vs price ratio Google Cloud has been used.

9.2 Google Cloud App Engine for Model Deployment

For our trained sentiment prediction model and recommendation model Google App Engine is ideal. It provides following

1. Trace of network traffic of all requests made to the server process.
2. Flexible hardware configuration on selected regions.
3. Support of Logging of errors.
4. Pausing the deployed model (to reduce the cost when new version of App is being developed or is being ideal)
5. Connecting more storage bucket depending on requirements.
6. Scaling up or down the hardware configuration according to need. (Upgrading due to more users or vice-versa)

Deployment Process to Google App Engine:-

I) New project is selected from console or existing project form list

Fig 9.2: Project selection from console

II) After the selection of ProjectID from console. All files are prepared from local machine with writing necessary methods.
   a. Flask with REST-API can be used to write the server code to perform network requests.
   b. After writing and testing the methods on LocalHost, files are ready to be uploaded to Google Cloud

III) For uploading the project files, few files are needed to be created which is required by Google App Engine

a)app.yaml : Contains the Hardware configuration of server.

Example of app.yaml used in this implementation:-

```
#the app runtime will be on python
runtime: python
#App Engine flexible environment automatically scales your app up
and down while balancing the load.
env: flex
```

```
#timeout is 120 seconds, "-t 120" is not mentioned it will be by
default of 30 secs
entrypoint: gunicorn -t 120 -b :$PORT main:app
runtime_config:
python_version: 3
manual_scaling:
instances: 1
# 4 cpu cores with 20 gb ram
# for a single core max ram can be 6.5 GB, hence for 4 cores
4x6.5 is max
resources:
cpu: 4
memory_gb: 20
disk_size_gb: 20
handlers:
- url: /.*
script: auto
```

b)requirements.txt : Contains list of libraries that needed to be installed at server compute environment.

Example of requirements.txt used in this implementation:-

```
Flask==1.0.2
numpy==1.14.6
https://download.pytorch.org/whl/cpu/torch-1.1.0-cp36-cp36m-
linux_x86_64.whl
https://download.pytorch.org/whl/cpu/torchvision-0.3.0-cp36-
cp36m-linux_x86_64.whl
gunicorn==19.9.0
Flask-RESTful
tweepy==3.7.0
pandas==0.24.2
sklearn
psutil
```

IV) After add files are ready Google Cloud SDK Command is called to deploy the app

```
F:\Google Cloud Major Project Final>gcloud app deploy
```

9.3 Firebase for authentication

Firebase is another service provide by Google to connect the Android/iOS Apps to google services. Some basics services provided by Firebase are:

1. User authentication
2. Cloud Store
3. ML Kit
4. Real time Database

Since Google App Engine has been already connected with a storage bucket, we only need Firebase to authenticate users into Android App or register new users. Library functions can be called to perform requests.

To setup firebase for project:

I) Select the existing project from console or create a new one

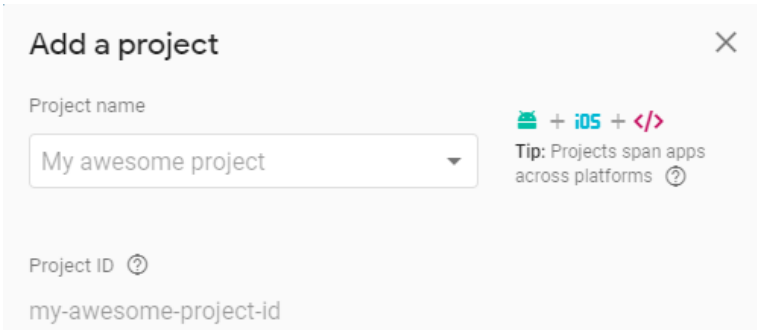


Fig 9.3: Firebase project selection form console

II) To connect the android app,In the center of the Firebase console's project overview page, click the **Android** icon to launch the setup workflow.

a. If you've already added an app to your Firebase project, click **Add app** to display the platform options.

b). .Enter your app's application ID in the **Android package name** field.

c). Follow the instruction to download JSON Configuration file and put that into android app directory and edit Build Gradle Accordingly.

III) All register and login functions can be accessed form documentation. Each user registered can be seen from the Firebase console.

Identifier	Providers	Created	Signed In	User UID
bhakta@gmail.com		Jul 18, 2019	Jul 18, 2019	EOzQxdUjmgUtuLvvQvt1nKMBi7s1
manugond@gmail.com		Jul 21, 2019	Jul 21, 2019	ZGt24TGdCIcedWhvPDIb1BJlXl22
arupdas@gmail.com		Jul 16, 2019	Jul 16, 2019	kWrh5El5nnPTGI1qw3YNtTqqGUI2
arup@gmail.com		Jul 16, 2019	Jul 21, 2019	uJQRE8DjYoUodtel9qtVwziYCbi1
sample@gmail.com		Jul 16, 2019	Jul 16, 2019	ue5BVBLbzkXyPFWCS3SGgWSh9...

Rows per page: 50 1-5 of 5

Fig 9.4: User authentication database from firebase console

9.4 Introduction to Android

Android is a Linux based operating system it is designed primarily for touch screen mobile devices such as smart phones and tablet computers. The operating system have developed a lot in last 15 years starting from black and white phones to recent smart phones or mini computers. One of the most widely used mobile OS these days is android. The android is software that was founded in Palo Alto of California in 2003.

The architecture of OS is defined in Fig(5)

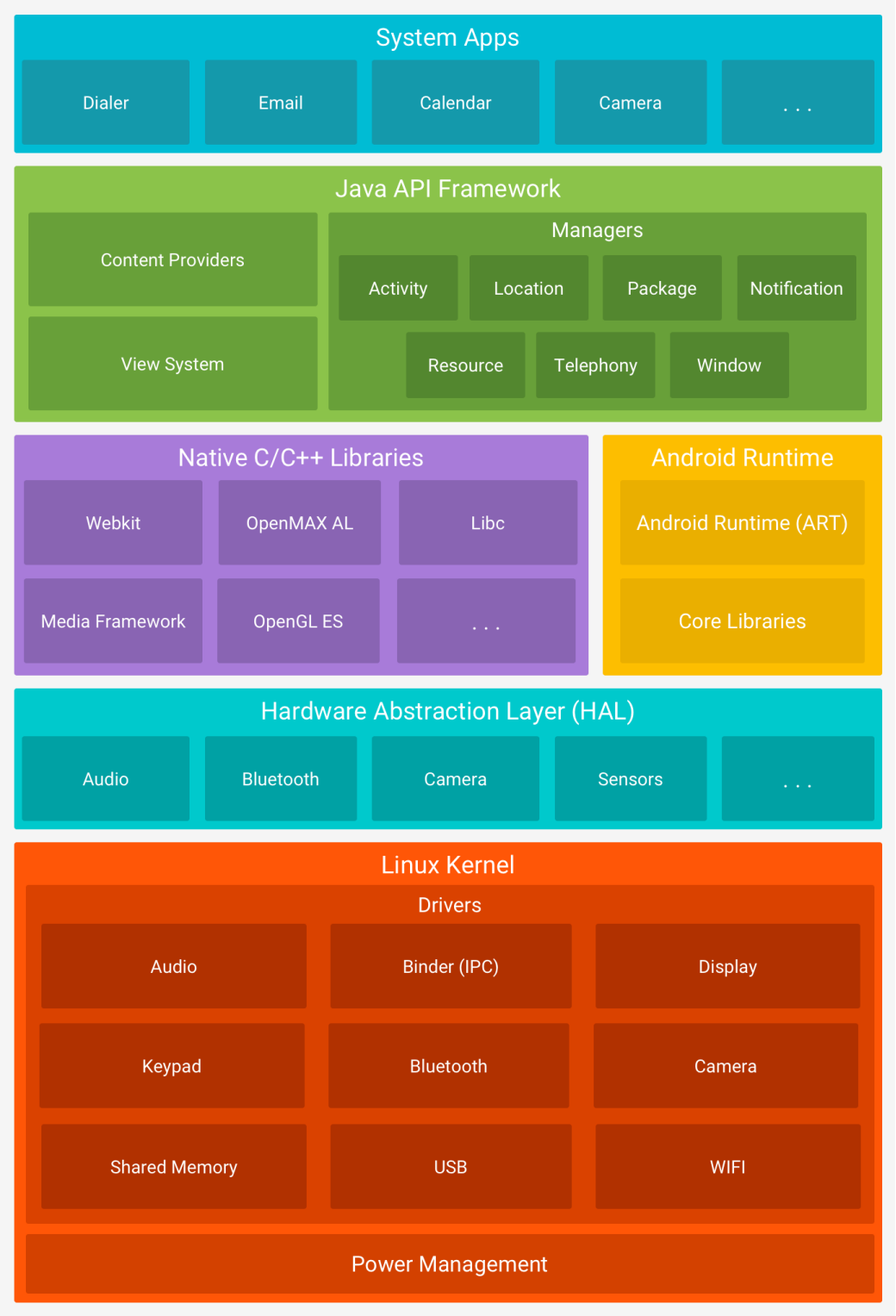


Fig 9.5: Android OS Architecture

Java Code that needs to be implemented here will use the Java API Framework and Android Runtime. Calls to the Hardware Abstraction Layer(HAL) will be made through the inherited methods.

Lifecycle of an activity in Android:

Fig(6) shows activity lifecycle of activity. As the user begins to leave the activity, the system calls methods to dismantle the activity. In some cases, this dismantlement is only partial; the activity still resides in memory (such as when the user switches to another app), and can still come back to the foreground. If the user returns to that activity, the activity resumes from where the user left off. The system's likelihood of killing a given process—along with the activities in it—depends on the state of the activity at the time.

Depending on the complexity of your activity, you probably don't need to implement all the lifecycle methods. However, it's important that you understand each one and implement those that ensure your app behaves the way users expect

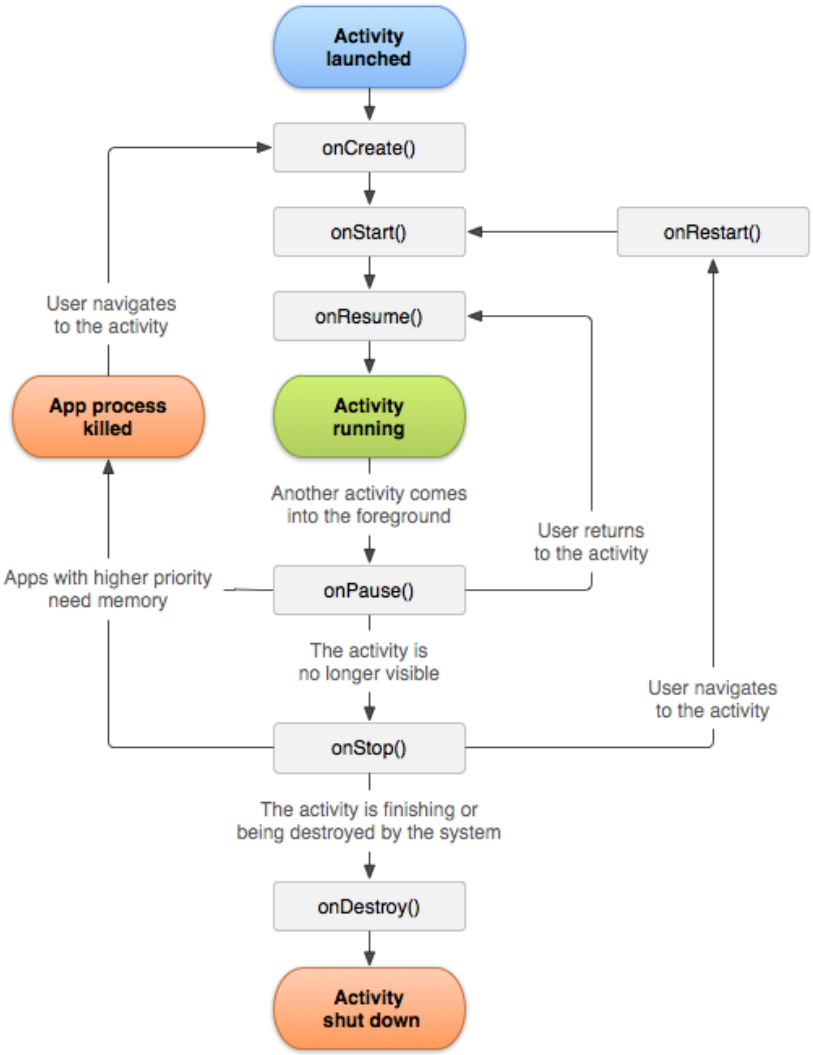


Fig 9.6: Activity lifecycle of android activity

onCreate()

You must implement this callback, which fires when the system first creates the activity. On activity creation, the activity enters the *Created* state. In the onCreate() method, you perform basic application startup logic that should happen only once for the entire life of the activity. For example, your implementation of onCreate() might bind data to lists, associate the activity with a ViewModel, and instantiate some class-scope variables. This method receives the parameter savedInstanceState, which is a Bundle object containing the activity's previously saved state. If the activity has never existed before, the value of the Bundle object is null.

If you have a lifecycle-aware component that is hooked up to the lifecycle of your activity it will receive the ON_CREATE event. The method annotated with @OnLifecycleEvent will be called so your lifecycle-aware component can perform any setup code it needs for the created state.

onStart()

When the activity enters the Started state, the system invokes this callback. The onStart() call makes the activity visible to the user, as the app prepares for the activity to enter the foreground and become interactive. For example, this method is where the app initializes the code that maintains the UI.

When the activity moves to the started state, any lifecycle-aware component tied to the activity's lifecycle will receive the ON_START event.

The onStart() method completes very quickly and, as with the Created state, the activity does not stay resident in the Started state. Once this callback finishes, the activity enters the *Resumed* state, and the system invokes the onResume() method.

9.5 Connecting Android App with Deployed Model on Cloud

The deployed model on cloud can process requests when API url is called.

The implemented function that their URLs are:

1. **Sentiment Analysis on single text:**https://<app-address>/sentiment/<text>
2. **Analyze Twitter Sentiment on a text:**https://<app-address>/analyze/<query>
3. **Add movie to user likes:**https://<app-address>/addLike/<UserID+likedMovie>
4. **Add movie to user dislikes:**https://<app-address>/addDislike/<dislikedMovie>
5. **Get recommended movies:** https://<app-address>/getRecommendation/<userID>

The method can be written to perform network request from an activity by inheriting **AsyncTask** which is used to perform request in background without affecting main UI thread. After the result is received the corresponding view layouts are created on screen to show the result.

Overall connected system is described in Fig(9.7).

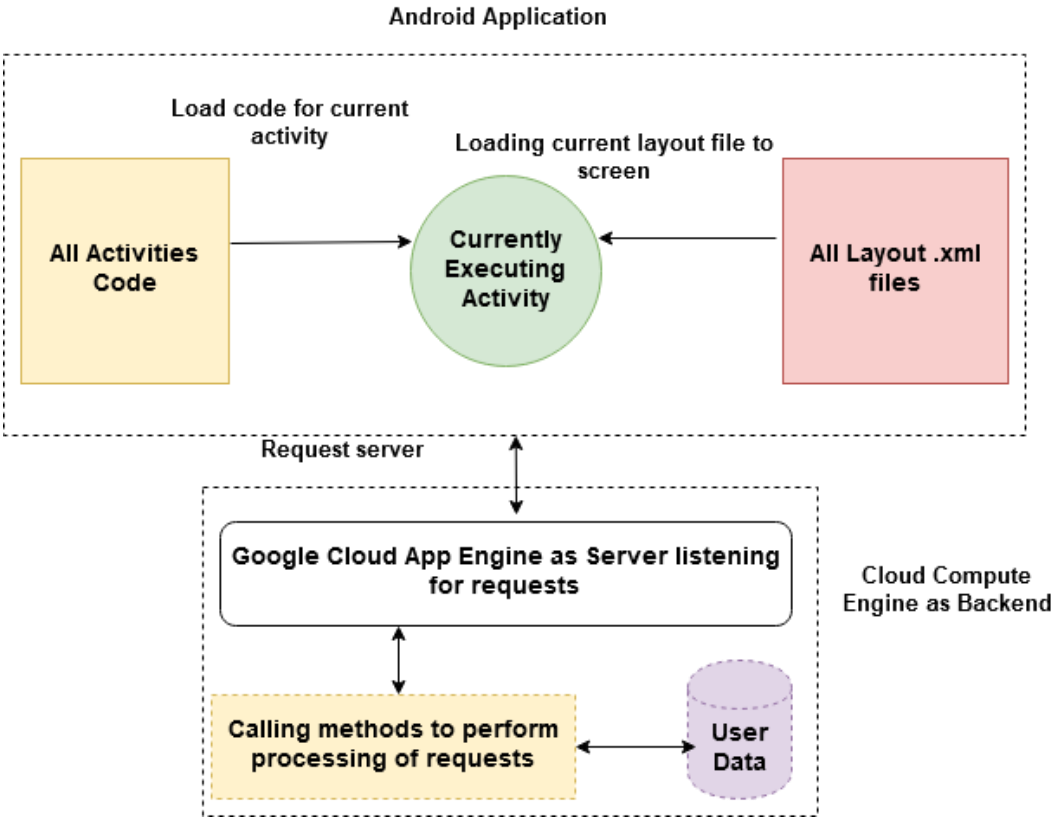


Fig (9.7): Overview of connecting the Android App to the services provided by Google Cloud App Engine

10. Test and Verification of Application

10.1 Registration Screen Testing

Test Case ID	Input Data description	Expected Result	Actual Result	Status	Figure ID
TR001	Valid email & password	Registration Successful	Registration Successful	Pass	Fig(TR1)
TR002	Invalid email & valid password	Registration Failed	Registration Failed	Pass	Fig(TR2)
TR003	Valid email & invalid password	Registration Failed	Registration Failed	Pass	Fig(TR3)
TR004	Already Registered information	Registration Failed	Registration Failed	Pass	Fig(TR4)

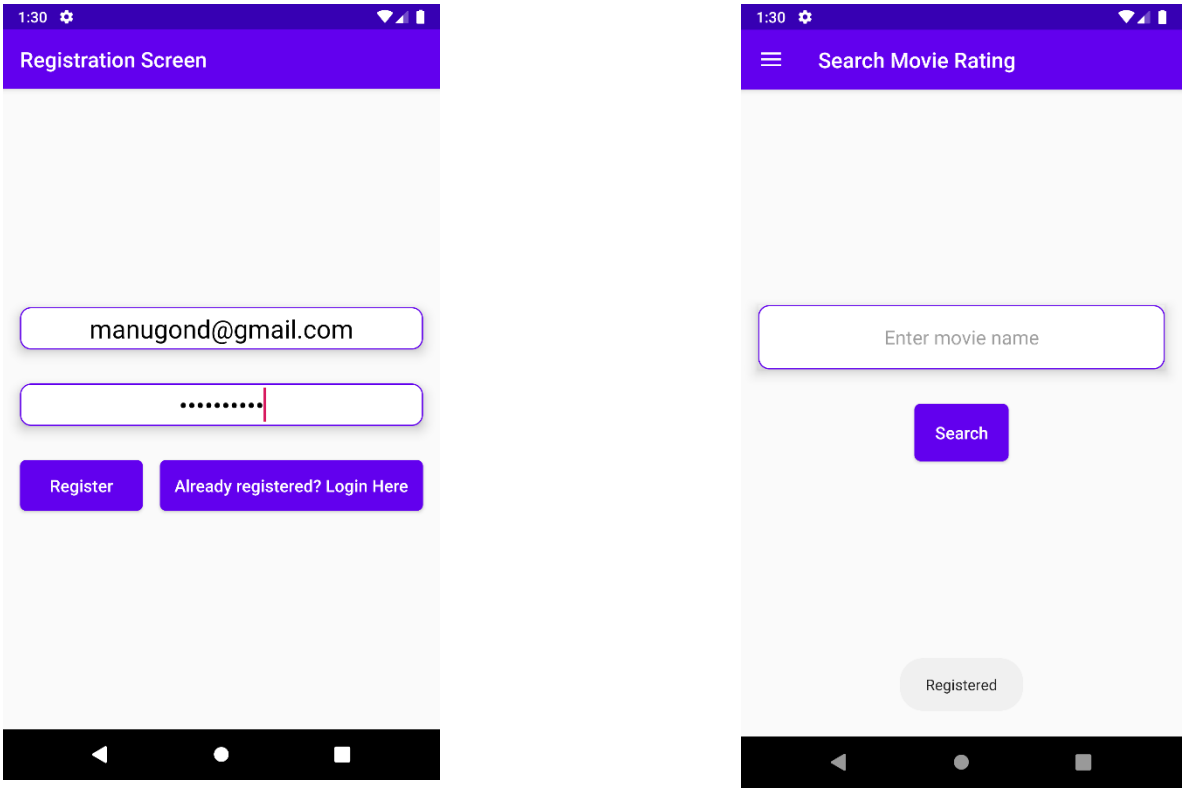
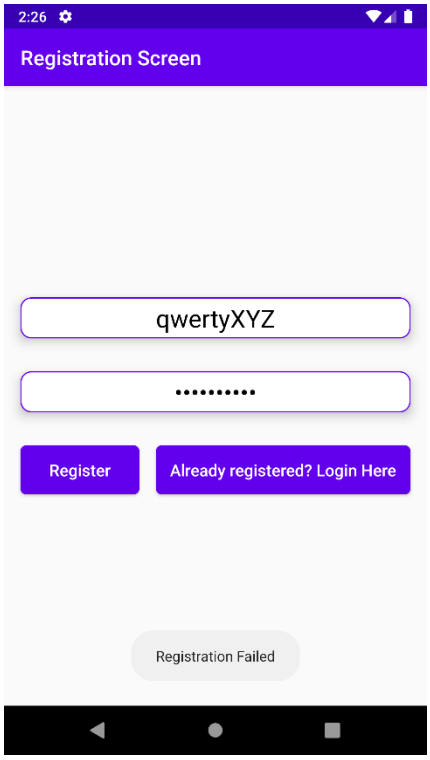
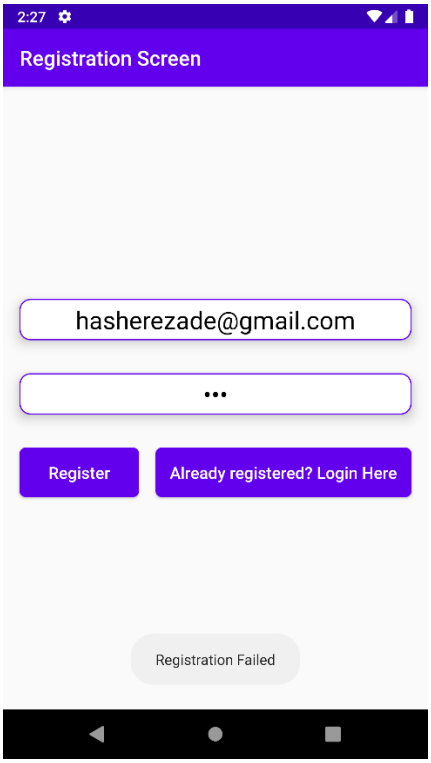


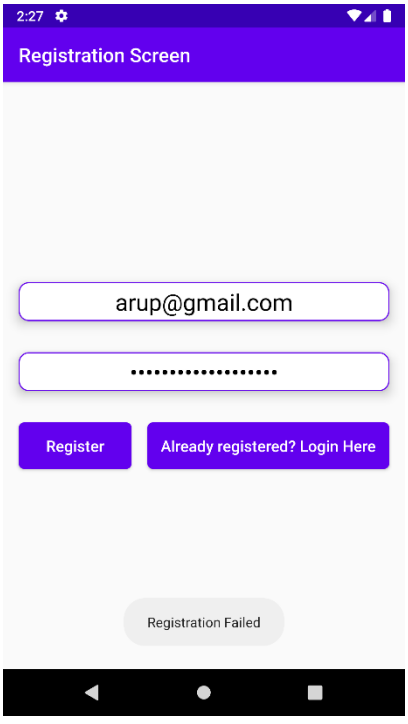
Fig (TR1)



Fig(TR2)



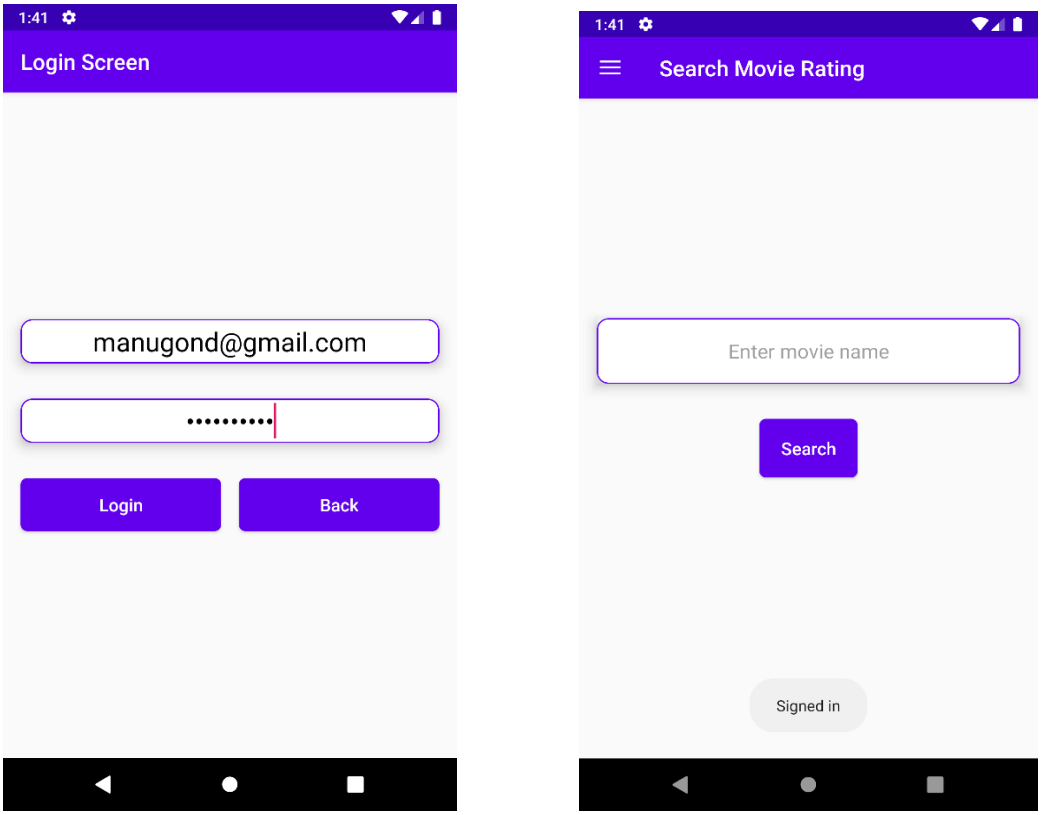
Fig(TR3)



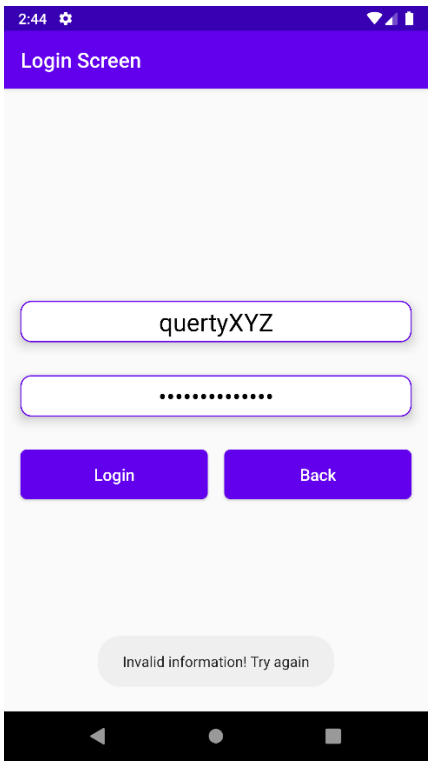
Fig(TR4)

10.2 Login Screen Testing

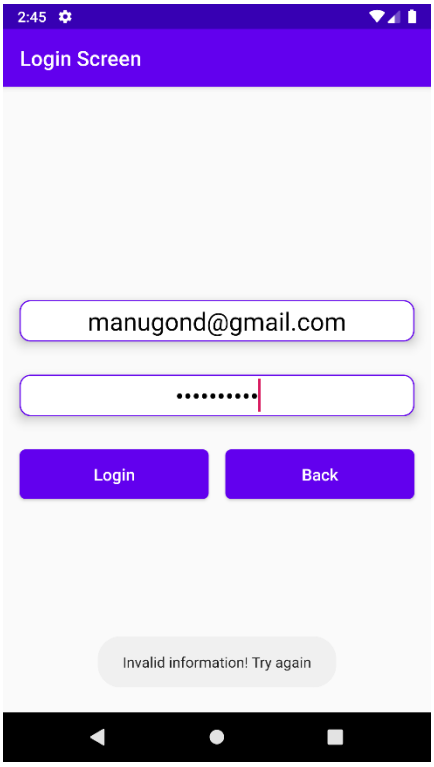
Test Case ID	Input Data description	Expected Result	Actual Result	Status	Figure ID
TL001	Valid email & password	Login Successful	Login Successful	Pass	Fig(TL1)
TL002	Invalid email & valid password	Login Failed	Login Failed	Pass	Fig(TL2)
TL003	Valid email & invalid password	Login Failed	Login Failed	Pass	Fig(TL3)
TL004	Invalid email & password	Login Failed	Login Failed	Pass	Fig(Tl4)



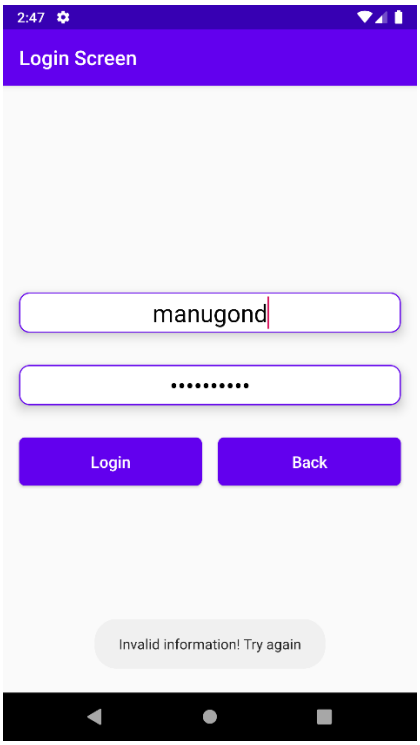
Fig(TL1)



Fig(TL2)



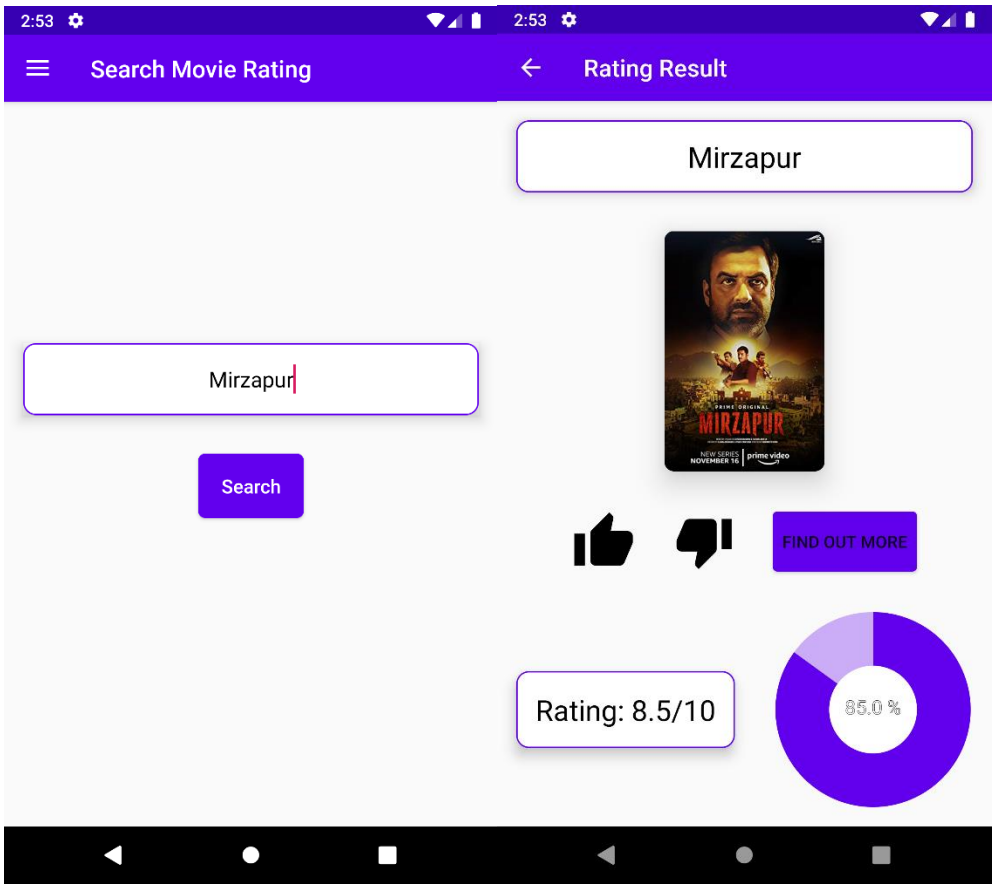
Fig(TL3)



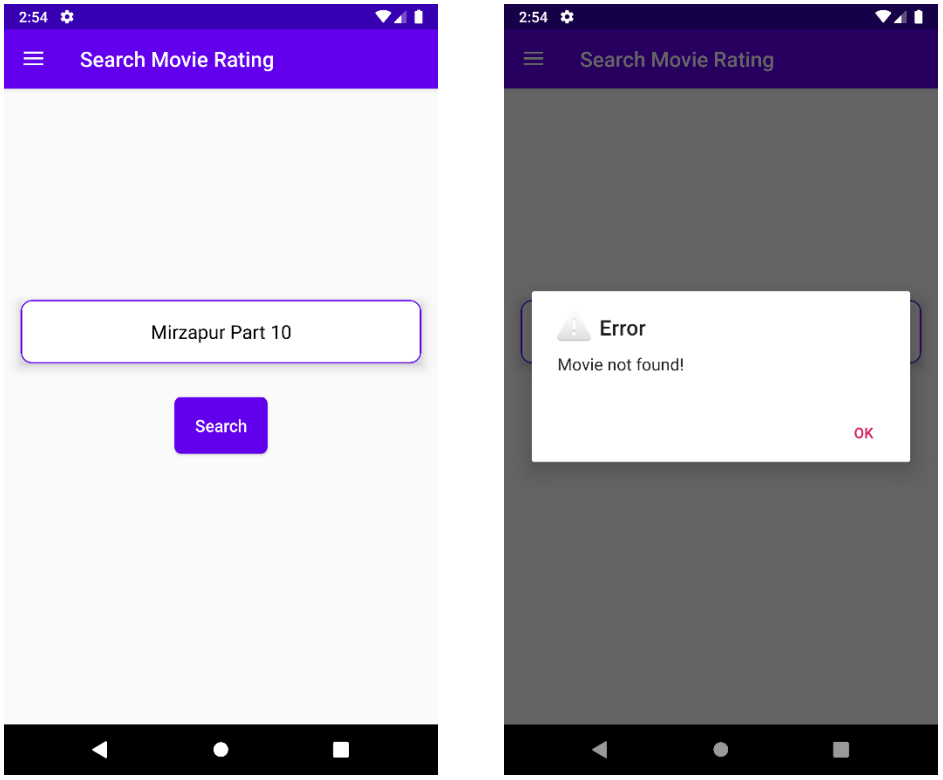
Fig(TL4)

10.3 Movie Rating Search Testing

Test Case ID	Input Data description	Expected Result	Actual Result	Status	Figure ID
TS001	Valid movie name	Result of rating	Result of rating	Pass	Fig(TS1)
TS002	Invalid movie name	Error Message	Error Message	Pass	Fig(TS2)



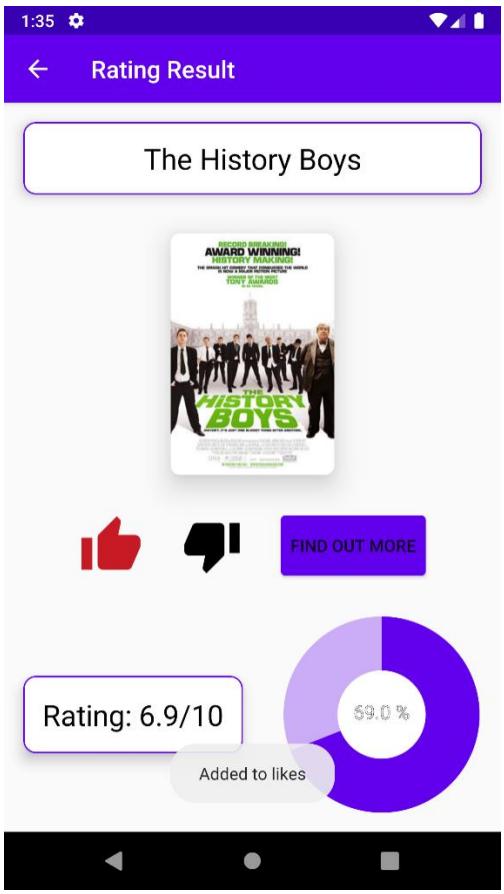
Fig(TS1)


2:54
Search Movie Rating
Mirzapur Part 10
Search
2:54
Search Movie Rating
Error
Movie not found!
OK

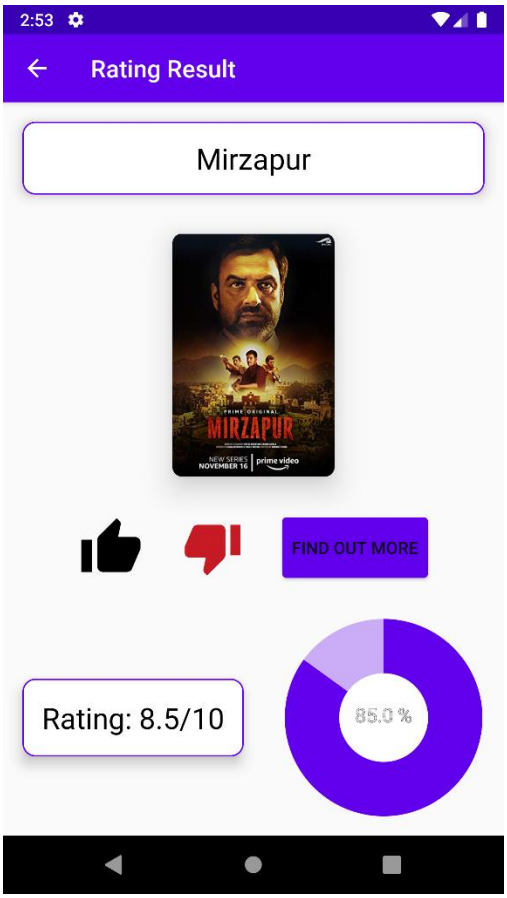
Fig(TS2)

10.4 Like/Dislike on Movie Testing

Test Case ID	Input Data description	Expected Result	Actual Result	Status	Figure ID
TLD001	Like Movie	Added to liked list	Added to liked list	Pass	Fig(TLD1)
TLD002	Dislike Movie	Added to disliked list	Added to disliked list	Pass	Fig(TLD2)



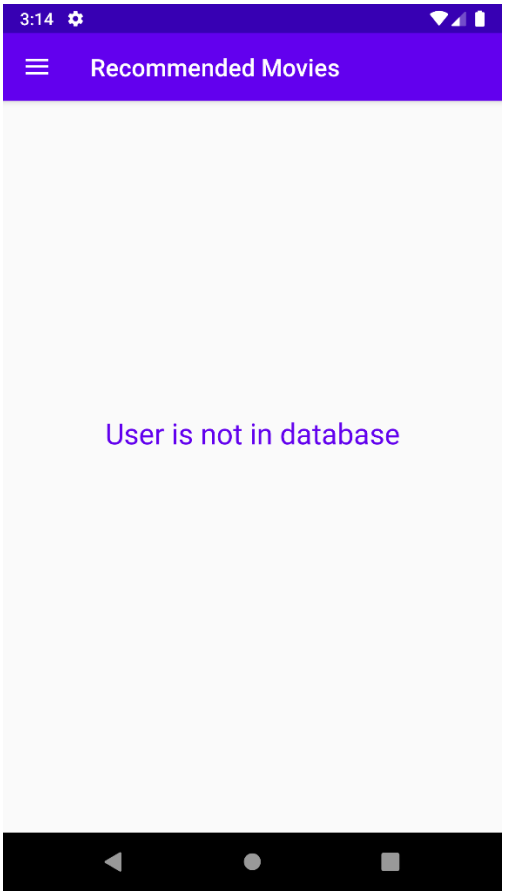
Fig(TLD1)



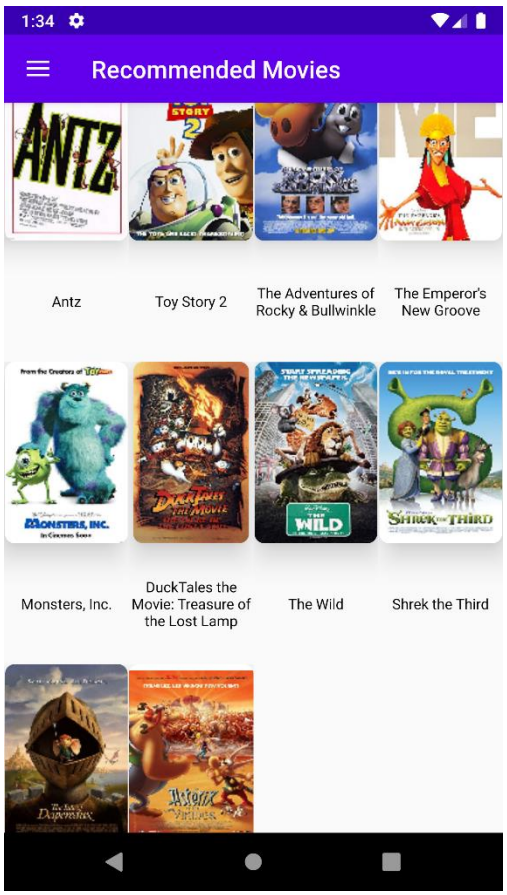
Fig(TLD2)

10.5 Recommended Movie Testing

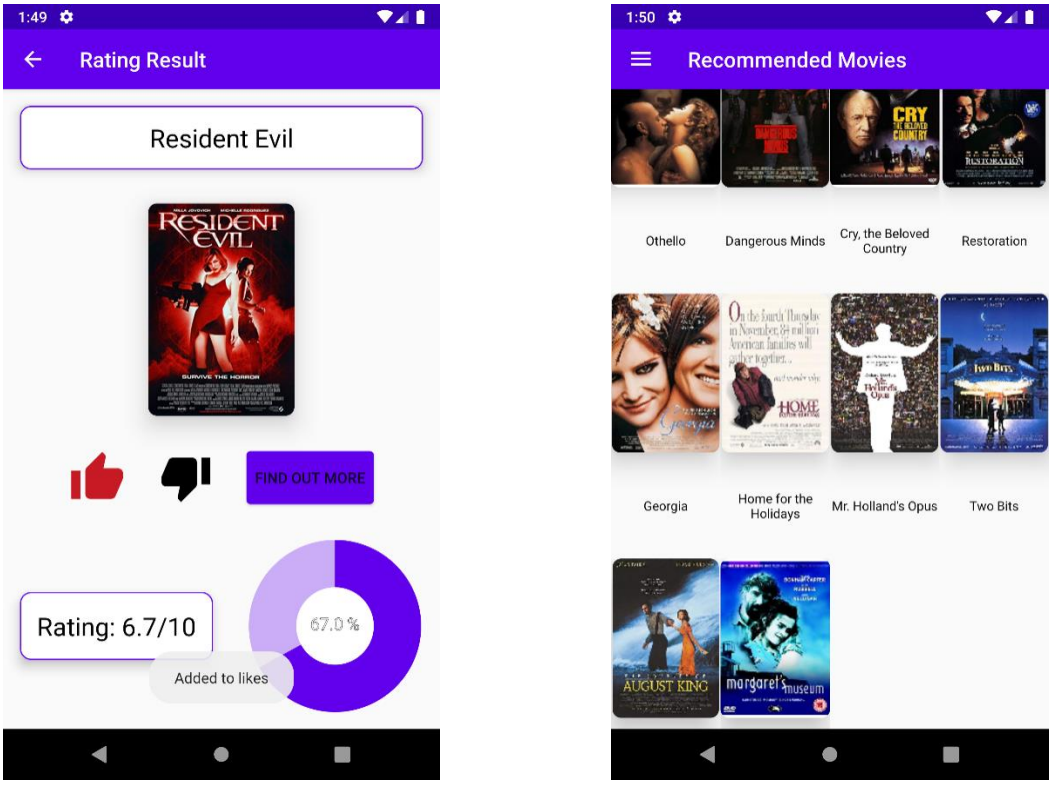
Test Case ID	Input Data/State description	Expected Result	Actual Result	Status	Figure ID
TRM001	New user without any liked movie in database	User not in database	User not in database	Pass	Fig(TRM1)
TRM002	User in database with liked movies	Recommended movies	Recommended Movies	Pass	Fig(TRM2)
TRM003	Like another movie and see the recommendation page	Changed recommended movies	Changed recommended movies	Pass	Fig(TRM3)



Fig(TRM1)



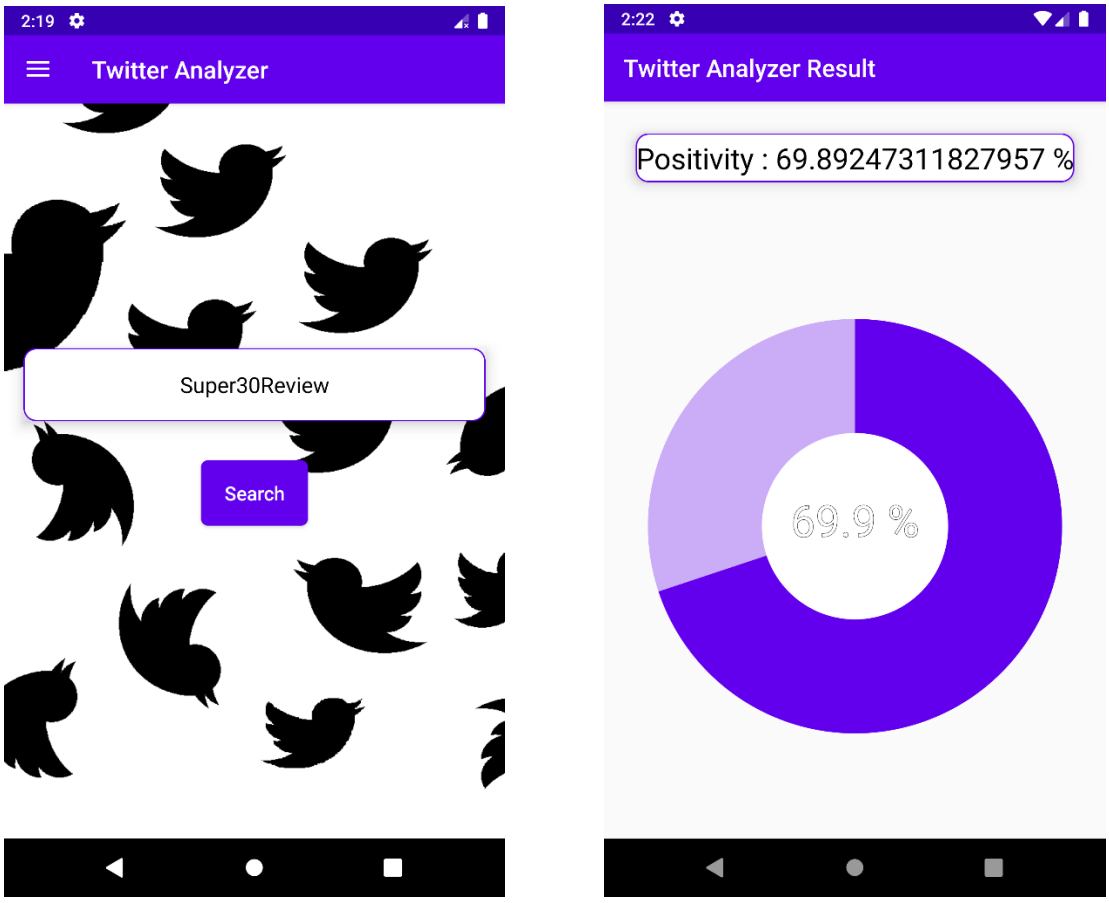
Fig(TRM2)


1:49
Rating Result
Resident Evil
RESIDENT EVIL
SURVIVE THE HORROR
FIND OUT MORE
Rating: 6.7/10
67.0 %
Added to likes
1:50
Recommended Movies
Othello
Dangerous Minds
Cry, the Beloved Country
Restoration
Georgia
Home for the Holidays
Mr. Holland's Opus
Two Bits
AUGUST KING
margaret's museum

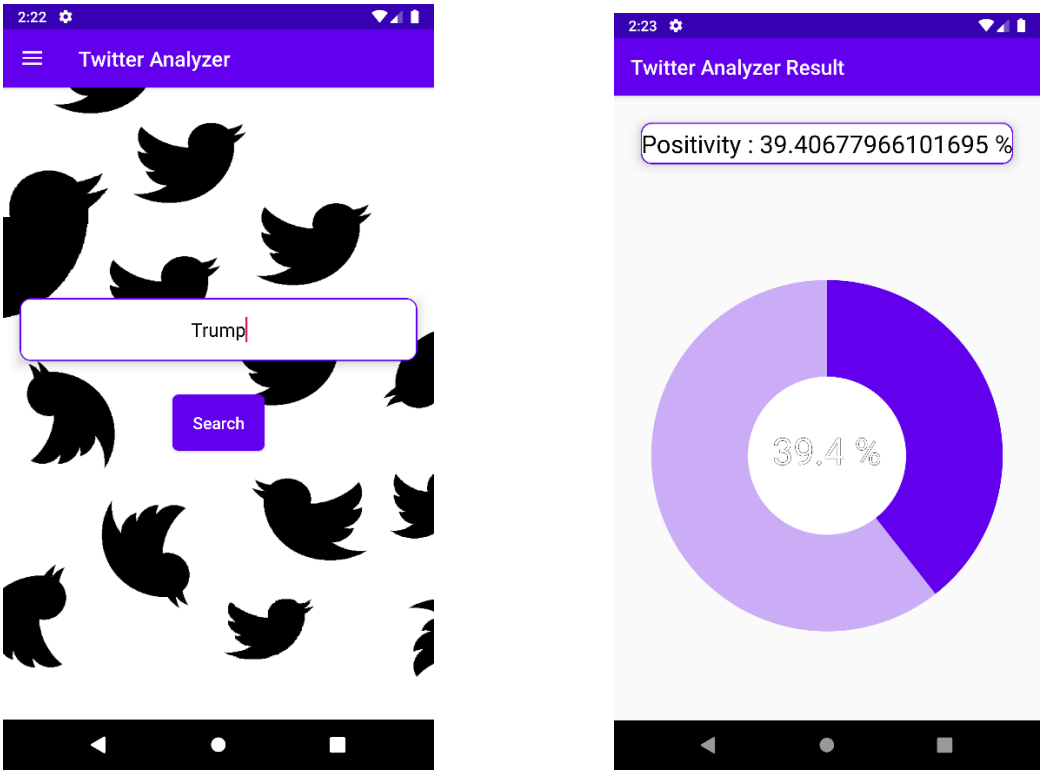
Fig(TRM3)

10.6 Twitter Analyzer Testing

Test Case ID	Input Data/State description	Expected Result	Actual Result	Status	Figure ID
TAN001	Search string with expected positive result	Positive score	Positive score	Pass	Fig(TAN1)
TAN002	Search string with expected negative result	Negative score	Negative score	Pass	Fig(TAN2)



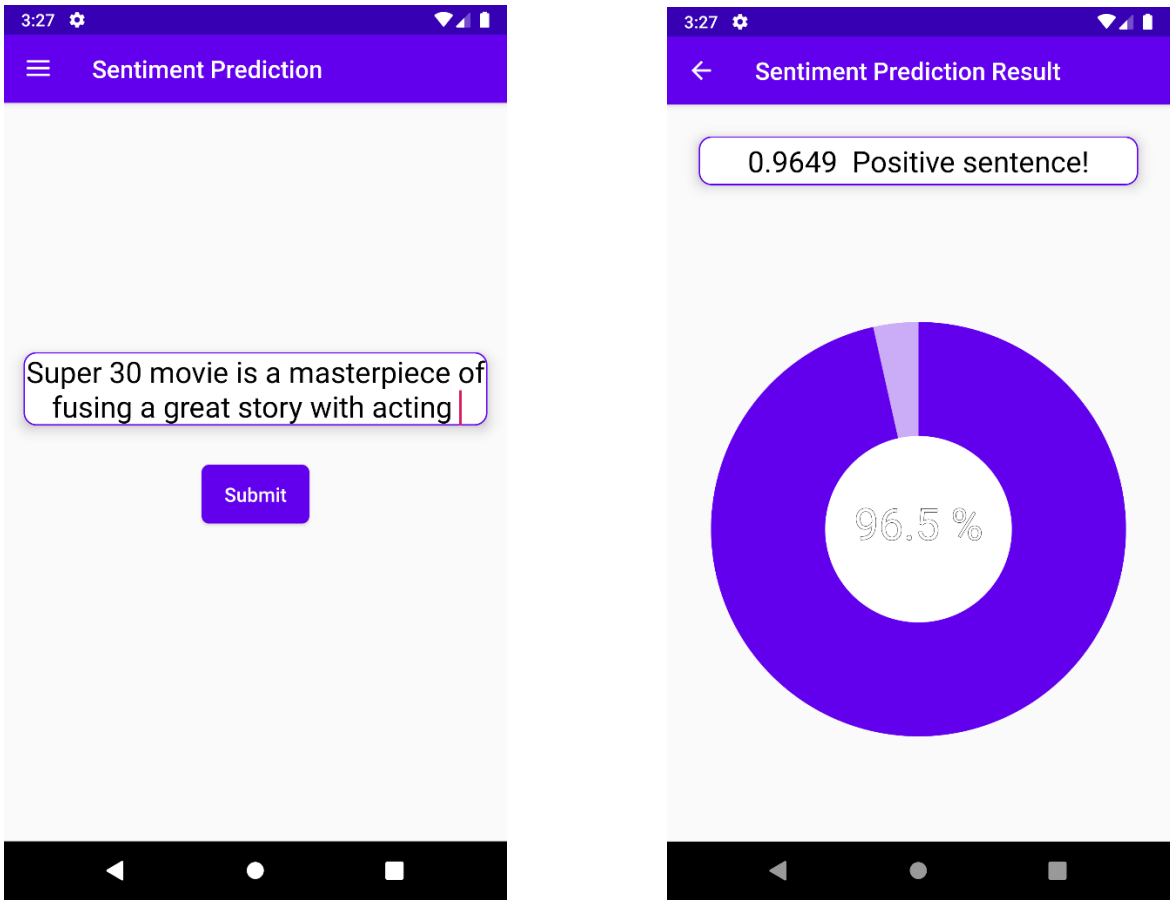
Fig(TAN1)


2:22
Twitter Analyzer
Trump
Search
2:23
Twitter Analyzer Result
Positivity : 39.40677966101695 %
39.4 %

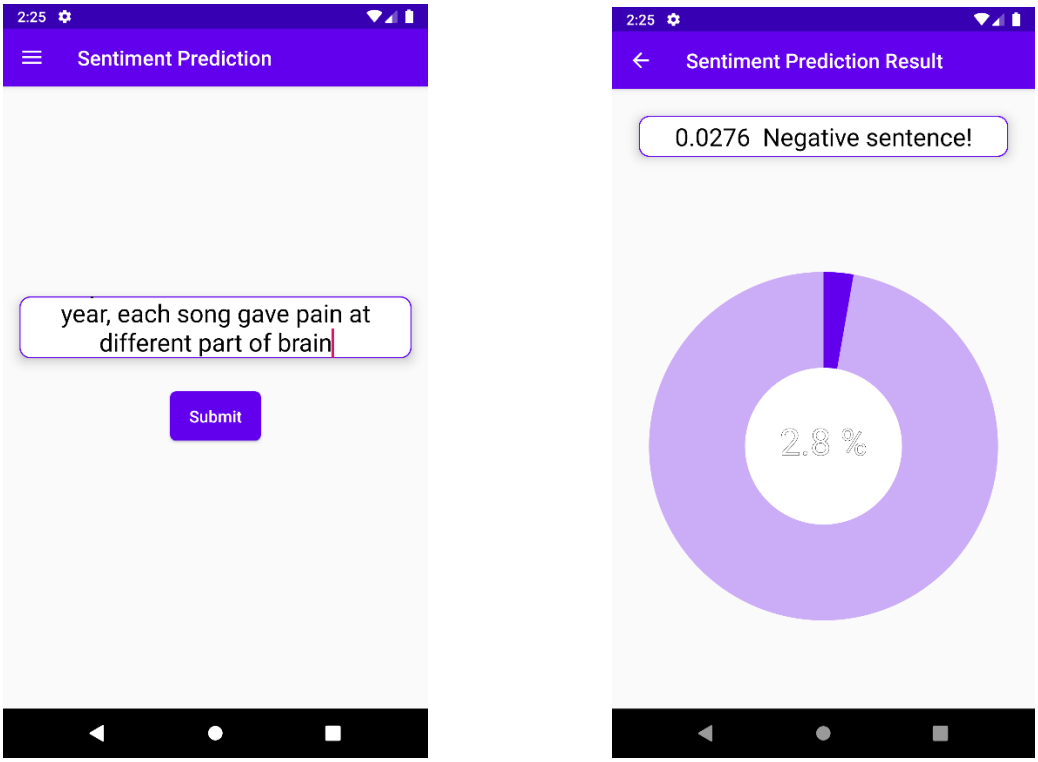
Fig(TAN2)

10.7 Sentiment Analysis Testing

Test Case ID	Input Data/State description	Expected Result	Actual Result	Status	Figure ID
TSA001	Pass positive text	Positive score	Positive score	Pass	Fig(TSA1)
TSA002	Pass negative score	Negative score	Negative score	Pass	Fig(TSA2)



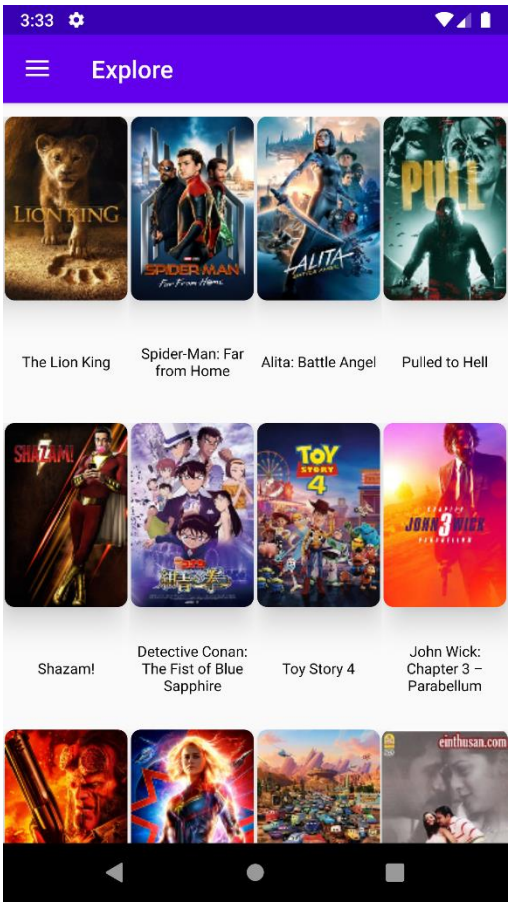
Fig(TSA1)


2:25
Sentiment Prediction
year, each song gave pain at different part of brain
Submit
2:25
Sentiment Prediction Result
0.0276 Negative sentence!
2.8 %

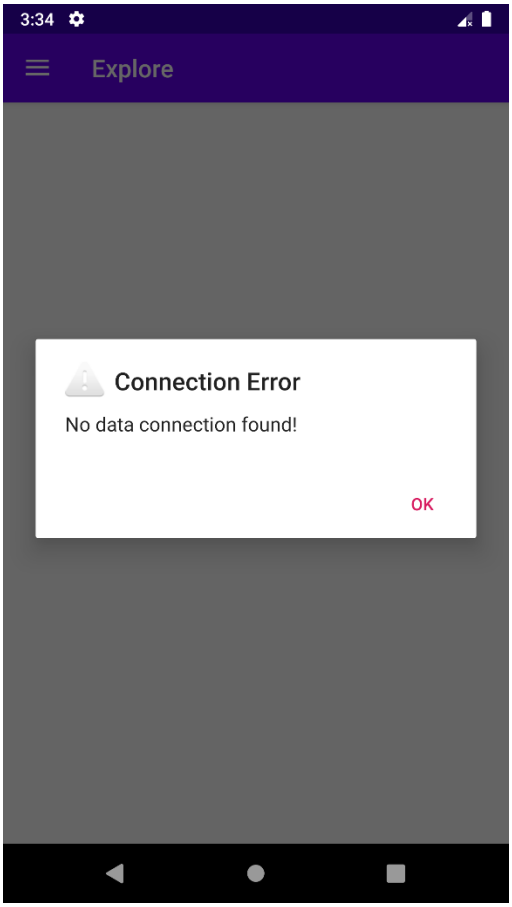
Fig(TSA2)

10.8 Trending Page Testing

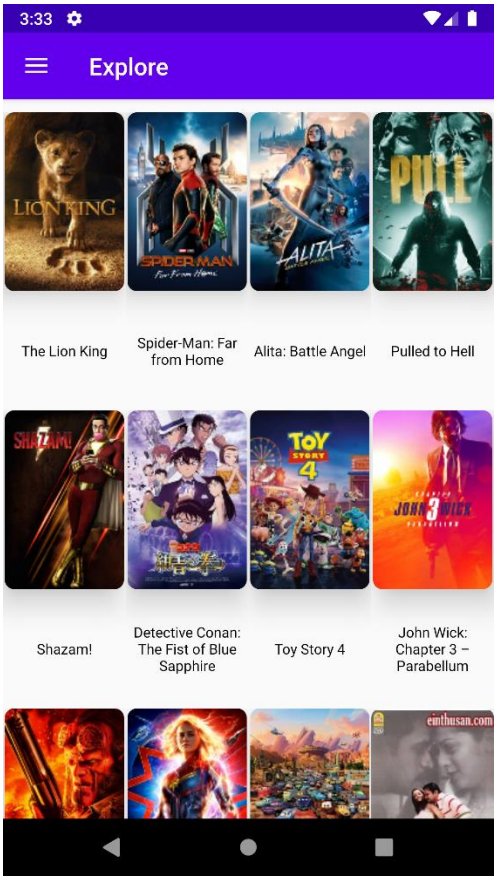
Test Case ID	Input Data/State description	Expected Result	Actual Result	Status	Figure ID
TTP001	Open explore page with data connection on	Show trending Movies	Show trending movies	Pass	Fig(TTP1)
TTP002	Open explore page with data connection off	Show connection error	Show connection error	Pass	Fig(TTP2)
TTP003	From explore page click on a movie	Show more details about movie	Show more details about movie	Pass	Fig(TTP3)

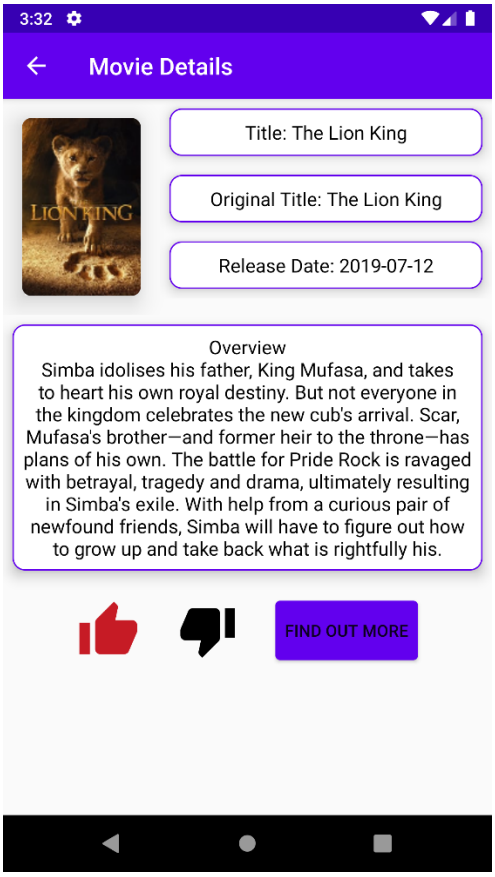


Fig(TTP1)



Fig(TTP2)

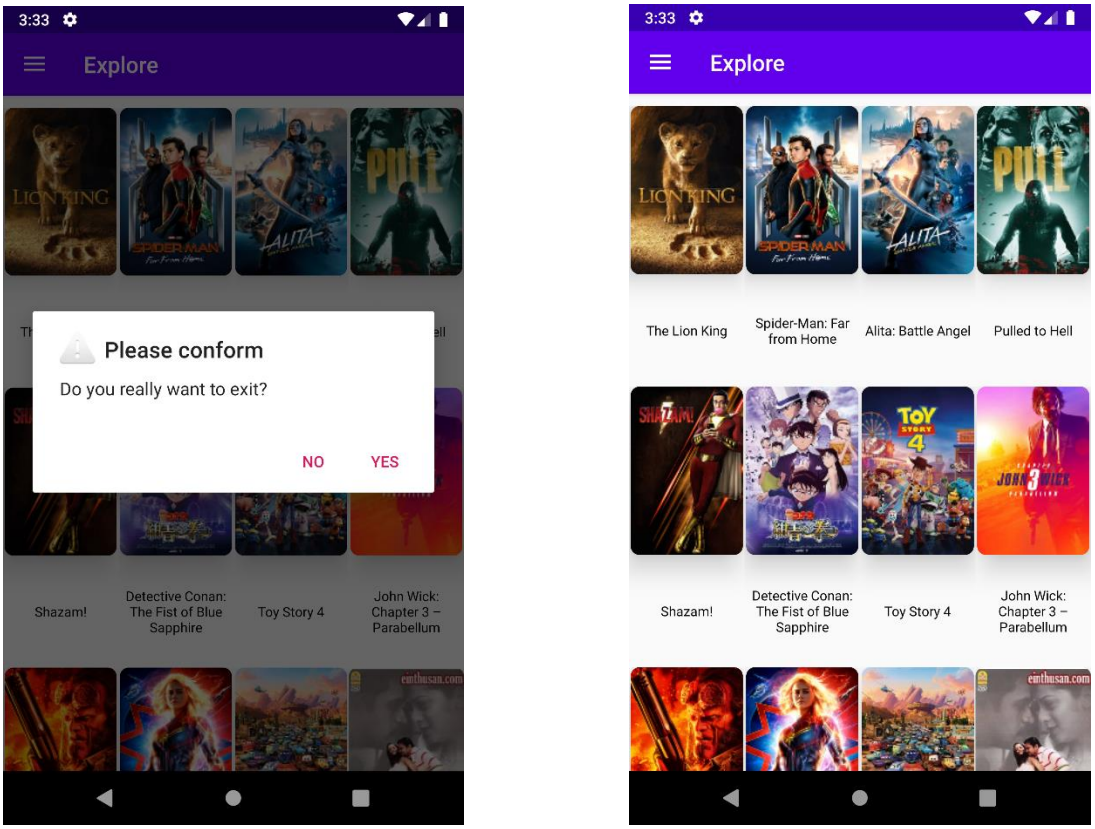

3:33
Explore
The Lion King
Spider-Man: Far from Home
Alita: Battle Angel
Pulled to Hell
Shazam!
Detective Conan: The Fist of Blue Sapphire
Toy Story 4
John Wick: Chapter 3 – Parabellum


3:32
Movie Details
Title: The Lion King
Original Title: The Lion King
Release Date: 2019-07-12
Overview
Simba idolises his father, King Mufasa, and takes to heart his own royal destiny. But not everyone in the kingdom celebrates the new cub's arrival. Scar, Mufasa's brother—and former heir to the throne—has plans of his own. The battle for Pride Rock is ravaged with betrayal, tragedy and drama, ultimately resulting in Simba's exile. With help from a curious pair of newfound friends, Simba will have to figure out how to grow up and take back what is rightfully his.
FIND OUT MORE

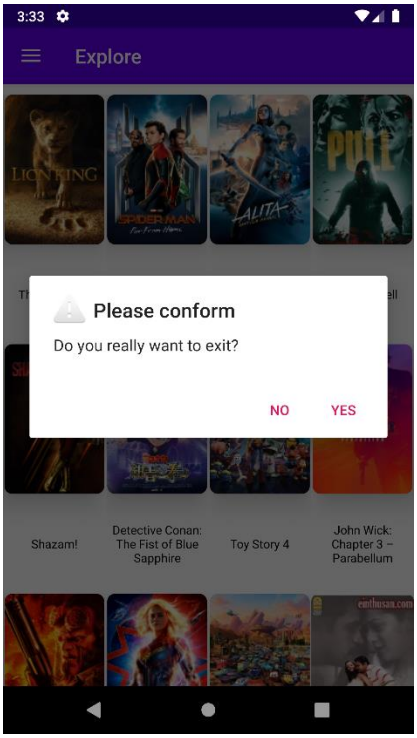
Fig(TTP3)

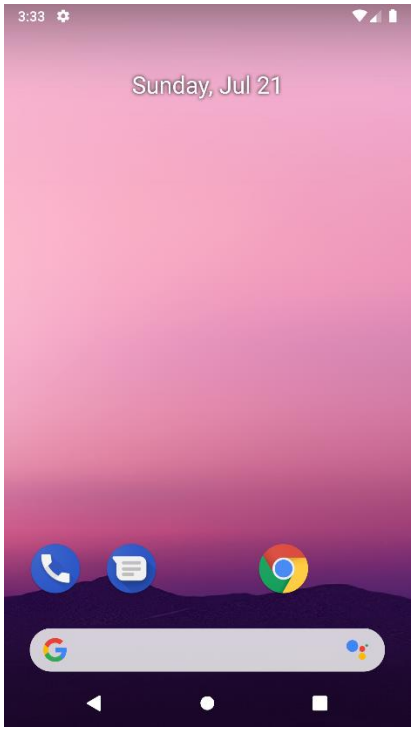
10.9 Logout and Exit Commands Testing

Test Case ID	Input Data/State description	Expected Result	Actual Result	Status	Figure ID
TLE001	Select no for exit	Stay on same fragment	Stayed on same fragment	Pass	Fig(TLE1)
TLE002	Select yes for exit	Go to home screen	Go to home screen	Pass	Fig(TLE2)
TLE003	Select no for logout	Stay on the same fragment	Stayed on the same fragment	Pass	Fig(TLE3)
TLE004	Select yes for logout	Direct to registration screen	Directed to registration screen	Pass	Fig(TLE4)

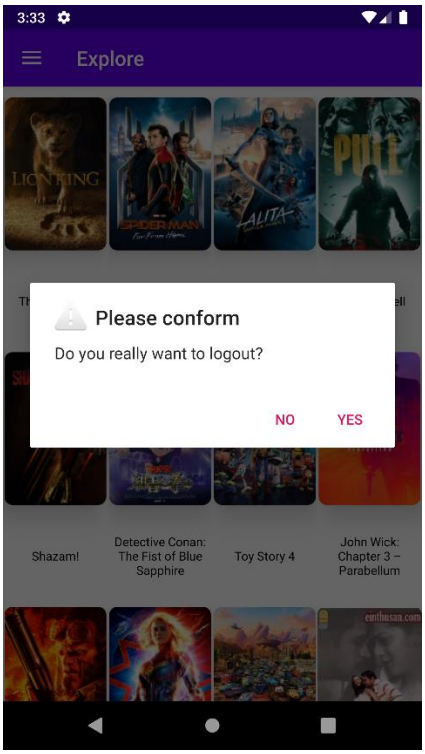


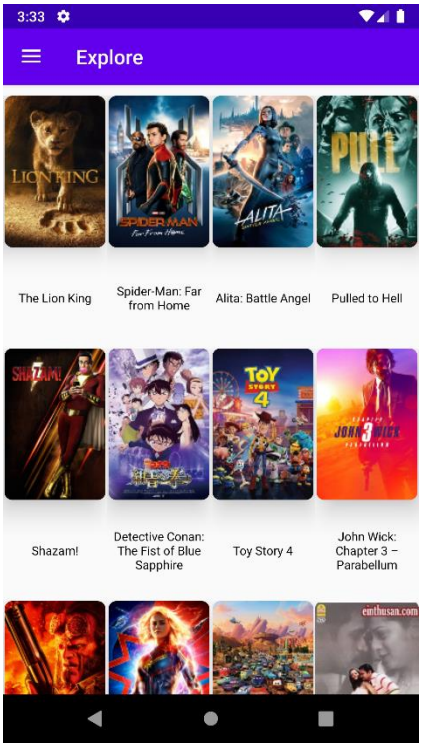
Fig(TLE1)



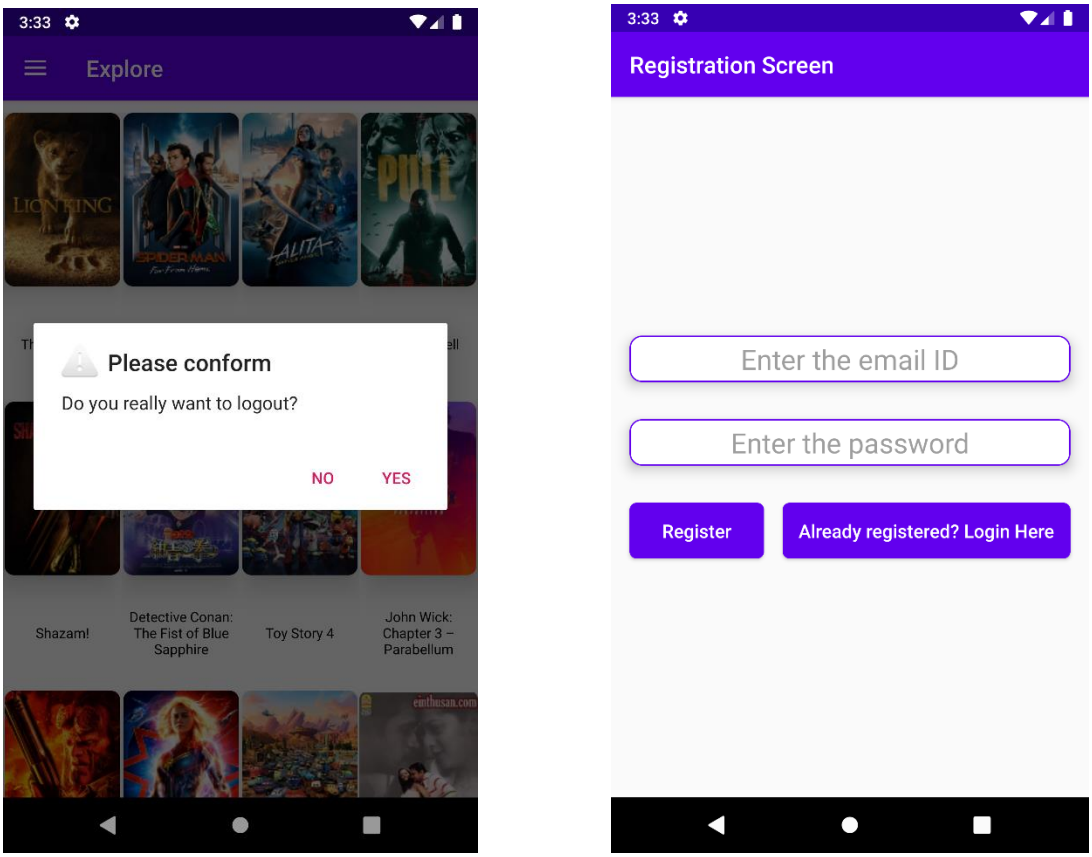


Fig(TLE2)





Fig(TLE3)


3:33
Explore
Please conform
Do you really want to logout?
NO
YES
Shazam!
Detective Conan: The Fist of Blue Sapphire
Toy Story 4
John Wick: Chapter 3 – Parabellum
3:33
Registration Screen
Enter the email ID
Enter the password
Register
Already registered? Login Here

Fig(TLE4)

10.10 Remaining View Layouts with description

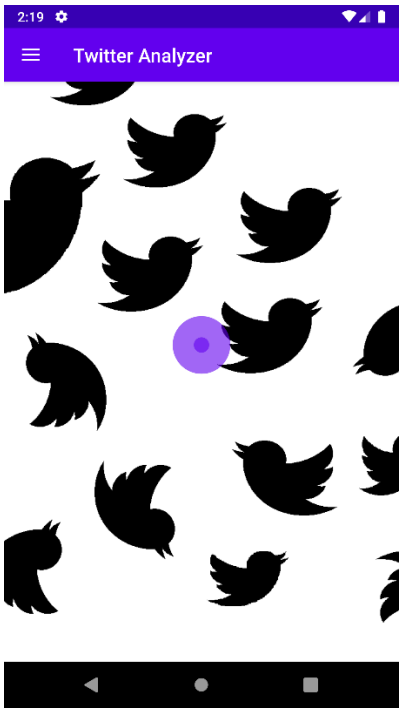


Fig 1: Loading animation for Twitter Analyzer fragment

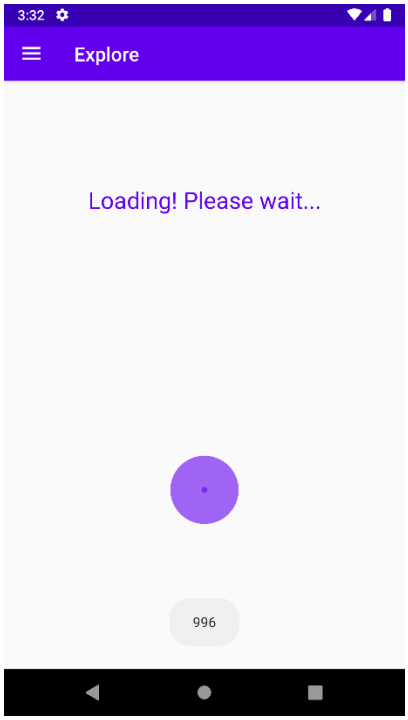


Fig 2: Loading animation for explore

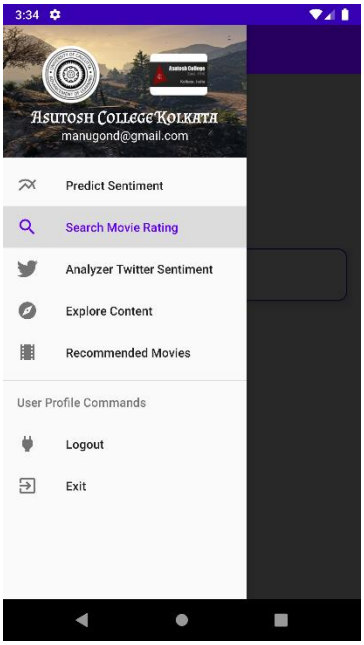


Fig 3: Navigation Bar

11. Future Scope

11.1 Conclusion

Nowadays, sentiment analysis or opinion mining is a hot topic in machine learning. We are still far to detect the sentiments of s corpus of texts very accurately because of the complexity in the English language and even more if we consider other languages such as Chinese. Also the recommendation system using content based filtering has its own drawbacks which can be avoided with hybrid recommendation system.

In this project we tried to show the basic way of classifying text into positive or negative category using Deep Learning approach using tools which includes **PyTorch, Google Colab for training on GPU**. We could further improve our classifier by trying to extract more features from the text, trying different kinds of features, tuning the parameters of the created model, and training with more data. The recommendation system used content based filtering as its core. Since the whole system has been implementd with modular approach, it is easy to improve the one section of system such as sentiment analysis model without affecting the whole system. Overall recommendation system can be improved with a **hybrid system** approach using Deep Learning or even further with use of **private user data** and **Encrypted Deep Learning**.

11.2 Future Scopes

Sentiment analysis, as an interdisciplinary field that crosses natural language processing, artificial intelligence, and text mining. We have seen that Sentiment Analysis can be used for analyzing opinions in blogs, newspaper, articles, Product reviews, Social Media websites, Movie-review websites where a third person narrates his/her views. Recommendation system implemented here using setntiment analysis is working with content based data. When user base grows the following can be done:

1. Migrate to a hybrid recommendation system for much better results.

2. Add caching of results on server to avoid computation overhead.

3. Categorise the movies based on region and language preference.

4. Add localization to the android app, so it can be used in most phones supporting difference languages for rich user experice.

5. Migrate to GPUs for much faster computation.

6. Connect API methods to book tickets directly from app.

7. Add support for Firebase Database to store thumbnails locally, hence reducing network requests.

The project implemented has future research scopes that might include the performance evaluation of the implemented system with the currently used systems in well knows websites. The hybrid recommendation system can be made more accurate with the use of private data from companies that can be used to train the model with Encrypted Deep Learning approach, so that the information about user doesnot is leaked and on the other hand, more accurate system can be created.

12. References

1. Kingma, D.P. and Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
2. Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T. and Qin, B., 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (Vol. 1, pp. 1555-1565).
3. Williams, R. J.; Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity, D. (1 February 2013). Chauvin, Yves; Rumelhart, David E., eds. Backpropagation: Theory, Architectures, and Applications. Psychology Press. ISBN 978-1-134-77581-1.
4. Willmott, C.J. and Matsuura, K., 2005. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate research*, *30*(1), pp.79-82.
5. Palangi, H., Deng, L., Shen, Y., Gao, J., He, X., Chen, J., Song, X. and Ward, R., 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, *24*(4), pp.694-707.
6. Wang, Y., Huang, M. and Zhao, L., 2016. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing* (pp. 606-615).
7. Kumar, S., Halder, S.S., De, K. and Roy, P.P., 2018. Movie Recommendation System using Sentiment Analysis from Microblogging Data. *arXiv preprint arXiv:1811.10804*.

13. Appendix

13.1 Appendix A: Sentiment Analysis Model

Data Processing Python Code

myModelHelper.py

```
import numpy as np
import os
from string import punctuation
from collections import Counter
import torch
from torch.utils.data import TensorDataset,DataLoader
import torch.nn as nn

defcreateData(positiveDataPath,negativeDataPath):
    reviews=[]
    labels=[]
    all_files_neg = os.listdir(negativeDataPath)
    all_files_pos = os.listdir(positiveDataPath)
for file_location in all_files_pos:
withopen(positiveDataPath+file_location,'r',encoding="utf8")as f:
            reviews.append(f.read())
            labels.append("positive")

for file_location in all_files_neg:
withopen(negativeDataPath+file_location,'r',encoding="utf8")as f:
            reviews.append(f.read())
            labels.append("negative")
    myDict={}
for i inrange(len(reviews)):
       myDict[i]=i
    reviewArray=np.arange(len(reviews))
    np.random.shuffle(reviewArray)
    reviewString=""
    labelString=""
for i inlist(reviewArray):
        reviewString=reviewString+reviews[i]+"\n"
        labelString=labelString+labels[myDict[i]]+"\n"

return reviewString,labelString


deffeaturesPadding(reviews,sequenceLength):
  features=np.zeros((len(reviews),sequenceLength),dtype=int)
```

```
for i,row inenumerate(reviews):
    features[i,-len(row):]=np.array(row)[:sequenceLength]
return features


defgetFeaturesLabels(reviews,labels,seq_length):
  reviews=reviews.lower()
  all_text=''.join([c for c in reviews if c notin punctuation])
  reviews_split=all_text.split('\n')
  all_text=' '.join(reviews_split)
  words=all_text.split()
  counts=Counter(words)
  vocab=sorted(counts,key=counts.get,reverse=True)
  vocab_to_int={word:ii for ii, word inenumerate(vocab,1)}
  reviews_ints=[]
for review in reviews_split:
    reviews_ints.append([vocab_to_int[word] for word in review.split()])
  labels_split=labels.split('\n')
  encoded_labels=np.array([1if label=='positive'else0for label in
labels_split])
  reviews_lens=Counter([len(x) for x in reviews_ints])
  non_zero_idx=[ii for ii,review inenumerate(reviews_ints) iflen(review)!=0]
  reviews_ints=[reviews_ints[ii] for ii in non_zero_idx]
  encoded_labels=np.array([encoded_labels[ii] for ii in non_zero_idx])
  features=featuresPadding(reviews_ints,seq_length)
return features,encoded_labels,vocab_to_int


defcreateTrainTestValidateData(split_frac,features,encoded_labels):
  split_idx = int(len(features)*split_frac)
  train_x, remaining_x = features[:split_idx], features[split_idx:]
  train_y, remaining_y = encoded_labels[:split_idx],
encoded_labels[split_idx:]
  test_idx = int(len(remaining_x)*0.5)
  val_x, test_x = remaining_x[:test_idx], remaining_x[test_idx:]
  val_y, test_y = remaining_y[:test_idx], remaining_y[test_idx:]
return train_x,train_y,test_x,test_y,val_x,val_y
```

LSTM Cell implementation

```
import random

import numpy as np
import math

defsigmoid(x):
return1. / (1 + np.exp(-x))

defsigmoid_derivative(values):
return values*(1-values)

deftanh_derivative(values):
return1. - values ** 2

defrand_arr(a, b, *args):
    np.random.seed(0)
return np.random.rand(*args) * (b - a) + a

classLstmParam:
def__init__(self, mem_cell_ct, x_dim):
self.mem_cell_ct = mem_cell_ct
self.x_dim = x_dim
        concat_len = x_dim + mem_cell_ct

# all weight matrices
self.wg = rand_arr(-0.1, 0.1, mem_cell_ct, concat_len)
self.wi = rand_arr(-0.1, 0.1, mem_cell_ct, concat_len)
self.wf = rand_arr(-0.1, 0.1, mem_cell_ct, concat_len)
self.wo = rand_arr(-0.1, 0.1, mem_cell_ct, concat_len)

self.bg = rand_arr(-0.1, 0.1, mem_cell_ct)
self.bi = rand_arr(-0.1, 0.1, mem_cell_ct)
self.bf = rand_arr(-0.1, 0.1, mem_cell_ct)
self.bo = rand_arr(-0.1, 0.1, mem_cell_ct)

self.wg_diff = np.zeros((mem_cell_ct, concat_len))
self.wi_diff = np.zeros((mem_cell_ct, concat_len))
self.wf_diff = np.zeros((mem_cell_ct, concat_len))
self.wo_diff = np.zeros((mem_cell_ct, concat_len))
self.bg_diff = np.zeros(mem_cell_ct)
self.bi_diff = np.zeros(mem_cell_ct)
self.bf_diff = np.zeros(mem_cell_ct)
self.bo_diff = np.zeros(mem_cell_ct)

defapply_diff(self, lr = 1):
```

```
self.wg -= lr * self.wg_diff
self.wi -= lr * self.wi_diff
self.wf -= lr * self.wf_diff
self.wo -= lr * self.wo_diff
self.bg -= lr * self.bg_diff
self.bi -= lr * self.bi_diff
self.bf -= lr * self.bf_diff
self.bo -= lr * self.bo_diff

self.wg_diff = np.zeros_like(self.wg)
self.wi_diff = np.zeros_like(self.wi)
self.wf_diff = np.zeros_like(self.wf)
self.wo_diff = np.zeros_like(self.wo)
self.bg_diff = np.zeros_like(self.bg)
self.bi_diff = np.zeros_like(self.bi)
self.bf_diff = np.zeros_like(self.bf)
self.bo_diff = np.zeros_like(self.bo)

classLstmState:
def__init__(self, mem_cell_ct, x_dim):
self.g = np.zeros(mem_cell_ct)
self.i = np.zeros(mem_cell_ct)
self.f = np.zeros(mem_cell_ct)
self.o = np.zeros(mem_cell_ct)
self.s = np.zeros(mem_cell_ct)
self.h = np.zeros(mem_cell_ct)
self.bottom_diff_h = np.zeros_like(self.h)
self.bottom_diff_s = np.zeros_like(self.s)

classLstmNode:
def__init__(self, lstm_param, lstm_state):

self.state = lstm_state
self.param = lstm_param

self.xc = None

defbottom_data_is(self, x, s_prev = None, h_prev = None):

if s_prev isNone: s_prev = np.zeros_like(self.state.s)
if h_prev isNone: h_prev = np.zeros_like(self.state.h)

self.s_prev = s_prev
self.h_prev = h_prev


        xc = np.hstack((x,  h_prev))
self.state.g = np.tanh(np.dot(self.param.wg, xc) + self.param.bg)
```

```
self.state.i = sigmoid(np.dot(self.param.wi, xc) + self.param.bi)
self.state.f = sigmoid(np.dot(self.param.wf, xc) + self.param.bf)
self.state.o = sigmoid(np.dot(self.param.wo, xc) + self.param.bo)
self.state.s = self.state.g * self.state.i + s_prev * self.state.f
self.state.h = self.state.s * self.state.o

self.xc = xc

deftop_diff_is(self, top_diff_h, top_diff_s):

        ds = self.state.o * top_diff_h + top_diff_s
        do = self.state.s * top_diff_h
        di = self.state.g * ds
        dg = self.state.i * ds
        df = self.s_prev * ds


        di_input = sigmoid_derivative(self.state.i) * di
        df_input = sigmoid_derivative(self.state.f) * df
        do_input = sigmoid_derivative(self.state.o) * do
        dg_input = tanh_derivative(self.state.g) * dg


self.param.wi_diff += np.outer(di_input, self.xc)
self.param.wf_diff += np.outer(df_input, self.xc)
self.param.wo_diff += np.outer(do_input, self.xc)
self.param.wg_diff += np.outer(dg_input, self.xc)
self.param.bi_diff += di_input
self.param.bf_diff += df_input
self.param.bo_diff += do_input
self.param.bg_diff += dg_input


        dxc = np.zeros_like(self.xc)
        dxc += np.dot(self.param.wi.T, di_input)
        dxc += np.dot(self.param.wf.T, df_input)
        dxc += np.dot(self.param.wo.T, do_input)
        dxc += np.dot(self.param.wg.T, dg_input)


self.state.bottom_diff_s = ds * self.state.f
self.state.bottom_diff_h = dxc[self.param.x_dim:]

classLstmNetwork():
def__init__(self, lstm_param):
self.lstm_param = lstm_param
self.lstm_node_list = []
self.x_list = []
```

```
defy_list_is(self, y_list, loss_layer):

assertlen(y_list) == len(self.x_list)
        idx = len(self.x_list) - 1
        loss = loss_layer.loss(self.lstm_node_list[idx].state.h, y_list[idx])
        diff_h = loss_layer.bottom_diff(self.lstm_node_list[idx].state.h,
y_list[idx])
        diff_s = np.zeros(self.lstm_param.mem_cell_ct)
self.lstm_node_list[idx].top_diff_is(diff_h, diff_s)
        idx -= 1
while idx >= 0:
            loss += loss_layer.loss(self.lstm_node_list[idx].state.h,
y_list[idx])
            diff_h = loss_layer.bottom_diff(self.lstm_node_list[idx].state.h,
y_list[idx])
            diff_h += self.lstm_node_list[idx + 1].state.bottom_diff_h
            diff_s = self.lstm_node_list[idx + 1].state.bottom_diff_s
self.lstm_node_list[idx].top_diff_is(diff_h, diff_s)
            idx -= 1

return loss

defx_list_clear(self):
self.x_list = []

defx_list_add(self, x):
self.x_list.append(x)
iflen(self.x_list) >len(self.lstm_node_list):
            lstm_state = LstmState(self.lstm_param.mem_cell_ct,
self.lstm_param.x_dim)
self.lstm_node_list.append(LstmNode(self.lstm_param, lstm_state))
        idx = len(self.x_list) - 1
if idx == 0:
self.lstm_node_list[idx].bottom_data_is(x)
else:
            s_prev = self.lstm_node_list[idx - 1].state.s
            h_prev = self.lstm_node_list[idx - 1].state.h
self.lstm_node_list[idx].bottom_data_is(x, s_prev, h_prev)
```

Google Colab Notebook Code to run model on GPU

```
from google.colab import files,drive
drive.mount('/content/drive/')

# Get the generated helper functions to process the data and create proper
datasets to be converted into tensors
!wget -c https://raw.githubusercontent.com/3ZadeSSG/Minor-Project-Sentiment-
Analysis/master/myModelHelper.py

# Get data directly from url
!wget http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz

# Extract data
!tar -xvzf aclImdb_v1.tar.gz

import torch
import torch.nn as nn
from torch.utils.data import TensorDataset,DataLoader
from myModelHelper import *
import matplotlib.pyplot as plt

classSentimentNetwork(nn.Module):
def__init__(self,vocabulary_size,output_size,embedding_dimension,hidden_dimens
ion,number_of_layers,dropout_probability=0.5):
super(SentimentNetwork,self).__init__()
# class data for weight matrix size
self.output_size=output_size
self.number_of_layers=number_of_layers
self.hidden_dimension=hidden_dimension
# create the embedding layer to reduce dimension
# and LSTM layer containing LSTM cells
self.embedding=nn.Embedding(vocabulary_size,embedding_dimension)
self.lstm=nn.LSTM(embedding_dimension,hidden_dimension,number_of_layers,dropou
t=dropout_probability,batch_first=True)

# Create dropout layer as a regularization method to reduct overfitting
# this will disable some units in forward pass, thus preventing
# a particular set of node's weights getting updated while others remain
unused
self.dropout=nn.Dropout(dropout_probability)

# attach final  linear layer with sigmoid function
self.finalLayer=nn.Linear(hidden_dimension,output_size)
self.sigmoid=nn.Sigmoid()

defforward(self,x,hidden):
```

```
    batch_size=x.size(0)
    embedding_output=self.embedding(x)

#lstm will take the current input and hidden state as input
# and will generate the output to be feed at the linear layer
    lstm_output,hidden=self.lstm(embedding_output,hidden)
    lstm_output=lstm_output.contiguous().view(-1,self.hidden_dimension)

    output=self.dropout(lstm_output)
    output=self.finalLayer(output)

#call the sigmoid function on current output of final layer
    sigmoid_output=self.sigmoid(output)
    sigmoid_output=sigmoid_output.view(batch_size,-1)
#only get the last position output for all batches
    sigmoid_output=sigmoid_output[:,-1]

#current sigmoid output and a hidden state to be fed as input for next pass
# into the LSTM cells, so that they will be dependent on the previous state
return sigmoid_output,hidden

definitialize_hidden_state(self,batch_size):
# At first the hidden state will not hold any information, hence we need to
# initialize them with zeros
# Number_of_Layers x Batch_Size x Hidden_Dimension
    weight=next(self.parameters()).data
if(torch.cuda.is_available()):
# if GPU is available then initialize the weights parallely

hidden=(weight.new(self.number_of_layers,batch_size,self.hidden_dimension).zer
o_().cuda(),weight.new(self.number_of_layers,batch_size,self.hidden_dimension)
.zero_().cuda())
else:

hidden=(weight.new(self.number_of_layers,batch_size,self.hidden_dimension).zer
o_(),weight.new(self.number_of_layers,batch_size,self.hidden_dimension).zero_(
))

return hidden



from torch.optim import Optimizer
import  math
classAdamOptimizerAlgorithm(Optimizer):


def__init__(self, params, lr=1e-3, betas=(0.9, 0.999), eps=1e-8):
        defaults = dict(lr=lr, betas=betas, eps=eps)
```

```
super(AdamOptimizerAlgorithm, self).__init__(params, defaults)

defstep(self):
for group inself.param_groups:
for p in group['params']:
if p.grad isNone:
continue
                grad = p.grad.data
                state = self.state[p]

# State initialization in case of initial state
iflen(state) == 0:
                    state['step'] = 0
# Exponential moving average of gradient values
                    state['exp_avg'] = torch.zeros_like(p.data)
# Exponential moving average of squared gradient values
                    state['exp_avg_sq'] = torch.zeros_like(p.data)


                exp_avg, exp_avg_sq = state['exp_avg'], state['exp_avg_sq']

                beta1, beta2 = group['betas']

                state['step'] += 1

#following is just a formula implementaion of the algorithm
                exp_avg.mul_(beta1).add_(1 - beta1, grad)
                exp_avg_sq.mul_(beta2).addcmul_(1 - beta2, grad, grad)

                denom = exp_avg_sq.sqrt().add_(group['eps'])

                b_1 = 1 - beta1 ** state['step']
                b_2 = 1 - beta2 ** state['step']
                step_size = group['lr'] * math.sqrt(b_2) / b_1

                p.data.addcdiv_(-step_size, exp_avg, denom)  # -
step_size*(exp_avg/denom) operation done in single step



defbuildNetwork():
#Our model parameters
  vocabulary_size=len(vocab_to_int)+1
  output_size=1
  embedding_dimension=600
  hidden_dimension=512
  number_of_layers=3
```

```
model=SentimentNetwork(vocabulary_size,output_size,embedding_dimension,hidden_
dimension,number_of_layers)

# Alpha value
  learning_rate=0.003

# Error calculating formula (Mean Square Error)
  criterion = nn.MSELoss()

#Adam optimization technique for first-order gradient-based optimization
  optimizer=AdamOptimizerAlgorithm(model.parameters(),lr=learning_rate)


return model,criterion,optimizer



deftrainNetwork(model,epochs,print_interval,criterion,optimizer):
  Iteration=[]
  Training_Loss=[]
  Validation_Loss=[]

  minimum_validation_loss=np.Inf

# Since each epoch goes over all data and
# each iteration goes over the data in batches
  count=0

# move model to GPU
  model.cuda()

#put the model into training model so that the gradients will
# be calculated
  model.train()
for i inrange(epochs):
# for each epoch perform a forward pass for all batches
# and around epoch 4 reduce the learning rate
    hidden_state=model.initialize_hidden_state(batch_size)

if(i==3):
print("\n\nReducing the learning rate!\n" )
#optimizer=torch.optim.Adam(model.parameters(),lr=0.0002)
      optimizer=AdamOptimizerAlgorithm(model.parameters(),lr=0.0002)

for inputs,labels in train_loader:
      count+=1
# move the data to GPU
      inputs,labels=inputs.cuda(),labels.cuda()
```

```
#create new variable for hidden state otherwise it will include all
#pervious states
      hidden_state=tuple([element.data for element in hidden_state])

      model.zero_grad()

      inputs=inputs.long()
      output,hidden_state=model(inputs,hidden_state)

#calculate the loss to backpropagare to model
#loss=torch.sqrt(criterion(output.squeeze(),labels.float()))
      loss=criterion(output.squeeze(),labels.float())
      loss.backward()


#nn.utils.clip_grad_norm_(model.parameters(),clip)
      optimizer.step()

if(count%print_interval==0):
        temp_hidden=model.initialize_hidden_state(batch_size)
        temp_losses=[]
        model.eval()
for inputs,labels in validation_loader:
          temp_hidden=tuple([element.data for element in temp_hidden])
          inputs,labels=inputs.cuda().long(),labels.cuda()

          output,temp_hidden=model(inputs,temp_hidden)
          validation_loss=criterion(output.squeeze(),labels.float())
          temp_losses.append(validation_loss.item())

        model.train()
print("Epoch: {}\tIteration: {}\tTraining Loss: {:.7f}\tValidation Loss:
{:.7f}".format(
        (i+1),count,loss.item(),np.mean(temp_losses)))
        Iteration.append(count)
        Training_Loss.append(loss.item())
        Validation_Loss.append(np.mean(temp_losses))

if np.mean(temp_losses)<minimum_validation_loss:
            save_checkpoint(model,"checkpoint.pth")
            minimum_validation_loss=np.mean(temp_losses)
print("Validation loss decreased  hence saving checkpoint successfully")



  plt.plot(Iteration,Training_Loss)
  plt.plot(Iteration,Validation_Loss)
```

```
  plt.xlabel('Iteration')
  plt.ylabel('Loss')
  plt.ylim(0.0,1.0)
  plt.legend(['Training Loss','Validation Loss'], loc='upper left')
  plt.show()



defsave_checkpoint(model,fileLocation):
# save the current state_dict which contains all the weights of the network
  torch.save(model.state_dict(),fileLocation)



defload_Checkpoint(fileLocation):
#model=buildNetwork()
  model,criterion,optimizer=buildNetwork()
  model.load_state_dict(torch.load(fileLocation))
return model


defcheckAccuracy(model,test_loader):
#move model to GPU
  model.cuda()

  test_losses=[]
  correct_prediction=0

#set initial state to zero
  hidden_state=model.initialize_hidden_state(batch_size)

# set the model into evaluation mode, so that we don't need to calculate
# the gradients
  model.eval()

for inputs,labels in test_loader:

# move the data and labels into GPU
    inputs,labels=inputs.cuda().long(),labels.cuda().long()

    output,hidden_state=model(inputs,hidden_state)
# labels are initially long or int, we need to conver them into float
# so that loss cab be calculated in float value
    test_loss=criterion(output.squeeze(),labels.float())
    test_losses.append(test_loss.item())

# get the prediction either 1 or 0
    prediction=torch.round(output.squeeze())
```

```
    correct_tensor=prediction.eq(labels.float().view_as(prediction))
    correct=np.squeeze(correct_tensor.cpu().numpy())
    correct_prediction+=np.sum(correct)

#print(np.mean(test_losses))
print("Accuracy: {:.4f}".format(correct_prediction/len(test_loader.dataset)))



deftokenize_sentence(sentence):
  sentence=sentence.lower()

#remove punctuation
  sentence=''.join([letter for letter in sentence if letter notin
punctuation])

  test_words=sentence.split()
  tokens=[]
  sample=[]
for word in test_words:
if word in vocab_to_int:
          sample.append(word)
  tokens.append([vocab_to_int[word] for word in sample])
return tokens


defpredict(model,sentence,sequence_length=200):
  model.cuda()
  model.eval()

  test_ints=tokenize_sentence(sentence)
  features=featuresPadding(test_ints,sequence_length)
  feature_tensor=torch.from_numpy(features)
  batch_size=feature_tensor.size(0)

  hidden_state=model.initialize_hidden_state(batch_size)
  feature_tensor=feature_tensor.cuda().long()
  output,hidden_state=model(feature_tensor,hidden_state)
  prediction=torch.round(output.squeeze())
if(prediction.item()==0):
print("{:.4f}\t Negative sentence!".format(output.item()))
else:
print("{:.4f}\t Positive sentence!".format(output.item()))


negativeDataPath="aclImdb/train/neg/"
positiveDataPath="aclImdb/train/pos/"
fileLocation='/content/drive/My Drive/SEM-III
Project/SentimentAnalysisCheckpoint.pth'
```

```
reviews,labels=createData(positiveDataPath,negativeDataPath)

len(reviews),len(labels)

seq_length=200


features,encoded_labels,vocab_to_int=getFeaturesLabels(reviews,labels,seq_leng
th)

train_x,train_y,test_x,test_y,val_x,val_y=createTrainTestValidateData(0.8,feat
ures,encoded_labels)


train_data = TensorDataset(torch.from_numpy(train_x),
torch.from_numpy(train_y))
validation_data = TensorDataset(torch.from_numpy(val_x),
torch.from_numpy(val_y))
test_data = TensorDataset(torch.from_numpy(test_x), torch.from_numpy(test_y))
batch_size=100
train_loader = DataLoader(train_data, shuffle=True, batch_size=batch_size)
validation_loader = DataLoader(validation_data, shuffle=True,
batch_size=batch_size)
test_loader = DataLoader(test_data, shuffle=True, batch_size=batch_size)


print("Training: \t",np.shape(train_x))
print("Validation: \t",np.shape(val_x))
print("Test: \t\t",np.shape(test_x))


model,criterion,optimizer=buildNetwork()


model.cuda()


print_interval = 100
epochs = 16


import time
start_time=time.time()
trainNetwork(model,epochs,print_interval,criterion,optimizer)
print("\n\nTraining Time: {} minutes".format((time.time()-start_time)/60))


checkAccuracy(model,test_loader)
```

```
checkAccuracy(model,train_loader)

checkAccuracy(model,validation_loader)

torch.cuda.empty_cache
```

Code to perform prediction on the basis of processed data dictionary and trained model

commons.py

```
import torch.nn as nn
import io
from string import punctuation
from collections import Counter
import torch
import numpy as np
#############################################################################
deffeaturesPadding(sentence,sequenceLength):
  features=np.zeros((len(sentence),sequenceLength),dtype=int)
for i,row inenumerate(sentence):
    features[i,-len(row):]=np.array(row)[:sequenceLength]
return features
#############################################################################
deftokenize_sentence(sentence,vocab_to_int):
  sentence=sentence.lower()
  sentence=''.join([letter for letter in sentence if letter notin
punctuation])
  test_words=sentence.split()
  tokens=[]
  sample=[]
for word in test_words:
if word in vocab_to_int:
          sample.append(word)
  tokens.append([vocab_to_int[word] for word in sample])
return tokens
#############################################################################
defpredict(model,sentence, vocab_to_int,sequence_length=200):
    test_ints=tokenize_sentence(sentence,vocab_to_int)
    features=featuresPadding(test_ints,sequence_length)
    feature_tensor=torch.from_numpy(features)
    batch_size=feature_tensor.size(0)

    hidden_state=model.initialize_hidden_state(batch_size)
    feature_tensor=feature_tensor.long()
    output,hidden_state=model(feature_tensor,hidden_state)
```

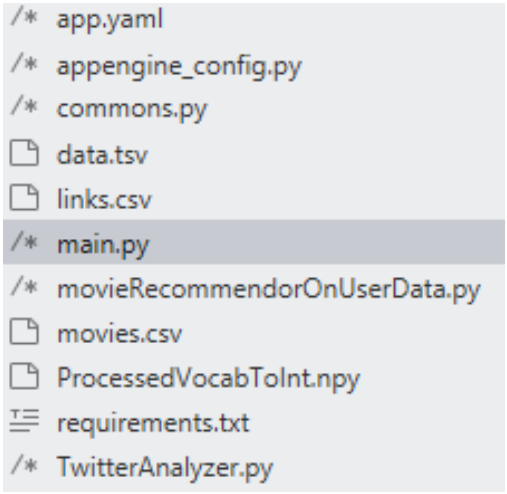
```
    prediction=torch.round(output.squeeze())
if(prediction.item()==0):
       resultString="{:.4f}  Negative sentence!".format(output.item())
else:
       resultString="{:.4f}  Positive sentence!".format(output.item())
return resultString
########################################################################
defcreateData():
    vocab_to_int = np.load('vocab_to_int.npy').item()
return vocab_to_int
############################################################
classSentimentNetwork(nn.Module):
def__init__(self,vocabulary_size,output_size,embedding_dimension,hidden_dimens
ion,number_of_layers,dropout_probability=0.5):
super(SentimentNetwork,self).__init__()
self.output_size=output_size
self.number_of_layers=number_of_layers
self.hidden_dimension=hidden_dimension
self.embedding=nn.Embedding(vocabulary_size,embedding_dimension)
self.lstm=nn.LSTM(embedding_dimension,hidden_dimension,number_of_layers,dropou
t=dropout_probability,batch_first=True)
self.dropout=nn.Dropout(dropout_probability)
self.finalLayer=nn.Linear(hidden_dimension,output_size)
self.sigmoid=nn.Sigmoid()

defforward(self,x,hidden):
    batch_size=x.size(0)
    embedding_output=self.embedding(x)
    lstm_output,hidden=self.lstm(embedding_output,hidden)
    lstm_output=lstm_output.contiguous().view(-1,self.hidden_dimension)
    output=self.dropout(lstm_output)
    output=self.finalLayer(output)
    sigmoid_output=self.sigmoid(output)
    sigmoid_output=sigmoid_output.view(batch_size,-1)
    sigmoid_output=sigmoid_output[:,-1]
return sigmoid_output,hidden

definitialize_hidden_state(self,batch_size):
    weight=next(self.parameters()).data

hidden=(weight.new(self.number_of_layers,batch_size,self.hidden_dimension).zer
o_(),weight.new(self.number_of_layers,batch_size,self.hidden_dimension).zero_(
))
return hidden
########################################################################
##############
from torch.optim import Optimizer
import  math
```

```
classAdamOptimizerAlgorithm(Optimizer):


def__init__(self, params, lr=1e-3, betas=(0.9, 0.999), eps=1e-8):
        defaults = dict(lr=lr, betas=betas, eps=eps)
super(AdamOptimizerAlgorithm, self).__init__(params, defaults)

defstep(self):
for group inself.param_groups:
for p in group['params']:
if p.grad isNone:
continue
                grad = p.grad.data
                state = self.state[p]

# State initialization in case of initial state
iflen(state) == 0:
                    state['step'] = 0
# Exponential moving average of gradient values
                    state['exp_avg'] = torch.zeros_like(p.data)
# Exponential moving average of squared gradient values
                    state['exp_avg_sq'] = torch.zeros_like(p.data)


                exp_avg, exp_avg_sq = state['exp_avg'], state['exp_avg_sq']
                beta1, beta2 = group['betas']

                state['step'] += 1

#following is just a formula implementaion of the algorithm
                exp_avg.mul_(beta1).add_(1 - beta1, grad)
                exp_avg_sq.mul_(beta2).addcmul_(1 - beta2, grad, grad)

                denom = exp_avg_sq.sqrt().add_(group['eps'])

                b_1 = 1 - beta1 ** state['step']
                b_2 = 1 - beta2 ** state['step']
                step_size = group['lr'] * math.sqrt(b_2) / b_1

                p.data.addcdiv_(-step_size, exp_avg, denom)
#############################################################################
#######
defbuildNetwork(vocab_to_int):
  vocabulary_size=len(vocab_to_int)+1
  output_size=1
  embedding_dimension=600
  hidden_dimension=512
  number_of_layers=3
```

```
model=SentimentNetwork(vocabulary_size,output_size,embedding_dimension,hidden_
dimension,number_of_layers)
  learning_rate=0.003
  criterion=nn.MSELoss()
  optimizer=AdamOptimizerAlgorithm(model.parameters(),lr=learning_rate)
return model
##############################################################################
###########
defload_checkpoint(filepath,vocab_to_int):
    model=buildNetwork(vocab_to_int)
    model.load_state_dict(torch.load(filepath,map_location='cpu'))
    model.cpu()
return model
##############################################################################
###############
defgetSentimentPredictionResult(inputReview):
    vocab_to_int=createData()
    model=load_checkpoint('SentimentAnalysisCheckpoint.pth',vocab_to_int)
    seq_length=200
return predict(model,inputReview,vocab_to_int,seq_length)
```

13.2 Appendix B: Google Cloud App Engine

List of Files:



1. app.yaml

```
#the app runtime will be on python
runtime: python

#App Engine flexible environment automatically scales your app up and down
while balancing the load.
env: flex

#timeout is 120 seconds, "-t 120" is not mentioned it will be by default of 30
secs
entrypoint: gunicorn -t 120 -b :$PORT main:app

runtime_config:
python_version: 3

manual_scaling:
instances: 1

# 4 cpu cores with 20 gb ram
# for a single core max ram can be 6.5 GB, hence for 4 cores 4x6.5 is max
resources:
cpu: 4
memory_gb: 20
disk_size_gb: 20

handlers:
- url: /.*
```

```
script: auto
```

2. appengine_config.py

```
from google.appengine.ext import vendor

# Add any libraries installed in the "lib" folder.
vendor.add('lib')
```

3. commons.py

```
''' same code from the Training With Preprocessed Data Notebook
    with getSetSentimentPredictionResult function written to load
    model, perform tokenization on given string and return the result
 '''

import torch.nn as nn
import io
from string import punctuation
from collections import Counter
import torch
import numpy as np
#################################################################################
deffeaturesPadding(sentence,sequenceLength):
  features=np.zeros((len(sentence),sequenceLength),dtype=int)
for i,row inenumerate(sentence):
    features[i,-len(row):]=np.array(row)[:sequenceLength]
return features
#################################################################################
deftokenize_sentence(sentence,vocab_to_int):
  sentence=sentence.lower()
  sentence=''.join([letter for letter in sentence if letter notin
punctuation])
  test_words=sentence.split()
  tokens=[]
  sample=[]
for word in test_words:
if word in vocab_to_int:
          sample.append(word)
  tokens.append([vocab_to_int[word] for word in sample])
return tokens
#################################################################################

defpredict(model,sentence, vocab_to_int,sequence_length=200):
    test_ints=tokenize_sentence(sentence,vocab_to_int)
try:
```

```
        features=featuresPadding(test_ints,sequence_length)
        feature_tensor=torch.from_numpy(features)
        batch_size=feature_tensor.size(0)

        hidden_state=model.initialize_hidden_state(batch_size)
        feature_tensor=feature_tensor.long()

        output,hidden_state=model(feature_tensor,hidden_state)
        prediction=torch.round(output.squeeze())
if(prediction.item()==0):
            resultString="{:.4f}  Negative sentence!".format(output.item())
else:
            resultString="{:.4f}  Positive sentence!".format(output.item())

except:
        resultString="0"

return resultString
#########################################################################
defcreateData():
    vocab_to_int = np.load('ProcessedVocabToInt.npy',allow_pickle=True).item()
return vocab_to_int
#################################################################
classSentimentNetwork(nn.Module):
def__init__(self,vocabulary_size,output_size,embedding_dimension,hidden_dimens
ion,number_of_layers,dropout_probability=0.5):
super(SentimentNetwork,self).__init__()
self.output_size=output_size
self.number_of_layers=number_of_layers
self.hidden_dimension=hidden_dimension
self.embedding=nn.Embedding(vocabulary_size,embedding_dimension)
self.lstm=nn.LSTM(embedding_dimension,hidden_dimension,number_of_layers,dropou
t=dropout_probability,batch_first=True)
self.dropout=nn.Dropout(dropout_probability)
self.finalLayer=nn.Linear(hidden_dimension,output_size)
self.sigmoid=nn.Sigmoid()

defforward(self,x,hidden):
    batch_size=x.size(0)
    embedding_output=self.embedding(x)
    lstm_output,hidden=self.lstm(embedding_output,hidden)
    lstm_output=lstm_output.contiguous().view(-1,self.hidden_dimension)
    output=self.dropout(lstm_output)
    output=self.finalLayer(output)
    sigmoid_output=self.sigmoid(output)
    sigmoid_output=sigmoid_output.view(batch_size,-1)
    sigmoid_output=sigmoid_output[:,-1]
return sigmoid_output,hidden
```

```
definitialize_hidden_state(self,batch_size):
    weight=next(self.parameters()).data

hidden=(weight.new(self.number_of_layers,batch_size,self.hidden_dimension).zer
o_(),weight.new(self.number_of_layers,batch_size,self.hidden_dimension).zero_(
))

return hidden
################################################################################
##############
from torch.optim import Optimizer
import  math
################################################################################
#######
defbuildNetwork(vocab_to_int):
  vocabulary_size=len(vocab_to_int)+1
  output_size=1
  embedding_dimension=400
  hidden_dimension=256
  number_of_layers=3

model=SentimentNetwork(vocabulary_size,output_size,embedding_dimension,hidden_
dimension,number_of_layers)
return model
################################################################################
############
defload_checkpoint(filepath,vocab_to_int):
    model=buildNetwork(vocab_to_int)
    model.load_state_dict(torch.load(filepath,map_location='cpu'))
return model
################################################################################
###############
defgetSentimentPredictionResult(inputReview):
    vocab_to_int=createData()
    model=load_checkpoint('checkpoint.pth',vocab_to_int)
    seq_length=200
    result=predict(model,inputReview,vocab_to_int,seq_length)

# once done clear out the memory by deleting model to reduce memory usage
del model
del vocab_to_int
del inputReview
return result
```

4. main.py

```
from flask import Flask,request
from flask_restful import Resource,Api
```

```
from commons import *
import sys
import logging
import os
from TwitterAnalyzer import *
from movieRecommendorOnUserData import *

app=Flask(__name__)
api=Api(app)

#########################################################################
'''
Input : Query string to be searched on twitter and analyze the percentage of
Positive tweets about that topic
Output: Percentage value of positive tweets
Uses: 1. Trainded model to analyzer each tweet's sentiment.
      2. Vocabulary to integer mapping dictonary to encode tweets.
      3. Twitter API key to retrive tweets
'''
classAnalyze(Resource):
    defget(self,query):
        result=query.split('+')
        search=' '.join(result)
        final_result=getTwitterSentiment(search)
        del search
        del result
        return {'result ':final_result}

#########################################################################
'''
Input: A text whose sentiment result is about to be found
Output: Sentiment Score(Between 0-1) with Positive/Negative Label
Uses: 1. Trained Model for prediction.
      2. Vocabulary to Integer Mapping dictonary to encode input.
'''
classMulti(Resource):
    defget(self,num):
        result=num.split('+')
        review=' '.join(result)
        predicted_result=getSentimentPredictionResult(review)
        del review
        del result
        return {'result ':predicted_result}

#########################################################################
'''
Input:  UserID and Movie Title.
```

```
Output: Returns successs message after adding the movie to user likes. If user
is new, then new data entry is created for user
Uses: 1. User's Like and Dislike data mapped with UserID
'''

classAddLike(Resource):
    defget(self,likedMovie):
        #print("\n================\nPath Status")
        #print(os.path.isdir("/home"))

        result=likedMovie.split('+')
        userID=result[0]
        result=result[1:]
        movieTitle=' '.join(result)
        final_result=addNewLike(userID,movieTitle)
        return {'result ':final_result}

################################################################################
'''
Input:  UserID and Movie Title.(Format: UserID+MovieTitle, Example:
user123+Toy+Story)
Output: Returns successs message after adding the movie to user dislikes. If
user is new, then new data entry is created for user
Uses: 1. User's Like and Dislike data mapped with UserID
'''
classAddDislike(Resource):
    defget(self,dislikedMovie):
        result=dislikedMovie.split('+')
        userID=result[0]
        result=result[1:]
        movieTitle=' '.join(result)
        final_result=addNewDislike(userID,movieTitle)
        return {'result ':final_result}

################################################################################
'''
Input: UserID
Output: JSON response of recommended movies to the user, each entry containing
            1. Movie Title
            2. Poster URL
            3. Overview
            4. Release Date
Uses: Content based recommendation method
'''
classGetRecommendation(Resource):
    defget(self,userID):
        final_result=getRecommendedMoviesID(userID)
        return final_result
```

```
# Bind each class to corresponding calling strings
api.add_resource(Multi,'/sentiment/<num>')
api.add_resource(Analyze,'/analyze/<query>')
api.add_resource(AddLike,'/addLike/<likedMovie>')
api.add_resource(AddDislike,'/addDislike/<dislikedMovie>')
api.add_resource(GetRecommendation,'/getRecommendation/<userID>')


if__name__=='__main__':
    app.run(port=os.getenv('PORT',5000))
    app.logger.addHandler(logging.StreamHandler(sys.stdout))
    app.logger.setLevel(logging.ERROR)
```

5. movieRecommendorOnUserData.py

```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import linear_kernel
import urllib.request
import json
import os
import psutil

userLikeDataLocation='/home/userLikeData.npy'
userDislikeDataLocation='/home/userDislikeData.npy'

defcreateFiles():
    like={}
    dislike={}
    saveUserData(userLikeDataLocation,like)
    saveUserData(userDislikeDataLocation,dislike)
print("\n\n===============Creating Files===============\n\n")

defgetTitleIndex(title,tempDF):
try:
return tempDF[tempDF['title'].str.contains(
title+".*",regex=True,na=False,case=False)]['movieId'].tolist()[0]
except:
return -1

defgetRating(imdb_id,ratings):
try:
return ratings[ratings['tconst']==imdb_id]['averageRating'].tolist()[0]
except:
return -1
```

```
defgetRecommendedMoviesID(userID):
    ratings=pd.read_csv('data.tsv', sep='\t', header=0)
    ratings=ratings[['tconst','averageRating']]

if(not os.path.isfile(userLikeDataLocation)):
        createFiles()

print("\n=========Status: Load user data")

    userData=loadUserData(userLikeDataLocation)
ifnot userID in userData.keys():
return {"status":"False",
"result":"User is not in database"}
else:
        tempList=userData[userID]
        title=""
if(len(tempList)>0):
            title=tempList[-1]
else:
return {"status":"False",
"result":"User hasn't liked any movie so far"}

        userData=loadUserData(userDislikeDataLocation)
        dislikeList=[]
if userID in userData.keys():
            dislikeList=userData[userID]



        MoviesID=[]
        data_location="movies.csv"
        movies=pd.read_csv(data_location)
print("\n=========Status: Reding movies")


tf=TfidfVectorizer(analyzer="word",ngram_range=(1,2),min_df=0,stop_words="engl
ish")
print(psutil.virtual_memory())
print("\n=========Status: Creating TfIdfMatrix")
        tfidf_matrix=tf.fit_transform(movies['genres'])
print(psutil.virtual_memory())
print("\n=========Status: Created TfIdfMatrix")
print(psutil.virtual_memory())

        cosine_sim=linear_kernel(tfidf_matrix,tfidf_matrix)
print(psutil.virtual_memory())
print("\n=========Status: Creted CosinefMatrix")
```

```
        titles=movies['title']
        indices=pd.Series(movies.index,index=movies['title'])

        tempDF=movies[['movieId','title']].iloc[:,:]

        idx=getTitleIndex(title,tempDF)
if(idx==-1):
return {"status":"False",
"result":"No movie with the "+title+" title in database"}
else:
            idx=idx-1
            sim_scores=list(enumerate(cosine_sim[idx]))
            sim_scores=sorted(sim_scores,key=lambdax: x[1],reverse=True)
            sim_scores=sim_scores[1:21]
            movie_indices=[i[0] for i in sim_scores]
            RecommendationResult=titles.iloc[movie_indices]

            linkDF=pd.read_csv('links.csv')

for element in RecommendationResult[:15]:

searchID=movies[movies['title']==element]['movieId'].tolist()[0]

imdbIndex=linkDF[linkDF['movieId']==searchID]['imdbId'].tolist()[0]
                imdbIndex=str(imdbIndex)
while(notlen(imdbIndex)==8):
                    imdbIndex='0'+imdbIndex

                MoviesID.append('tt'+imdbIndex)

#######################################
            imdbIndex=linkDF[linkDF['movieId']==idx+1]['imdbId'].tolist()[0]
            imdbIndex=str(imdbIndex)
while(notlen(imdbIndex)==8):
                imdbIndex='0'+imdbIndex
            TitleIndexID='tt'+imdbIndex

            titleIndexScore=getRating(TitleIndexID,ratings)
            finalResult=[]
for element in MoviesID:
                score=getRating(element,ratings)
                score=10-abs(score-titleIndexScore)
                finalResult.append((element,score))

            contents=sorted(finalResult, key=lambdax: x[1],reverse=True)
            contents=contents[:10]
            resultDict=[]
```

```
for url,_ in contents:
                requestURL="https://www.omdbapi.com/?apikey=815fe2a8&i="+url
                element=urllib.request.urlopen(requestURL).read()
                root=json.loads(element)
ifnot root['Title'] in dislikeList:
                    movieDict={
"title":root['Title'],
"release":root['Released'],
"overview":root['Plot'],
"genre":root['Genre'],
"poster":root['Poster'],
                            }
                    resultDict.append(movieDict)

                responseJSON={"status":"True",
"result":resultDict}


return responseJSON



defloadUserData(fileLocation):
return np.load(fileLocation,allow_pickle=True).item()

defsaveUserData(fileLocation,data):
    np.save(fileLocation,data)


defaddNewLike(userID,movieTitle):
if(not os.path.isfile(userLikeDataLocation)):
        createFiles()

    userData=loadUserData(userLikeDataLocation)
if userID in userData.keys():
        tempList=userData[userID]
ifnot movieTitle in tempList:
            tempList.append(movieTitle)
if(len(tempList)>5):
                tempList=tempList[1:]
        userData[userID]=tempList

try:
            dislikeData=loadUserData(userDislikeDataLocation)
            dislikeData[userID].remove(movieTitle)
            saveUserData(userDislikeDataLocation,dislikeData)
except:
del dislikeData
```

```
else:
        userData[userID]=[movieTitle]

    saveUserData(userLikeDataLocation,userData)
returnTrue


defaddNewDislike(userID,movieTitle):
if(not os.path.isfile(userLikeDataLocation)):
        createFiles()

    userData=loadUserData(userDislikeDataLocation)
if userID in userData.keys():
        tempList=userData[userID]
ifnot movieTitle in tempList:
            tempList.append(movieTitle)
        userData[userID]=tempList

try:
            likeData=loadUserData(userLikeDataLocation)
            likeData[userID].remove(movieTitle)
            saveUserData(userLikeDataLocation,likeData)
except:
del likeData

else:
        userData[userID]=[movieTitle]

    saveUserData(userDislikeDataLocation,userData)
returnTrue
```

6. requirements.txt

```
Flask==1.0.2
numpy==1.14.6
https://download.pytorch.org/whl/cpu/torch-1.1.0-cp36-cp36m-linux_x86_64.whl
https://download.pytorch.org/whl/cpu/torchvision-0.3.0-cp36-cp36m-
linux_x86_64.whl
gunicorn==19.9.0
Flask-RESTful
tweepy==3.7.0
pandas==0.24.2
sklearn
psutil
```

7. TwitterAnalyzer.py

```
'''All necessary imports
 'commons' is imported to use the getSentimentPredictionResult function
```

```
 All print statements are not required, but are given so that user can see the
log messages
 using 'gcloud app logs tail' if deployed online, else can run locally and see
the result.
'''
import tweepy
import pandas as pd
import csv
from commons import *
import time
import re
import os

#############################################################################

defreadDataFromTwitter(SearchString):
#To login you need Goto twitter developer console and generate your own keys
    consumer_key =
    consumer_secret =
    access_token =
    access_token_secret =


#Perform authentication with Twitter
    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    api = tweepy.API(auth,wait_on_rate_limit=True)
    reviews=[]
print("\n\tStatus: Logged in successfully.")

#Search the string to get 300 twitter results

results=tweepy.Cursor(api.search,q=SearchString,tweet_mode='extended').items(3
00)
print("\n\tStatus: Reading data.")
for review in results:
        reviews.append(review.full_text)
print("\n\tStatus: Reading data complete.")

#store the indexes and convert the generated list to a dataframe
    indexes=list(range(len(reviews)))
    df=pd.DataFrame({'Index': indexes,'Tweet': reviews})

print("\n\tStatus: Saving original data.")

print("\n\tStatus: Original data saved successfully.")
#return dataframe
return df
```

```
#############################################################################
'''
Funciton to clean the tweets by removing:
1. Emojis
2. URLs
3. Additional Padding
4. Tweets with RT only
'''
defcleanTweet(text):
    result=re.sub(r'@[A-Za-z0-9]+','',text)
    result=re.sub('https?://[A-Za-z0-9./]+','',result)
    result.replace(u"\ufffd", "?")
    result=re.sub("[^a-zA-Z]", " ",result)
    result=re.sub("RT", "",result)
return result

#############################################################################

'''
Function to clean whole dataframe and return only those which will be fed to
model
It removes:
1. Duplicate Tweets
2. Tweets with length less than 50 character after clearning with Regular
Expression
'''
defcleanData(df):
print("\n\tStatus: Reading and cleaning unwanted data.")

    dropIndex=[]
    dfNew=pd.DataFrame(columns=['Index','Tweet'])

for i inrange(len(df['Tweet'])):
        result=cleanTweet(df['Tweet'][i])

if(result==""):
            dropIndex.append(i)
elif(len(result)<50):
            dropIndex.append(i)
else:
            dfNew=dfNew.append({'Tweet':result},ignore_index=True)


print("\n\tStatus: Total dropped data: {}".format(len(dropIndex)))
print("\n\tStatus: Data cleaned successfully.")
    tempSize=len(dfNew)

#Drop all duplicate data from dataframe
    dfNew=dfNew.drop_duplicates(subset=['Tweet'], keep='first', inplace=False)
```

```
#dfNew.to_csv('Cleaned'+fileName)
print("\n\tStatus: No of dropped duplicates :{}".format(tempSize-len(dfNew)))

#return dataframe with unique tweets
return dfNew

#############################################################################

defgetTwitterSentiment(SearchString):
#start the time count
    start_time=time.time()
    fileName=readDataFromTwitter(SearchString)

#call to clean tweets
    cleanedDataFrame=cleanData(fileName)

# Load the model and vocabularyToInt mapping dictonary
print("\n\tStatus: Loading vocab.")
    vocab_to_int=createData()
print("\n\tStatus: Loading model.")
    model=load_checkpoint('checkpoint.pth',vocab_to_int)

# Set sequence length to same as the one used in training
    seq_length=200
print("\n\tStatus: Model loaded successfully.")

print("\n\tStatus: Initiating model and feeding inputs\n\tPlease wait it might
take a while.")
sum=0.0
    index=0
    invalid=0
    PositiveList=[]
    NegativeList=[]
    TotalList=[]

#For Each tweet pass it to model and place the score into Positive/Negative to
get percentage
for text in cleanedDataFrame['Tweet']:
        result=predict(model,text,vocab_to_int,seq_length)
if(result=='0'):
print("One invalid data entry found...ignoring that from calculation!")
            invalid+=1
else:
            if(float(result.split(" ")[0]))<0.5:
                NegativeList.append(float(result.split(" ")[0]))
            else:
                PositiveList.append(float(result.split(" ")[0]))
```

```
            sum+=float(result.split(" ")[0])

if(index%10==0):
print("\n\t\tProgress: {}/{}".format(index,len(cleanedDataFrame)))
        index+=1

print("\n\t\tProgress: {}/{}".format(index,len(cleanedDataFrame)))
print("\n\tStatus: Result returned successfully.")
print("\n\tTotal processing time: {} minutes".format((time.time()-
start_time)/60))
print("\n\tTotal Number of Processed Responses:
{}".format(len(PositiveList)+len(NegativeList)))
print("\n\tTotal Number of Positive Responses: {}".format(len(PositiveList)))
print("\n\tTotal Number of Negative Responses: {}".format(len(NegativeList)))
print("\n\tAverage: {}".format(sum/(len(PositiveList)+len(NegativeList))))

#Return percentage of positive tweets
return ((len(PositiveList))*100)/(len(PositiveList)+len(NegativeList))

#############################################################################
```

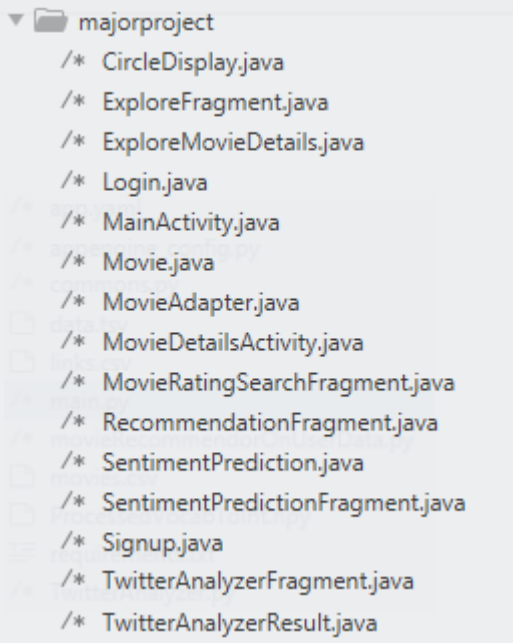
13.3 Appendix C: Android Application

AndroidMainfest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifestxmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.majorproject">

<uses-permissionandroid:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permissionandroid:name="android.permission.INTERNET"/>

<application
android:allowBackup="true"
android:hardwareAccelerated="true"
android:icon="@mipmap/ic_logo_university"
android:label="@string/app_name"
android:largeHeap="true"
android:roundIcon="@mipmap/ic_logo_university"
android:supportsRtl="true"
android:theme="@style/AppTheme">
<activityandroid:name=".TwitterAnalyzerResult"
android:label="Twitter Analyzer Result"></activity>
<activity
android:name=".ExploreMovieDetails"
android:label="Movie Details"
android:parentActivityName=".MainActivity"/>
<activity
android:name=".SentimentPrediction"
android:label="@string/sentiment_activity_result"
android:parentActivityName=".MainActivity"/>
<activity
android:name=".MovieDetailsActivity"
android:label="@string/movie_rating_result"
android:parentActivityName=".MainActivity"/>
<activity
android:name=".Login"
android:label="Login Screen"/>
<activity
android:name=".Signup"
android:label="Registration Screen"/>
<activity
android:name=".MainActivity"
android:configChanges="orientation|keyboardHidden"
android:label="@string/app_name"
android:screenOrientation="portrait"
android:theme="@style/AppThemeNavigation">
<intent-filter>
<actionandroid:name="android.intent.action.MAIN"/>
```

```
<categoryandroid:name="android.intent.category.LAUNCHER"/>
</intent-filter>
</activity>

<meta-data
android:name="preloaded_fonts"
android:resource="@array/preloaded_fonts"/>
</application>

</manifest>
```

List of Activity Files:



1. CircleDisplay.java

```
package com.example.majorproject;

import android.animation.ObjectAnimator;
import android.annotation.SuppressLint;
import android.content.Context;
import android.content.res.Resources;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Paint.Align;
import android.graphics.Paint.Style;
import android.graphics.PointF;
import android.graphics.RectF;
import android.util.AttributeSet;
```

```
import android.util.DisplayMetrics;
import android.util.Log;
import android.view.GestureDetector;
import android.view.GestureDetector.OnGestureListener;
import android.view.MotionEvent;
import android.view.View;
import android.view.animation.AccelerateDecelerateInterpolator;

import java.text.DecimalFormat;


@SuppressLint("NewApi")
publicclassCircleDisplayextendsViewimplementsOnGestureListener {

    /**
     * paint used for drawing the text
     */
publicstaticfinalintPAINT_TEXT = 1;
    /**
     * paint representing the value bar
     */
publicstaticfinalintPAINT_ARC = 2;
    /**
     * paint representing the inner (by default white) area
     */
publicstaticfinalintPAINT_INNER = 3;
privatestaticfinalStringLOG_TAG = "CircleDisplay";
    /**
     * the unit that is represented by the circle-display
     */
privateStringmUnit = "%";
    /**
     * startangle of the view
     */
privatefloatmStartAngle = 270f;
    /**
     * field representing the minimum selectable value in the display - the
     * minimum interval
     */
privatefloatmStepSize = 1f;
    /**
     * angle that represents the displayed value
     */
privatefloatmAngle = 0f;
    /**
     * current state of the animation
     */
privatefloatmPhase = 0f;
    /**
```

```
     * the currently displayed value, can be percent or actual value
     */
privatefloatmValue = 0f;
    /**
     * the maximum displayable value, depends on the set value
     */
privatefloatmMaxValue = 0f;
    /**
     * percent of the maximum width the arc takes
     */
privatefloatmValueWidthPercent = 50f;
    /**
     * if enabled, the inner circle is drawn
     */
privatebooleanmDrawInner = true;
    /**
     * if enabled, the center text is drawn
     */
privatebooleanmDrawText = true;
    /**
     * if enabled, touching and therefore selecting values is enabled
     */
privatebooleanmTouchEnabled = true;
    /**
     * represents the alpha value used for the remainder bar
     */
privateintmDimAlpha = 80;
    /**
     * the decimalformat responsible for formatting the values in the view
     */
privateDecimalFormatmFormatValue = newDecimalFormat("###,###,###,##0.0");
    /**
     * array that contains values for the custom-text
     */
privateString[] mCustomText = null;
    /**
     * rect object that represents the bounds of the view, needed for drawing
     * the circle
     */
privateRectFmCircleBox = newRectF();
privatePaintmArcPaint;
privatePaintmInnerCirclePaint;
privatePaintmTextPaint;
    /**
     * object animator for doing the drawing animations
     */
privateObjectAnimatormDrawAnimator;
    /**
```

```
     * boolean flag that indicates if the box has been setup
     */
privatebooleanmBoxSetup = false;
    /**
     * listener called when a value has been selected on touch
     */
privateSelectionListenermListener;
    /**
     * gesturedetector for recognizing single-taps
     */
privateGestureDetectormGestureDetector;

publicCircleDisplay(Contextcontext) {
super(context);
init();
    }

publicCircleDisplay(Contextcontext, AttributeSetattrs) {
super(context, attrs);
init();
    }

publicCircleDisplay(Contextcontext, AttributeSetattrs, intdefStyleAttr) {
super(context, attrs, defStyleAttr);
init();
    }

privatevoidinit() {

        mBoxSetup = false;

        mArcPaint = newPaint(Paint.ANTI_ALIAS_FLAG);
mArcPaint.setStyle(Style.FILL);
mArcPaint.setColor(Color.rgb(192, 255, 140));

        mInnerCirclePaint = newPaint(Paint.ANTI_ALIAS_FLAG);
mInnerCirclePaint.setStyle(Style.FILL);
mInnerCirclePaint.setColor(Color.WHITE);

        mTextPaint = newPaint(Paint.ANTI_ALIAS_FLAG);
mTextPaint.setStyle(Style.STROKE);
mTextPaint.setTextAlign(Align.CENTER);
mTextPaint.setColor(Color.BLACK);
mTextPaint.setTextSize(Utils.convertDpToPixel(getResources(), 24f));

        mDrawAnimator = ObjectAnimator.ofFloat(this, "phase", mPhase,
1.0f).setDuration(3000);
mDrawAnimator.setInterpolator(newAccelerateDecelerateInterpolator());
```

```
        mGestureDetector = newGestureDetector(getContext(), this);
    }

    @Override
protectedvoidonDraw(Canvascanvas) {
super.onDraw(canvas);

if (!mBoxSetup) {
            mBoxSetup = true;
setupBox();
        }

drawWholeCircle(canvas);

drawValue(canvas);

if (mDrawInner)
drawInnerCircle(canvas);

if (mDrawText) {

if (mCustomText != null)
drawCustomText(canvas);
else
drawText(canvas);
        }
    }

    /**
     * draws the text in the center of the view
     *
     * @paramc
     */
privatevoiddrawText(Canvasc) {
c.drawText(mFormatValue.format(mValue * mPhase) + " " + mUnit, getWidth() / 2,
getHeight() / 2 + mTextPaint.descent(), mTextPaint);
    }

    /**
     * draws the custom text in the center of the view
     *
     * @paramc
     */
privatevoiddrawCustomText(Canvasc) {

intindex = (int) ((mValue * mPhase) / mStepSize);
```

```
if (index <mCustomText.length) {
c.drawText(mCustomText[index], getWidth() / 2,
getHeight() / 2 + mTextPaint.descent(), mTextPaint);
        } else {
Log.e(LOG_TAG, "Custom text array not long enough.");
        }
    }

    /**
     * draws the background circle with less alpha
     *
     * @paramc
     */
privatevoiddrawWholeCircle(Canvasc) {
mArcPaint.setAlpha(mDimAlpha);

floatr = getRadius();

c.drawCircle(getWidth() / 2, getHeight() / 2, r, mArcPaint);
    }

    /**
     * draws the inner circle of the view
     *
     * @paramc
     */
privatevoiddrawInnerCircle(Canvasc) {

c.drawCircle(getWidth() / 2, getHeight() / 2, getRadius() / 100f
                * (100f - mValueWidthPercent), mInnerCirclePaint);
    }

    /**
     * draws the actual value slice/arc
     *
     * @paramc
     */
privatevoiddrawValue(Canvasc) {

mArcPaint.setAlpha(255);

floatangle = mAngle * mPhase;

c.drawArc(mCircleBox, mStartAngle, angle, true, mArcPaint);

// Log.i(LOG_TAG, "CircleBox bounds: " + mCircleBox.toString() +
// ", Angle: " + angle + ", StartAngle: " + mStartAngle);
    }
```

```
    /**
     * sets up the bounds of the view
     */
privatevoidsetupBox() {

intwidth = getWidth();
intheight = getHeight();

floatdiameter = getDiameter();

        mCircleBox = newRectF(width / 2 - diameter / 2, height / 2 - diameter
/ 2, width / 2
                + diameter / 2, height / 2 + diameter / 2);
    }

    /**
     * shows the given value in the circle view
     *
     * @paramtoShow
     * @paramtotal
     * @paramanimated
     */
publicvoidshowValue(floattoShow, floattotal, booleananimated) {

        mAngle = calcAngle(toShow / total * 100f);
        mValue = toShow;
        mMaxValue = total;

if (animated)
startAnim();
else {
            mPhase = 1f;
invalidate();
        }
    }

    /**
     * Sets the unit that is displayed next to the value in the center of the
     * view. Default "%". Could be "€" or "$" or left blank or whatever it is
     * you display.
     *
     * @paramunit
     */
publicvoidsetUnit(Stringunit) {
        mUnit = unit;
    }
```

```
    /**
     * Returns the currently displayed value from the view. Depending on the
     * used method to show the value, this value can be percent or actual
value.
     *
     * @return
     */
publicfloatgetValue() {
return mValue;
    }

publicvoidstartAnim() {
        mPhase = 0f;
mDrawAnimator.start();
    }

    /**
     * set the duration of the drawing animation in milliseconds
     *
     * @paramdurationmillis
     */
publicvoidsetAnimDuration(intdurationmillis) {
mDrawAnimator.setDuration(durationmillis);
    }

    /**
     * returns the diameter of the drawn circle/arc
     *
     * @return
     */
publicfloatgetDiameter() {
returnMath.min(getWidth(), getHeight());
    }

    /**
     * returns the radius of the drawn circle
     *
     * @return
     */
publicfloatgetRadius() {
returngetDiameter() / 2f;
    }

    /**
     * calculates the needed angle for a given value
     *
     * @parampercent
     * @return
```

```
     */
privatefloatcalcAngle(floatpercent) {
return percent / 100f * 360f;
    }

    /**
     * set the starting angle for the view
     *
     * @paramangle
     */
publicvoidsetStartAngle(floatangle) {
        mStartAngle = angle;
    }

    /**
     * returns the current animation status of the view
     *
     * @return
     */
publicfloatgetPhase() {
return mPhase;
    }

    /**
     * DONT USE THIS METHOD
     *
     * @paramphase
     */
publicvoidsetPhase(floatphase) {
        mPhase = phase;
invalidate();
    }

    /**
     * set this to true to draw the inner circle, default: true
     *
     * @paramenabled
     */
publicvoidsetDrawInnerCircle(booleanenabled) {
        mDrawInner = enabled;
    }

    /**
     * returns true if drawing the inner circle is enabled, false if not
     *
     * @return
     */
publicbooleanisDrawInnerCircleEnabled() {
```

```
return mDrawInner;
    }

    /**
     * set the drawing of the center text to be enabled or not
     *
     * @paramenabled
     */
publicvoidsetDrawText(booleanenabled) {
        mDrawText = enabled;
    }

    /**
     * returns true if drawing the text in the center is enabled
     *
     * @return
     */
publicbooleanisDrawTextEnabled() {
return mDrawText;
    }

    /**
     * set the color of the arc
     *
     * @paramcolor
     */
publicvoidsetColor(intcolor) {
mArcPaint.setColor(color);
    }

    /**
     * set the size of the center text in dp
     *
     * @paramsize
     */
publicvoidsetTextSize(floatsize) {
mTextPaint.setTextSize(Utils.convertDpToPixel(getResources(), size));
    }

    /**
     * set the thickness of the value bar, default 50%
     *
     * @parampercentFromTotalWidth
     */
publicvoidsetValueWidthPercent(floatpercentFromTotalWidth) {
        mValueWidthPercent = percentFromTotalWidth;
    }
```

```
    /**
     * Set an array of custom texts to be drawn instead of the value in the
     * center of the CircleDisplay. If set to null, the custom text will be
     * reset and the value will be drawn. Make sure the length of the array
corresponds with the maximum number of steps (set with setStepSize(float
stepsize).
     *
     * @paramcustom
     */
publicvoidsetCustomText(String[] custom) {
        mCustomText = custom;
    }

    /**
     * sets the number of digits used to format values
     *
     * @paramdigits
     */
publicvoidsetFormatDigits(intdigits) {

StringBufferb = newStringBuffer();
for (inti = 0; i < digits; i++) {
if (i == 0)
b.append(".");
b.append("0");
        }

        mFormatValue = newDecimalFormat("###,###,###,##0" + b.toString());
    }

    /**
     * set the aplha value to be used for the remainder of the arc, default 80
     * (use value between 0 and 255)
     *
     * @paramalpha
     */
publicvoidsetDimAlpha(intalpha) {
        mDimAlpha = alpha;
    }

    /**
     * sets the given paint object to be used instead of the original/default
     * one
     *
     * @paramwhich, e.g. CircleDisplay.PAINT_TEXT to set a new text paint
     * @paramp
     */
publicvoidsetPaint(intwhich, Paintp) {
```

```
switch (which) {
case PAINT_ARC:
                mArcPaint = p;
break;
case PAINT_INNER:
                mInnerCirclePaint = p;
break;
case PAINT_TEXT:
                mTextPaint = p;
break;
        }
    }

    /**
     * returns the current stepsize of the display, default 1f
     *
     * @return
     */
publicfloatgetStepSize() {
return mStepSize;
    }

    /**
     * Sets the stepsize (minimum selection interval) of the circle display,
     * default 1f. It is recommended to make this value not higher than 1/5 of
     * the maximum selectable value, and not lower than 1/200 of the maximum
     * selectable value. For a maximum value of 100 for example, a stepsize
     * between 0.5 and 20 is recommended.
     *
     * @paramstepsize
     */
publicvoidsetStepSize(floatstepsize) {
        mStepSize = stepsize;
    }

    /**
     * returns the center point of the view in pixels
     *
     * @return
     */
publicPointFgetCenter() {
returnnewPointF(getWidth() / 2, getHeight() / 2);
    }

    /**
     * returns true if touch-gestures are enabled, false if not
     *
```

```
     * @return
     */
publicbooleanisTouchEnabled() {
return mTouchEnabled;
    }

    /**
     * Enable touch gestures on the circle-display. If enabled, selecting
values
     * onTouch() is possible. Set a SelectionListener to retrieve selected
     * values. Do not forget to set a value before selecting values. By
default
     * the maxvalue is 0f and therefore nothing can be selected.
     *
     * @paramenabled
     */
publicvoidsetTouchEnabled(booleanenabled) {
        mTouchEnabled = enabled;
    }

    /**
     * set a selection listener for the circle-display that is called whenever
a
     * value is selected onTouch()
     *
     * @paraml
     */
publicvoidsetSelectionListener(SelectionListenerl) {
        mListener = l;
    }

    @Override
publicbooleanonTouchEvent(MotionEvente) {
if (mTouchEnabled) {

if (mListener == null)
Log.w(LOG_TAG,
"No SelectionListener specified. Use setSelectionListener(...) to set a
listener for callbacks when selecting values.");

// if the detector recognized a gesture, consume it
if (mGestureDetector.onTouchEvent(e))
returntrue;

floatx = e.getX();
floaty = e.getY();

// get the distance from the touch to the center of the view
```

```
floatdistance = distanceToCenter(x, y);
floatr = getRadius();

// touch gestures only work when touches are made exactly on the
// bar/arc
if (distance >= r - r * mValueWidthPercent / 100f&& distance < r) {

switch (e.getAction()) {

// case MotionEvent.ACTION_DOWN:
// if (mListener != null)
// mListener.onSelectionStarted(mValue, mMaxValue);
// break;
caseMotionEvent.ACTION_MOVE:

updateValue(x, y);
invalidate();
if (mListener != null)
mListener.onSelectionUpdate(mValue, mMaxValue);
break;
caseMotionEvent.ACTION_UP:
if (mListener != null)
mListener.onValueSelected(mValue, mMaxValue);
break;
                }
            }

returntrue;
        } else
returnsuper.onTouchEvent(e);
    }

    /**
     * updates the display with the given touch position, takes stepsize into
     * consideration
     *
     * @paramx
     * @paramy
     */
privatevoidupdateValue(floatx, floaty) {

// calculate the touch-angle
floatangle = getAngleForPoint(x, y);

// calculate the new value depending on angle
floatnewVal = mMaxValue * angle / 360f;

// if no stepsize
```

```
if (mStepSize == 0f) {
            mValue = newVal;
            mAngle = angle;
return;
        }

floatremainder = newVal % mStepSize;

// check if the new value is closer to the next, or the previous
if (remainder <= mStepSize / 2f) {

            newVal = newVal - remainder;
        } else {
            newVal = newVal - remainder + mStepSize;
        }

// set the new values
        mAngle = getAngleForValue(newVal);
        mValue = newVal;
    }

    @Override
publicbooleanonSingleTapUp(MotionEvente) {

// get the distance from the touch to the center of the view
floatdistance = distanceToCenter(e.getX(), e.getY());
floatr = getRadius();

// touch gestures only work when touches are made exactly on the
// bar/arc
if (distance >= r - r * mValueWidthPercent / 100f&& distance < r) {

updateValue(e.getX(), e.getY());
invalidate();

if (mListener != null)
mListener.onValueSelected(mValue, mMaxValue);
        }

returntrue;
    }

    /**
     * returns the angle relative to the view center for the given point on
the
     * chart in degrees. The angle is always between 0 and 360°, 0° is NORTH
     *
     * @paramx
```

```
     * @paramy
     * @return
     */
publicfloatgetAngleForPoint(floatx, floaty) {

PointFc = getCenter();

doubletx = x - c.x, ty = y - c.y;
doublelength = Math.sqrt(tx * tx + ty * ty);
doubler = Math.acos(ty / length);

floatangle = (float) Math.toDegrees(r);

if (x >c.x)
            angle = 360f - angle;

        angle = angle + 180;

// neutralize overflow
if (angle >360f)
            angle = angle - 360f;

return angle;
    }

    /**
     * returns the angle representing the given value
     *
     * @paramvalue
     * @return
     */
publicfloatgetAngleForValue(floatvalue) {
return value / mMaxValue * 360f;
    }

    /**
     * returns the value representing the given angle
     *
     * @paramangle
     * @return
     */
publicfloatgetValueForAngle(floatangle) {
return angle / 360f * mMaxValue;
    }

    /**
     * returns the distance of a certain point on the view to the center of
the
```

```
     * view
     *
     * @paramx
     * @paramy
     * @return
     */
publicfloatdistanceToCenter(floatx, floaty) {

PointFc = getCenter();

floatdist = 0f;

floatxDist = 0f;
floatyDist = 0f;

if (x >c.x) {
            xDist = x - c.x;
        } else {
            xDist = c.x - x;
        }

if (y >c.y) {
            yDist = y - c.y;
        } else {
            yDist = c.y - y;
        }

// pythagoras
        dist = (float) Math.sqrt(Math.pow(xDist, 2.0) + Math.pow(yDist, 2.0));

return dist;
    }

    @Override
publicbooleanonDown(MotionEvente) {
// TODO Auto-generated method stub
returnfalse;
    }

    @Override
publicbooleanonFling(MotionEvente1, MotionEvente2, floatvelocityX,
floatvelocityY) {
// TODO Auto-generated method stub
returnfalse;
    }

    @Override
publicvoidonLongPress(MotionEvente) {
```

```
// TODO Auto-generated method stub

    }

    @Override
publicbooleanonScroll(MotionEvente1, MotionEvente2, floatdistanceX,
floatdistanceY) {
// TODO Auto-generated method stub
returnfalse;
    }

    @Override
publicvoidonShowPress(MotionEvente) {
// TODO Auto-generated method stub

    }

    /**
     * listener for callbacks when selecting values ontouch
     *
     * @author Philipp Jahoda
     */
publicinterfaceSelectionListener {

        /**
         * called everytime the user moves the finger on the circle-display
         *
         * @paramval
         * @parammaxval
         */
voidonSelectionUpdate(floatval, floatmaxval);

        /**
         * called when the user releases his finger fromt he circle-display
         *
         * @paramval
         * @parammaxval
         */
voidonValueSelected(floatval, floatmaxval);
    }

publicstaticabstractclassUtils {

        /**
         * This method converts dp unit to equivalent pixels, depending on
         * device density.
         *
         * @paramdp A value in dp (density independent pixels) unit. Which we
```

```
         *           need to convert into pixels
         * @return A float value to represent px equivalent to dp depending on
         * device density
         */
publicstaticfloatconvertDpToPixel(Resourcesr, floatdp) {
DisplayMetricsmetrics = r.getDisplayMetrics();
floatpx = dp * (metrics.densityDpi / 160f);
return px;
        }
    }
}
```

2. ExploreFragment.java

```
package com.example.majorproject;

import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.GridView;
import android.widget.LinearLayout;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AlertDialog;
import androidx.fragment.app.Fragment;
import androidx.swiperefreshlayout.widget.SwipeRefreshLayout;

import com.github.ybq.android.spinkit.style.DoubleBounce;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.ByteArrayOutputStream;
```

```
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.nio.charset.Charset;
import java.util.ArrayList;
import java.util.List;

import javax.net.ssl.HttpsURLConnection;

publicclassExploreFragmentextendsFragment {

//Your APIKEY to peform request
StringAPIKEY=""

//API address to get trending movies
StringtrendingMoviesURL =
"https://api.themoviedb.org/3/movie/popular?api_key=APIKEY&language=en-
US&page=1";

//API address to get poster of size 200px width
StringposterHeaderURL = "https://image.tmdb.org/t/p/w200";

//List to store Movie objects which contain details about each movie
ArrayList<Movie>MoviesList;

//Used to inherit ArrayAdapter of Android
MovieAdaptermAdapter;

//GridView to show the movies in grid based layout
GridViewmovieGridView;

//to store the bitmap posters downloaded form internet
ArrayList<Bitmap>posterList;

//Fragment Layout container containing gridview to show the result
LinearLayoutexploreFragmentView;

//Textview to show warning and error messages received form server
TextViewexploreFragmentWarning;

//progressbar to show loading icon when network request is made
ProgressBarprogressBar;

/* Default method called by android os when a new view is created*/
publicViewonCreateView(LayoutInflaterinflater, ViewGroupcontainer,
BundlesavedInstanceState) {
// Inflate the layout for this fragment
```

```
Viewview = inflater.inflate(R.layout.fragment_explore, container, false);

//Initialize all variables
        movieGridView = view.findViewById(R.id.listViewMovies);
        exploreFragmentView =
view.findViewById(R.id.linearLayoutExploreFragment);
        exploreFragmentWarning =
view.findViewById(R.id.textViewExploreStatusWarning);
        MoviesList = newArrayList<>();
        posterList = newArrayList<>();
        progressBar = view.findViewById(R.id.spin_kit);
DoubleBouncemyProgressBar = newDoubleBounce();
progressBar.setIndeterminateDrawable(myProgressBar);

//execute network request to get the movies and their posters form internet
IMDBAsyncTasktask = newIMDBAsyncTask();
task.execute(trendingMoviesURL);

/*add action listener to grid view, on clicking the poster, new activity
ExploreMovieDetails
        will be launched to show detailed view of movie
         */
movieGridView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
publicvoidonItemClick(AdapterView<?>adapterView, Viewview, inti, longl) {
MoviecurrentMovie = mAdapter.getItem(i);
IntentmovieDetailScreen = newIntent(getActivity(), ExploreMovieDetails.class);
BitmaptempPoster = currentMovie.getPoster();
ByteArrayOutputStreamstream = newByteArrayOutputStream();
tempPoster.compress(Bitmap.CompressFormat.PNG, 100, stream);
byte[] byteArray = stream.toByteArray();

//add all necessary information about the movie to be passed on the
ExploreMovieDetails Activity
movieDetailScreen.putExtra("moviePoster", byteArray);
movieDetailScreen.putExtra("movieTitle", currentMovie.getTitle());
movieDetailScreen.putExtra("movieOriginalTitle",
currentMovie.getOriginalTitle());
movieDetailScreen.putExtra("string_overview", currentMovie.getOverview());
movieDetailScreen.putExtra("movieReleaseDate", currentMovie.getReleaseDate());
startActivity(movieDetailScreen);
            }
        });

/*
        Pull down to refresh function, if movies are already loaded then no
network will be initiated
        Otherwise initiate network request
```

```
        * */
finalSwipeRefreshLayoutpullToRefresh = view.findViewById(R.id.pullToRefresh);
pullToRefresh.setOnRefreshListener(new SwipeRefreshLayout.OnRefreshListener()
{
            @Override
publicvoidonRefresh() {
if (MoviesList.size() == 0) {
//if movie list is empty then call method to try download data
refreshData();
pullToRefresh.setRefreshing(false);
                } else {
//If movies are already loaded then show the toast message
Toast.makeText(getActivity(), "Already Loaded with latest data",
Toast.LENGTH_LONG).show();
                }
            }
        });
return view;
    }

/*Method to perform downloading of data when refresh action is triggered
     * */
publicvoidrefreshData() {
IMDBAsyncTasktask = newIMDBAsyncTask();
task.execute(trendingMoviesURL);
    }

/*If fragment is resumed from a paused state call super and set the title to
current "Explore"
     * */
publicvoidonResume() {
super.onResume();
        ((MainActivity) getActivity())
                .setActionBarTitle("Explore");
    }

/*Method to check if device is connected to internet
     * */
publicbooleanisNetworkAvailable() {
try {
ConnectivityManagermConnectivityManager = (ConnectivityManager)
getActivity().getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfomNetworkInfo = mConnectivityManager.getActiveNetworkInfo();
return mNetworkInfo != null;

        } catch (NullPointerExceptione) {
returnfalse;
        }
```

```
    }

/*
    Show alert dialog function, to be used by function when no internet
connection is found or any other errors
    */
publicvoidshowAlertDialog(StringTITLE, StringMESSAGE) {
new AlertDialog.Builder(getActivity())
                .setTitle(TITLE)
                .setMessage(MESSAGE)
                .setNegativeButton("OK", null)
                .setIcon(android.R.drawable.ic_dialog_alert)
                .show();
    }

/*If result is not Error then initiate the moviePoster downloader and download
movie posters on list,
    and then call the UI update.
    If error then show alert dialog about connection error
    */
publicvoiddoActions(Stringresult) {
JSONObjectroot = null;
if (result.equals("Error")) {
showAlertDialog("Connection Error", "No data connection found!");
        } else {
//if result is not empty then try processing the JSON response and retrieve
details about movies
try {
                root = newJSONObject(result);
StringtotalPages = root.getString("total_pages");
Toast.makeText(getActivity(), totalPages,
Toast.LENGTH_LONG).show();
JSONArrayresultsArray = root.getJSONArray("results");
Stringmovie_title;
Stringmovie_original_title;
Stringlanguage;
Stringoverview;
Stringrelease_date;
Stringposter_url;
Bitmapmovie_poster;
for (inti = 0; i <resultsArray.length(); i++) {
JSONObjectdetails = resultsArray.getJSONObject(i);
                    movie_title = details.getString("title");
                    movie_original_title =
details.getString("original_title");
                    language = details.getString("original_language");
                    overview = details.getString("overview");
                    release_date = details.getString("release_date");
```

```
                    poster_url = posterHeaderURL +
details.getString("poster_path");

//Call class constructor of Movie class with necessary details and add new
Movie to List
MoviesList.add(newMovie(movie_title, movie_original_title, language, overview,
release_date, poster_url));
                }
            } catch (JSONExceptione) {
e.printStackTrace();
            }
        }
//Perform Poster Reterival from the urls received form response
performPosterRetrival();
    }

/*Get all poster urls in a string of array and pass it as argument of
ImageDownloader  Async Task,
   to download the posters
    */
publicvoidperformPosterRetrival() {
String[] posterUrls = newString[MoviesList.size()];
for (inti = 0; i <MoviesList.size(); i++) {
            posterUrls[i] = (MoviesList.get(i).getPosterUrl());
        }
DownloadImageFromInternetimageDownload = newDownloadImageFromInternet();
imageDownload.execute(posterUrls);

    }

/*Set the movie poster to downloaded bitmap image, since we are showing only
thumbnail
    on ImageView it is much better to scale down resolution to save space.
    * */
publicvoidsendMoviePoster(List<Bitmap>result) {
for (inti = 0; i <result.size(); i++) {
Bitmapb = result.get(i);
//Trim the dimension of image to half, since we only need to show a thumbnail
            b = Bitmap.createScaledBitmap(b, (b.getWidth() / 2),
(b.getHeight() / 2), false);
MoviesList.get(i).setPoster(b);
        }
        mAdapter = newMovieAdapter(getActivity(), MoviesList);
movieGridView.setAdapter(mAdapter);

//hide the warning textview and progress bar and show the fragment containing
movies
exploreFragmentWarning.setVisibility(View.GONE);
```

```
progressBar.setVisibility(View.GONE);
exploreFragmentView.setVisibility(View.VISIBLE);
    }

/*
     * Inherited AsyncTask to download the movie details form server and
     * pass the received string to process as a JSON response to reterive the
information
     */
publicclassIMDBAsyncTaskextendsAsyncTask<String, Void, String> {
        @Override
protectedvoidonPreExecute() {
super.onPreExecute();
progressBar.setVisibility(View.VISIBLE);
        }

        /**
         * If network is available then process request otherwise return Error
as result
         */
        @Override
protectedStringdoInBackground(String... strings) {
if (isNetworkAvailable()) {
Stringres = "str";
try {
URLurl = newURL(strings[0]);
Log.v("IMDB URL:", "\n\t\t\t====================" + strings[0]);
InputStreaminputStream;
HttpsURLConnectionrequestConnection = (HttpsURLConnection)
url.openConnection();
requestConnection.setReadTimeout(200000);
requestConnection.setConnectTimeout(100000);
requestConnection.setRequestMethod("GET");
requestConnection.connect();
                    inputStream = requestConnection.getInputStream();
                    res = readFromStream(inputStream);
requestConnection.disconnect();
return res;
                } catch (Exceptione) {
e.printStackTrace();
                }
            }
return"Error";
        }

//call doActions function to perform update on UI
        @Override
protectedvoidonPostExecute(Stringresult) {
```

```
doActions(result);
        }

privateStringreadFromStream(InputStreaminputStream) throwsIOException {
StringBuilderoutput = newStringBuilder();
if (inputStream != null) {
InputStreamReaderinputStreamReader = newInputStreamReader(inputStream,
Charset.forName("UTF-8"));
BufferedReaderreader = newBufferedReader(inputStreamReader);
Stringline = reader.readLine();
while (line != null) {
output.append(line);
                    line = reader.readLine();
                }
            }
returnoutput.toString();
        }
    }

/*=========================================================================
===
    Async Task to download the posters form array of urls, passed as string
array.
    * */
privateclassDownloadImageFromInternetextendsAsyncTask<String, Void,
List<Bitmap>> {
protectedList<Bitmap>doInBackground(String... urls) {
List<Bitmap>bitmaps = newArrayList<Bitmap>();
//String imageURL = urls[0];
Bitmapbimage = null;
for (Stringurl: urls) {
try {
InputStreamin = new java.net.URL(url).openStream();
                    bimage = BitmapFactory.decodeStream(in);

                } catch (Exceptione) {
Log.e("Error Message", e.getMessage());
e.printStackTrace();
                }
bitmaps.add(bimage);
            }
return bitmaps;
        }

//call sendMoviePoster to finally load the posters and show the results
protectedvoidonPostExecute(List<Bitmap>result) {
sendMoviePoster(result);
        }
```

```
    }
}
```

3. ExploreMovieDetails.java

```
package com.example.majorproject;

import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.auth.FirebaseAuth;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.net.URL;
import java.net.URLEncoder;
import java.nio.charset.Charset;

import javax.net.ssl.HttpsURLConnection;
/*
Class to show the movie details: Name, Release date, Overview and Poster.
Functionality:
    Like Button: Add the movie to liked movies list by callnig server with
userID and movie title
    Dislike Button: Add the movie to disliked movies list by callnig server
with userID and movie title
    Find Out More: Launch browser to find more details about movie from google
serach
* */
```

```
publicclassExploreMovieDetailsextendsAppCompatActivityimplementsView.OnClickLi
stener {
//For showing movie details, basic viewholders of data
TextViewtextViewOverview; //Overview of movie
TextViewtextViewTitle; //Title
TextViewtextViewOriginalTitle; //Original Title
TextViewtextViewReleaseDate; //Release Data
ImageViewimageViewPoster;  //Poster Thumbnail

//Layout buttons and request URLs
StringmovieName;
ButtonbuttonLaunchGoogle;
ButtonbuttonLike, buttonDislike;
StringlikeURL;
StringdislikeURL;
StringuID;

//flag to identify like or dislike actions, to be used by toast method
Booleanflag = true;

//Firebase authentication object to retrieve userID
FirebaseAuthfirebaseAuth;

/*Method by default called by Android whenever new View is created*/
    @Override
protectedvoidonCreate(BundlesavedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_explore_movie_details);

//Server API address for Like/Dislike Action
        likeURL = "https://major-project-final-246818.appspot.com/addLike/";
        dislikeURL = "https://major-project-final-
246818.appspot.com/addDislike/";

//Initialize all views and variables
        textViewTitle = findViewById(R.id.textViewMovieTitle);
        textViewOriginalTitle = findViewById(R.id.textViewMovieOriginalTitle);
        textViewReleaseDate = findViewById(R.id.textViewReleaseDate);
        textViewOverview = findViewById(R.id.textViewOverview);
        imageViewPoster = findViewById(R.id.imageViewPoster);
        buttonLaunchGoogle = findViewById(R.id.buttonLaunchInternet);
        buttonLike = findViewById(R.id.buttonLike);
        buttonDislike = findViewById(R.id.buttonDislike);

//Parse the values that was passed by previous view class
Bundleextras = getIntent().getExtras();
byte[] byteArray = extras.getByteArray("moviePoster");
```

```
Bitmapmovie Poster = BitmapFactory.decodeByteArray(byteArray, 0,
byteArray.length);

//Set the details of movie
imageViewPoster.setImageBitmap(moviePoster);
textViewOverview.setText("Overview" + "\n" +
extras.getString("string_overview"));
        movieName = extras.getString("movieTitle");
textViewTitle.setText("Title: " + movieName);
textViewOriginalTitle.setText("Original Title: " +
extras.getString("movieOriginalTitle"));
textViewReleaseDate.setText("Release Date: " +
extras.getString("movieReleaseDate"));

//Add action listener to the Like, Dislike and More details button
buttonLaunchGoogle.setOnClickListener(this);
buttonLike.setOnClickListener(this);
buttonDislike.setOnClickListener(this);

//get the user id to be added with url request to call API function
        firebaseAuth = FirebaseAuth.getInstance();
        uID = "user" + firebaseAuth.getCurrentUser().getUid();

//Process the user Like/Dislike API call address in advance
//Format is: API Address/UserID+MovieTitle
String[] input = movieName.split(" ");
Stringaddress2 = "";
for (inti = 0; i <input.length - 1; i++) {
            address2 += input[i] + "+";
        }
        address2 += input[input.length - 1];
        likeURL = likeURL + uID + "+" + address2;
        dislikeURL = dislikeURL + uID + "+" + address2;

    }

//For each layout clicked call the necessary functions to perform actions
accordingly
    @Override
publicvoidonClick(Viewview) {
if (view == buttonLaunchGoogle) {
//Launch the browser to find out more details about movie
StringescapedQuery = null;
try {
                escapedQuery = URLEncoder.encode(movieName, "UTF-8");
            } catch (UnsupportedEncodingExceptione) {
e.printStackTrace();
            }
```

```
Uriuri = Uri.parse("http://www.google.com/#q=" + escapedQuery);
Intentintent = newIntent(Intent.ACTION_VIEW, uri);
startActivity(intent);
        }
if (view == buttonLike) {
//Call server method to add movie to liked list for user
            flag = true;
buttonLike.setBackground(getResources().getDrawable(R.drawable.ic_button_like_
red));
buttonDislike.setBackground(getResources().getDrawable(R.drawable.ic_dislike))
;
LikeDislikeAsyncTasktask = newLikeDislikeAsyncTask();
task.execute(likeURL);
        }
if (view == buttonDislike) {
//Call server method to add movie to disliked list for user
            flag = false;
buttonLike.setBackground(getResources().getDrawable(R.drawable.ic_like));
buttonDislike.setBackground(getResources().getDrawable(R.drawable.ic_button_di
slike_red));
LikeDislikeAsyncTasktask = newLikeDislikeAsyncTask();
task.execute(dislikeURL);
        }
    }

/*Method to check if network is available*/
publicbooleanisNetworkAvailable() {
try {
ConnectivityManagermConnectivityManager = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfomNetworkInfo = mConnectivityManager.getActiveNetworkInfo();
return mNetworkInfo != null;

        } catch (NullPointerExceptione) {
returnfalse;
        }
    }

/*Method to call toast request for each response*/
publicvoiddoActions(Stringresult) {
if (result.equals("Error")) {
showToast("Server communication failure, please try again later");
        } else {
if (flag == true) {
showToast("Added to likes");
            } else {

showToast("Added to dislikes");
```

```
            }
        }
    }

/*Method to show short time toast messages*/
publicvoidshowToast(StringtoastMessage) {
Toast.makeText(ExploreMovieDetails.this, toastMessage,
Toast.LENGTH_LONG).show();
    }

/*Send data to url passed, the function written on sever will save data about
user accordingly*/
publicclassLikeDislikeAsyncTaskextendsAsyncTask<String, Void, String> {
        @Override
protectedvoidonPreExecute() {
super.onPreExecute();
        }

        @Override
protectedStringdoInBackground(String... strings) {
if (isNetworkAvailable()) {
Stringres = "str";
try {
URLurl = newURL(strings[0]);
InputStreaminputStream;
HttpsURLConnectionrequestConnection = (HttpsURLConnection)
url.openConnection();
requestConnection.setReadTimeout(200000);
requestConnection.setConnectTimeout(100000);
requestConnection.setRequestMethod("GET");
requestConnection.connect();
                    inputStream = requestConnection.getInputStream();
                    res = readFromStream(inputStream);
requestConnection.disconnect();
return res;
                } catch (Exceptione) {
e.printStackTrace();
                }
            }
return"Error";
        }

/*After network request is made, call the actions function to perform post
layout updates
         * Since onPostExecute doesn't allows layout update commands*/
        @Override
protectedvoidonPostExecute(Stringresult) {
doActions(result);
```

```
        }

//Stream reader used to retrieve data from inputStreamReader
privateStringreadFromStream(InputStreaminputStream) throwsIOException {
StringBuilderoutput = newStringBuilder();
if (inputStream != null) {
InputStreamReaderinputStreamReader = newInputStreamReader(inputStream,
Charset.forName("UTF-8"));
BufferedReaderreader = newBufferedReader(inputStreamReader);
Stringline = reader.readLine();
while (line != null) {
output.append(line);
                    line = reader.readLine();
                }
            }
returnoutput.toString();
        }
    }
}
```

4. Login.java

```
package com.example.majorproject;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

publicclassLoginextendsAppCompatActivityimplementsView.OnClickListener {
//Necessary views variables to be used on screen
EditTexteditTextEmail;
EditTexteditTextPassword;
ButtonbuttonLogin;
ButtonbuttonBack;
FirebaseAuthfirebaseAuth;
```

```
LinearLayoutdetails;

/* Default method called by android os when a new view is created*/
    @Override
protectedvoidonCreate(BundlesavedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_login);
        firebaseAuth = FirebaseAuth.getInstance();

//Initialize all layouts
        buttonLogin = findViewById(R.id.buttonLogin);
        buttonBack = findViewById(R.id.buttonBack);
        editTextEmail = findViewById(R.id.editTextSignInEmail);
        editTextPassword = findViewById(R.id.editTextSignInPassword);
        details = findViewById(R.id.details);


//add listener to the buttons
buttonLogin.setOnClickListener(this);
buttonBack.setOnClickListener(this);

if (firebaseAuth.getCurrentUser() != null) {
showToast("User is already logged in");
finish();
IntentmainActivity = newIntent(Login.this, MainActivity.class);
startActivity(mainActivity);
        }
    }

/*Function to perform user login task and show necessary messages about
progress
    And start main activity after successful login
    * */
publicvoiduserLogin() {
Stringemail = editTextEmail.getText().toString();
Stringpassword = editTextPassword.getText().toString();
firebaseAuth.signInWithEmailAndPassword(email, password)
                .addOnCompleteListener(this,
newOnCompleteListener<AuthResult>() {
                    @Override
publicvoidonComplete(@NonNullTask<AuthResult>task) {
if (task.isSuccessful()) {
//successfully registered toast message
showToast("Signed in");

//finish this activity and call new activity which is MainActivity of APP
finish();
IntentmainActivity = newIntent(Login.this, MainActivity.class);
startActivity(mainActivity);
```

```
                        } else {
//Show invalid message if UserID/Password doesn't match
showToast("Invalid information! Try again");
                        }
                    }
                });
    }

/*For each view clicked perform necessary actions associated to that*/
    @Override
publicvoidonClick(Viewview) {
if (view == buttonLogin) {
userLogin();
        }
if (view == buttonBack) {
//if back button is pressed then clear this activity from memory and start
registration screen
finish();
IntentsignupActivity = newIntent(Login.this, Signup.class);
startActivity(signupActivity);
        }
    }

/*Method to show short length toast messages*/
publicvoidshowToast(StringtoastMessage) {
Toast.makeText(Login.this, toastMessage,
Toast.LENGTH_LONG).show();
    }
}
```

5. MainActivity.java

```
package com.example.majorproject;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.MenuItem;
import android.view.View;
import android.widget.TextView;

import androidx.appcompat.app.ActionBarDrawerToggle;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.core.view.GravityCompat;
```

```
import androidx.drawerlayout.widget.DrawerLayout;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentTransaction;

import com.google.android.material.navigation.NavigationView;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

publicclassMainActivityextendsAppCompatActivity
implementsNavigationView.OnNavigationItemSelectedListener,
FragmentManager.OnBackStackChangedListener {
//TextView to show user Email Address
TextViewuserEmailDisplay;

//Firebase Authentication object to be used to check or logout current user
FirebaseAuthfirebaseAuth;

//Each fragments which can be accessed form navigation bar
FragmentexploreFragment;
FragmentsentimentFragment;
FragmentanalyzerFragment;
FragmentratingFragment;
FragmentrecommendationFragment;

/* Default method called by android os when a new view is created*/
    @Override
protectedvoidonCreate(BundlesavedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

//Initialize each layout variables
Toolbartoolbar = findViewById(R.id.toolbar);
setSupportActionBar(toolbar);
DrawerLayoutdrawer = findViewById(R.id.drawer_layout);
NavigationViewnavigationView = findViewById(R.id.nav_view);
ActionBarDrawerToggletoggle = newActionBarDrawerToggle(
this, drawer, toolbar, R.string.navigation_drawer_open,
R.string.navigation_drawer_close);
drawer.addDrawerListener(toggle);
toggle.syncState();
navigationView.setNavigationItemSelectedListener(this);

ViewheaderView = navigationView.getHeaderView(0);
        userEmailDisplay =
headerView.findViewById(R.id.textViewUserEmailDisplay);
```

```
//Check if user is logged in, if not then kill the activity and redirect to
registration screen
        firebaseAuth = FirebaseAuth.getInstance();
if (firebaseAuth.getCurrentUser() == null) {
finish();
IntentsignupActivity = newIntent(MainActivity.this, Signup.class);
startActivity(signupActivity);
        } else {
FirebaseUseruser = firebaseAuth.getCurrentUser();
userEmailDisplay.setText(user.getEmail());
        }

//if there is not saved fragment then begin with Movie Rating search fragment
if (savedInstanceState == null) {
getSupportFragmentManager().beginTransaction().replace(R.id.fragment_container
,
newMovieRatingSearchFragment()).commit();
navigationView.setCheckedItem(R.id.nav_movie_rating_search);
        }
getSupportFragmentManager().addOnBackStackChangedListener(this);

//Initialize each Fragment objects in advance to be called whenever navigation
bar is used to switch between them
        exploreFragment = newExploreFragment();
        sentimentFragment = newSentimentPredictionFragment();
        analyzerFragment = newTwitterAnalyzerFragment();
        ratingFragment = newMovieRatingSearchFragment();
        recommendationFragment = newRecommendationFragment();

    }

/*When back button is pressed perform drawer closing task if it is open
     * Otherwise default back button task*/
    @Override
publicvoidonBackPressed() {
DrawerLayoutdrawer = findViewById(R.id.drawer_layout);
if (drawer.isDrawerOpen(GravityCompat.START)) {
drawer.closeDrawer(GravityCompat.START);
        } else {
super.onBackPressed();
        }
    }

/*Method the replace the current fragment with another one form list,
    if another fragment is not already there then create one
    * */
privatevoidreplaceFragment(Stringtag) {
FragmentManagerfragmentManager = getSupportFragmentManager();
```

```
FragmentTransactiontransaction = fragmentManager.beginTransaction();
FragmentcurrentFragment =
fragmentManager.findFragmentById(R.id.fragment_container);
FragmentnextFragment = fragmentManager.findFragmentByTag(tag);

Log.d("Fragment", "f detached: " + currentFragment.toString());
transaction.detach(currentFragment);

if (nextFragment == null) {
            nextFragment = createFragment(tag);
transaction.add(R.id.fragment_container, nextFragment, tag);
        } else {
Log.d("Fragment", "f attach: " + nextFragment.toString());
transaction.attach(nextFragment);
        }
transaction.commit();
    }

/*Method to create fragment with its associated tag*/
privateFragmentcreateFragment(Stringtag) {
Fragmentresult = null;
switch (tag) {
case"explore":
                result = newExploreFragment();
break;
case"sentiment":
                result = newSentimentPredictionFragment();
break;
case"analyzer":
                result = newTwitterAnalyzerFragment();
break;
case"rating":
                result = newMovieRatingSearchFragment();
break;
case"recommendation":
                result = newRecommendationFragment();
break;

        }
return result;
    }

    @Override
publicbooleanonNavigationItemSelected(MenuItemitem) {
// Handle navigation view item clicks here.
intid = item.getItemId();
switch (id) {
```

```
//if exit button is pressed then cross check the action via showing alert
dialog
caseR.id.nav_exit: {
new AlertDialog.Builder(this)
                        .setTitle("Please conform")
                        .setMessage("Do you really want to exit?")
                        .setIcon(android.R.drawable.ic_dialog_alert)
                        .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
publicvoidonClick(DialogInterfacedialog, intwhichButton) {
//perform exit
Intentintent = newIntent(Intent.ACTION_MAIN);
intent.addCategory(Intent.CATEGORY_HOME);
intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
startActivity(intent);
                            }
                        })
                        .setNegativeButton("No", null).show();
            }
break;
caseR.id.nav_logout: {
//if logout button is pressed then cross check the action via showing alert
dialog
new AlertDialog.Builder(this)
                        .setTitle("Please conform")
                        .setMessage("Do you really want to logout?")
                        .setIcon(android.R.drawable.ic_dialog_alert)
                        .setPositiveButton("Yes", new
DialogInterface.OnClickListener() {

publicvoidonClick(DialogInterfacedialog, intwhichButton) {
//perform logout activity here
firebaseAuth.signOut();
finish();
IntentsignupScreen = newIntent(MainActivity.this, Signup.class);
startActivity(signupScreen);
                            }
                        })
                        .setNegativeButton("No", null).show();
            }
break;
caseR.id.nav_sentiment_prediction: {
//Replace current fragment with Sentiment Analysis on Text Fragment
replaceFragment("sentiment");
            }
break;
caseR.id.nav_movie_rating_search: {
//Replace current fragment with Movie Rating Search Fragment
```

```
replaceFragment("rating");
            }
break;
caseR.id.nav_twitter_analyzer: {
//Replace current fragment with Twitter Analyzer Fragment
replaceFragment("analyzer");
            }
break;
caseR.id.nav_explore: {
//Replace current fragment with Explore Fragment to show trending movies
replaceFragment("explore");
            }
break;
caseR.id.nav_recommendation: {
//Replace current fragment with Recommendation Fragment to show recommended
movies
replaceFragment("recommendation");
            }
break;
        }

DrawerLayoutdrawer = findViewById(R.id.drawer_layout);
drawer.closeDrawer(GravityCompat.START);
returntrue;
    }

/*On back button pressed form another child activity, show the fragment in
resumed state*/
    @Override
publicvoidonBackStackChanged() {
try {
Fragmentfragment =
getSupportFragmentManager().findFragmentById(R.id.fragment_container);
fragment.onResume();
        } catch (Exceptionex) {
ex.printStackTrace();
        }
    }

//Set Action bar title to the passed string
publicvoidsetActionBarTitle(Stringtitle) {
getSupportActionBar().setTitle(title);
    }
}
```

6. Movie.java

```
package com.example.majorproject;
```

```
import android.graphics.Bitmap;

/*Class to hold details about the movie, with method to retrieve each
attribute of movie
 * */
publicclassMovie {
Stringmovie_title;
Stringmovie_original_title;
Stringlanguage;
Stringoverview;
Stringrelease_date;
Stringposter_url;
Bitmapmovie_poster;

publicMovie(Stringmovie_title, Stringmovie_original_title, Stringlanguage,
Stringoverview, Stringrelease_date, Stringposter_url) {
this.movie_title = movie_title;
this.movie_original_title = movie_original_title;
this.language = language;
this.overview = overview;
this.release_date = release_date;
this.poster_url = poster_url;
    }

publicStringgetTitle() {
return movie_title;
    }

publicStringgetOriginalTitle() {
return movie_original_title;
    }

publicStringgetLanguage() {
return language;
    }

publicStringgetReleaseDate() {
return release_date;
    }

publicStringgetPosterUrl() {
return poster_url;
    }

publicStringgetOverview() {
return overview;
    }
```

```
publicBitmapgetPoster() {
return movie_poster;
    }

publicvoidsetPoster(Bitmapmovie_poster) {
this.movie_poster = movie_poster;
    }
}
```

7. MovieAdapter.java

```
package com.example.majorproject;

import android.app.Activity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

import java.util.ArrayList;

/*Inherited from ArrayAdapter to reduce memory overhead for layout which shows
a list.
Works by recycling views to create new one when scrolling
* */
publicclassMovieAdapterextendsArrayAdapter<Movie> {
publicMovieAdapter(Activitycontext, ArrayList<Movie>arr) {
super(context, 0, arr);
    }

    @Override
publicViewgetView(intposition, ViewconvertView, ViewGroupparent) {
//check if there is an existing list item view called convertView
ViewlistItemView = convertView;
if (listItemView == null) {
            listItemView =
LayoutInflater.from(getContext()).inflate(R.layout.movie_list, parent, false);
        }
MoviecurrentMovie = getItem(position);
TextViewmovieTitle = listItemView.findViewById(R.id.textViewMovieTitle);
ImageViewmoviePoster = listItemView.findViewById(R.id.imageViewPoster);
movieTitle.setText(currentMovie.getTitle());
moviePoster.setImageBitmap(currentMovie.getPoster());
return listItemView;
    }
}
```

8. MovieDetailsActivity.java

```
package com.example.majorproject;

import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.auth.FirebaseAuth;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.net.URL;
import java.net.URLEncoder;
import java.nio.charset.Charset;

import javax.net.ssl.HttpsURLConnection;
/*Activity similar to Explore Movie details, except it shows details of movie
with rating */

publicclassMovieDetailsActivityextendsAppCompatActivityimplementsView.OnClickL
istener {
BitmapmoviePoster;
ImageViewposterImageView;
TextViewmovieRating;
TextViewmovieTitleTextView;
CircleDisplaycd;
StringmovieTitle;
floatscore;
StringlikeURL;
StringdislikeURL;
```

```
StringuID;
Booleanflag = true;
FirebaseAuthfirebaseAuth;
ButtonbuttonLaunchGoogle;
ButtonbuttonLike, buttonDislike;

    @Override
protectedvoidonCreate(BundlesavedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_movie_details);
        likeURL = "https://major-project-final-246818.appspot.com/addLike/";
        dislikeURL = "https://major-project-final-
246818.appspot.com/addDislike/";
        buttonLaunchGoogle = findViewById(R.id.buttonLaunchInternet);
        buttonLike = findViewById(R.id.buttonLike);
        buttonDislike = findViewById(R.id.buttonDislike);


Bundleextras = getIntent().getExtras();
byte[] byteArray = extras.getByteArray("moviePoster");
        moviePoster = BitmapFactory.decodeByteArray(byteArray, 0,
byteArray.length);
        posterImageView = findViewById(R.id.image_view);
posterImageView.setImageBitmap(moviePoster);
        movieRating = findViewById(R.id.textViewMovieRating);

StringtempRating = extras.getString("movieRating");
movieRating.setText("Rating: " + tempRating);
        movieTitleTextView = findViewById(R.id.movieTitleTextView);
        movieTitle = extras.getString("movieTitle");
movieTitleTextView.setText(movieTitle);
        cd = findViewById(R.id.circleDisplay);
        score = Float.parseFloat((tempRating.split("/"))[0]);
showCircle();
posterImageView.setOnClickListener(this);
movieTitleTextView.setOnClickListener(this);
buttonLaunchGoogle.setOnClickListener(this);
buttonLike.setOnClickListener(this);
buttonDislike.setOnClickListener(this);
        firebaseAuth = FirebaseAuth.getInstance();
        uID = "user" + firebaseAuth.getCurrentUser().getUid();

String[] input = movieTitle.split(" ");
Stringaddress2 = "";
for (inti = 0; i <input.length - 1; i++) {
            address2 += input[i] + "+";
        }
        address2 += input[input.length - 1];
        likeURL = likeURL + uID + "+" + address2;
```

```
        dislikeURL = dislikeURL + uID + "+" + address2;
    }

voidshowCircle() {
cd.setAnimDuration(3000);
cd.setValueWidthPercent(55f);
cd.setTextSize(16f);
        score = score * 10;
cd.setColor(getResources().getColor(R.color.PrimaryPurple));
cd.setDrawText(true);
cd.setDrawInnerCircle(true);
cd.setFormatDigits(1);
cd.setTouchEnabled(false);
cd.setUnit("%");
cd.setStepSize(0.5f);
cd.showValue(score, 100f, true);
    }

    @Override
publicvoidonClick(Viewview) {
if (view == posterImageView) {
callGooogle();
        }
if (view == movieTitleTextView) {
callGooogle();
        }
if (view == buttonLaunchGoogle) {
StringescapedQuery = null;
try {
                escapedQuery = URLEncoder.encode(movieTitle, "UTF-8");
            } catch (UnsupportedEncodingExceptione) {
e.printStackTrace();
            }
Uriuri = Uri.parse("http://www.google.com/#q=" + escapedQuery);
Intentintent = newIntent(Intent.ACTION_VIEW, uri);
startActivity(intent);
        }
if (view == buttonLike) {
            flag = true;
buttonLike.setBackground(getResources().getDrawable(R.drawable.ic_button_like_
red));
buttonDislike.setBackground(getResources().getDrawable(R.drawable.ic_dislike))
;
MovieDetailsActivity.LikeDislikeAsyncTasktask = new
MovieDetailsActivity.LikeDislikeAsyncTask();
task.execute(likeURL);

        }
```

```
if (view == buttonDislike) {
            flag = false;
buttonLike.setBackground(getResources().getDrawable(R.drawable.ic_like));
buttonDislike.setBackground(getResources().getDrawable(R.drawable.ic_button_di
slike_red));
MovieDetailsActivity.LikeDislikeAsyncTasktask = new
MovieDetailsActivity.LikeDislikeAsyncTask();
task.execute(dislikeURL);
        }

    }

publicvoidcallGooogle() {
StringescapedQuery = null;
try {
            escapedQuery = URLEncoder.encode(movieTitle, "UTF-8");
        } catch (UnsupportedEncodingExceptione) {
e.printStackTrace();
        }
Uriuri = Uri.parse("http://www.google.com/#q=" + escapedQuery);
Intentintent = newIntent(Intent.ACTION_VIEW, uri);
startActivity(intent);
    }

publicbooleanisNetworkAvailable() {
try {
ConnectivityManagermConnectivityManager = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfomNetworkInfo = mConnectivityManager.getActiveNetworkInfo();
return mNetworkInfo != null;

        } catch (NullPointerExceptione) {
returnfalse;
        }
    }

publicvoiddoActions(Stringresult) {
if (result.equals("Error")) {
showToast("Server communication failure, please try again later");
        } else {
if (flag == true) {
showToast("Added to likes");
            } else {
showToast("Added to dislikes");
            }
        }
    }
```

```
publicvoidshowToast(StringtoastMessage) {
Toast.makeText(MovieDetailsActivity.this, toastMessage,
Toast.LENGTH_LONG).show();
    }

publicclassLikeDislikeAsyncTaskextendsAsyncTask<String, Void, String> {
        @Override
protectedvoidonPreExecute() {
super.onPreExecute();
        }

        @Override
protectedStringdoInBackground(String... strings) {
if (isNetworkAvailable()) {
Stringres = "str";
try {
URLurl = newURL(strings[0]);
InputStreaminputStream;
HttpsURLConnectionrequestConnection = (HttpsURLConnection)
url.openConnection();
requestConnection.setReadTimeout(200000);
requestConnection.setConnectTimeout(100000);
requestConnection.setRequestMethod("GET");
requestConnection.connect();
//if (requestConnection.getResponseCode() == 200) {
                    inputStream = requestConnection.getInputStream();
                    res = readFromStream(inputStream);
//}
requestConnection.disconnect();
Log.v("Result:", "\n\t\t\t====================" + res);
return res;
                } catch (Exceptione) {
e.printStackTrace();
                }
            }
return"Error";
        }

        @Override
protectedvoidonPostExecute(Stringresult) {
doActions(result);
///progressBar.setVisibility(View.INVISIBLE);
//details.setVisibility(View.VISIBLE);
        }

privateStringreadFromStream(InputStreaminputStream) throwsIOException {
StringBuilderoutput = newStringBuilder();
if (inputStream != null) {
```

```
InputStreamReaderinputStreamReader = newInputStreamReader(inputStream,
Charset.forName("UTF-8"));
BufferedReaderreader = newBufferedReader(inputStreamReader);
Stringline = reader.readLine();
while (line != null) {
output.append(line);
                    line = reader.readLine();
                }
            }
returnoutput.toString();
        }
    }
}
```

9. MovieRatingSearchFragment.java

```
package com.example.majorproject;

import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.ProgressBar;

import androidx.appcompat.app.AlertDialog;
import androidx.fragment.app.Fragment;

import com.github.ybq.android.spinkit.style.DoubleBounce;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.InputStream;
```

```
import java.io.InputStreamReader;
import java.net.URL;
import java.nio.charset.Charset;

import javax.net.ssl.HttpsURLConnection;

/*Fragment for search the movie details screen
 * */
publicclassMovieRatingSearchFragmentextendsFragmentimplementsView.OnClickListe
ner {
//All Layout and value container variables
BitmapmoviePoster;
ButtonbuttonMovieSearch;
StringAPIKEY = ""; //API KEY Generated from OMDBAPI Website
Stringaddress = "https://www.omdbapi.com/?apikey=" + APIKEY + "&t=";
StringmovieName = "";
ProgressBarprogressBar;
LinearLayoutdetails;
StringmovieRating;
StringmovieTitle;
EditTextinputBox;
StringposterUrl = "";

publicViewonCreateView(LayoutInflaterinflater, ViewGroupcontainer,
BundlesavedInstanceState) {
// Inflate the layout for this fragment
Viewview = inflater.inflate(R.layout.fragment_movie_rating_search, container,
false);

//Initialize all variables with view ids from layout file
        details = view.findViewById(R.id.details);
        progressBar = view.findViewById(R.id.spin_kit);
DoubleBouncemyProgressBar = newDoubleBounce();
progressBar.setIndeterminateDrawable(myProgressBar);
        inputBox = view.findViewById(R.id.editTextMovieName);
        buttonMovieSearch = view.findViewById(R.id.buttonMovieSearch);
buttonMovieSearch.setOnClickListener(this);
return view;
    }

/*If fragment is resumed from a paused state call super and set the title to
current "Search movie rating"
     * */
publicvoidonResume() {
super.onResume();
        ((MainActivity) getActivity())
                .setActionBarTitle("Search Movie Rating");
    }
```

```
/*Show alert dialog function, to be used by function when no internet
connection is found or any other errors*/

publicvoidshowAlertDialog(StringTITLE, StringMESSAGE) {
new AlertDialog.Builder(getActivity())
                .setTitle(TITLE)
                .setMessage(MESSAGE)
                .setNegativeButton("OK", null)
                .setIcon(android.R.drawable.ic_dialog_alert)
                .show();
    }

//Method to call poster downloaded AsyncTask
publicvoidstartMoviePosterDownloader() {
DownloadImageFromInternetimageDownload = newDownloadImageFromInternet();
imageDownload.execute(posterUrl);
    }

/*Call necessary functions for each view clicked on screen*/
    @Override
publicvoidonClick(Viewview) {
if (view == buttonMovieSearch) {
            movieName = inputBox.getText().toString();
String[] input = inputBox.getText().toString().split(" ");
Stringaddress2 = "";
for (inti = 0; i <input.length - 1; i++) {
                address2 += input[i] + "+";
            }
            address2 += input[input.length - 1];
Stringurl = address + address2;

IMDBAsyncTasktask = newIMDBAsyncTask();
task.execute(url);
        }
    }

/*Method to check if network connection is available*/
publicbooleanisNetworkAvailable() {
try {
ConnectivityManagermConnectivityManager = (ConnectivityManager)
getActivity().getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfomNetworkInfo = mConnectivityManager.getActiveNetworkInfo();
return mNetworkInfo != null;

        } catch (NullPointerExceptione) {
returnfalse;
        }
```

```
    }

/* Processing result to get necessary details form JSON response*/
publicvoiddoActions(Stringresult) {
JSONObjectroot = null;
if (result.equals("Error")) {
showAlertDialog("Connection Error", "No data connection found!");
progressBar.setVisibility(View.INVISIBLE);
details.setVisibility(View.VISIBLE);
        } else {
try {
                root = newJSONObject(result);
if (root.getString("Response").equals("True")) {
                    movieTitle = root.getString("Title");
                    posterUrl = root.getString("Poster");
JSONArrayratingsArray = root.getJSONArray("Ratings");
JSONObjectratings = ratingsArray.getJSONObject(0);
                    movieRating = ratings.getString("Value");
                    posterUrl = root.getString("Poster");
startMoviePosterDownloader();
                } else {
showAlertDialog("Error", root.getString("Error"));
progressBar.setVisibility(View.INVISIBLE);
details.setVisibility(View.VISIBLE);
                }
            } catch (JSONExceptione) {
e.printStackTrace();
            }
        }
    }

/*Method to call Details Activity to show details about searched movie*/
publicvoidsendMoviePoster(Bitmapresult) {
ByteArrayOutputStreamstream = newByteArrayOutputStream();
result.compress(Bitmap.CompressFormat.PNG, 100, stream);
byte[] byteArray = stream.toByteArray();
IntentviewPoster = newIntent(getActivity(), MovieDetailsActivity.class);
viewPoster.putExtra("moviePoster", byteArray);
viewPoster.putExtra("movieRating", movieRating);
viewPoster.putExtra("movieTitle", movieTitle);
progressBar.setVisibility(View.INVISIBLE);
details.setVisibility(View.VISIBLE);
startActivity(viewPoster);
    }

/*Inherited AsyncTask to perform background network request*/
publicclassIMDBAsyncTaskextendsAsyncTask<String, Void, String> {
        @Override
```

```
protectedvoidonPreExecute() {
super.onPreExecute();
progressBar.setVisibility(View.VISIBLE);
details.setVisibility(View.INVISIBLE);
        }

        @Override
protectedStringdoInBackground(String... strings) {
if (isNetworkAvailable()) {
Stringres = "str";
try {
URLurl = newURL(strings[0]);
InputStreaminputStream;
HttpsURLConnectionrequestConnection = (HttpsURLConnection)
url.openConnection();
requestConnection.setReadTimeout(200000);
requestConnection.setConnectTimeout(100000);
requestConnection.setRequestMethod("GET");
requestConnection.connect();
                    inputStream = requestConnection.getInputStream();
                    res = readFromStream(inputStream);
requestConnection.disconnect();
return res;
                } catch (Exceptione) {
e.printStackTrace();
                }
            }
return"Error";
        }

        @Override
protectedvoidonPostExecute(Stringresult) {
doActions(result);
        }

privateStringreadFromStream(InputStreaminputStream) throwsIOException {
StringBuilderoutput = newStringBuilder();
if (inputStream != null) {
InputStreamReaderinputStreamReader = newInputStreamReader(inputStream,
Charset.forName("UTF-8"));
BufferedReaderreader = newBufferedReader(inputStreamReader);
Stringline = reader.readLine();
while (line != null) {
output.append(line);
                    line = reader.readLine();
                }
            }
returnoutput.toString();
```

```
        }
    }

/*
     * Movie poster downloader AsyncTask to perform request in background
     */
privateclassDownloadImageFromInternetextendsAsyncTask<String, Void, Bitmap> {
protectedBitmapdoInBackground(String... urls) {
StringimageURL = urls[0];
Bitmapbimage = null;
try {
InputStreamin = new java.net.URL(imageURL).openStream();
                bimage = BitmapFactory.decodeStream(in);

            } catch (Exceptione) {
Log.e("Error Message", e.getMessage());
e.printStackTrace();
            }
return bimage;
        }

protectedvoidonPostExecute(Bitmapresult) {
sendMoviePoster(result);
        }
    }
}
```

10. RecommendationFragment.java

```
package com.example.majorproject;

import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.GridView;
import android.widget.LinearLayout;
import android.widget.ProgressBar;
```

```
import android.widget.TextView;

import androidx.appcompat.app.AlertDialog;
import androidx.fragment.app.Fragment;

import com.github.ybq.android.spinkit.style.DoubleBounce;
import com.google.firebase.auth.FirebaseAuth;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.nio.charset.Charset;
import java.util.ArrayList;
import java.util.List;

import javax.net.ssl.HttpsURLConnection;

publicclassRecommendationFragmentextendsFragment {

//Server API method address to get the JSON response about recommended Movies
StringrecommendedMoviesURL = "https://major-project-final-
246818.appspot.com/getRecommendation/";

//Holders for necessary views and userID
StringuID;
FirebaseAuthfirebaseAuth;
ArrayList<Movie>MoviesList;
MovieAdaptermAdapter;
GridViewmovieGridView;
ArrayList<Bitmap>posterList;
LinearLayoutrecommendationFragmentView;
TextViewrecommendationFragmentWarning;
ProgressBarprogressBar;

publicViewonCreateView(LayoutInflaterinflater, ViewGroupcontainer,
BundlesavedInstanceState) {
// Inflate the layout for this fragment
Viewview = inflater.inflate(R.layout.fragment_recommendation, container,
false);

//Initialize all variables
```

```
        movieGridView = view.findViewById(R.id.listViewRecommendedMovies);
        recommendationFragmentView =
view.findViewById(R.id.linearLayoutRecommendationFragment);
        recommendationFragmentWarning =
view.findViewById(R.id.textViewExploreStatusWarning);
        firebaseAuth = FirebaseAuth.getInstance();
        uID = "user" + firebaseAuth.getCurrentUser().getUid();

        MoviesList = newArrayList<>();
        posterList = newArrayList<>();
        progressBar = view.findViewById(R.id.spin_kit);
DoubleBouncemyProgressBar = newDoubleBounce();
progressBar.setIndeterminateDrawable(myProgressBar);

recommendationFragmentWarning.setVisibility(View.GONE);
RecommendationFragment.RecommendationAsyncTasktask = new
RecommendationFragment.RecommendationAsyncTask();
task.execute(recommendedMoviesURL + uID);

movieGridView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
publicvoidonItemClick(AdapterView<?>adapterView, Viewview, inti, longl) {
MoviecurrentMovie = mAdapter.getItem(i);
IntentmovieDetailScreen = newIntent(getActivity(), ExploreMovieDetails.class);
BitmaptempPoster = currentMovie.getPoster();
ByteArrayOutputStreamstream = newByteArrayOutputStream();
tempPoster.compress(Bitmap.CompressFormat.PNG, 100, stream);
byte[] byteArray = stream.toByteArray();
movieDetailScreen.putExtra("moviePoster", byteArray);
movieDetailScreen.putExtra("movieTitle", currentMovie.getTitle());
movieDetailScreen.putExtra("movieOriginalTitle",
currentMovie.getOriginalTitle());
movieDetailScreen.putExtra("string_overview", currentMovie.getOverview());
movieDetailScreen.putExtra("movieReleaseDate", currentMovie.getReleaseDate());
startActivity(movieDetailScreen);
            }
        });
return view;
    }

/*If fragment is resumed from a paused state call super and set the title to
current "Recommended Movies"
     * */
publicvoidonResume() {
super.onResume();
        ((MainActivity) getActivity())
                .setActionBarTitle("Recommended Movies");
    }
```

```
/*Method to check if data connection is available*/
publicbooleanisNetworkAvailable() {
try {
ConnectivityManagermConnectivityManager = (ConnectivityManager)
getActivity().getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfomNetworkInfo = mConnectivityManager.getActiveNetworkInfo();
return mNetworkInfo != null;

        } catch (NullPointerExceptione) {
returnfalse;
        }
    }

/*Method to show alert dialog for errors or success messages*/
publicvoidshowAlertDialog(StringTITLE, StringMESSAGE) {
new AlertDialog.Builder(getActivity())
                .setTitle(TITLE)
                .setMessage(MESSAGE)
                .setNegativeButton("OK", null)
                .setIcon(android.R.drawable.ic_dialog_alert)
                .show();
    }

/*Method to perform information retrieval from JSON response*/
publicvoiddoActions(Stringresult) {
JSONObjectroot = null;
if (result.equals("Error")) {
showAlertDialog("Connection Error", "No data connection found!");
        } else {
try {
                root = newJSONObject(result);
//If status is False then there was error on server, show the received message
if (root.getString("status").equals("False")) {
recommendationFragmentWarning.setText(root.getString("result"));
recommendationFragmentWarning.setVisibility(View.VISIBLE);
recommendationFragmentWarning.setVisibility(View.VISIBLE);
progressBar.setVisibility(View.GONE);
                }
//if no error was found then perform retrieve of movie details from JSON
response
else {
JSONArrayresultsArray = root.getJSONArray("result");
Stringmovie_title;
Stringmovie_original_title;
Stringlanguage;
Stringoverview;
Stringrelease_date;
```

```
Stringposter_url;
Bitmapmovie_poster;
for (inti = 0; i <resultsArray.length(); i++) {
JSONObjectdetails = resultsArray.getJSONObject(i);
                        movie_title = details.getString("title");
                        movie_original_title = details.getString("title");
                        language = "English";
                        overview = details.getString("overview");
                        release_date = details.getString("release");
                        poster_url = details.getString("poster");
MoviesList.add(newMovie(movie_title, movie_original_title, language, overview,
release_date, poster_url));
                    }
                }
            } catch (JSONExceptione) {
e.printStackTrace();
            }
        }
performPosterRetrival();
    }


publicvoidperformPosterRetrival() {
//get all poster urls into one single list to download those in a single batch
if (MoviesList.size() != 0) {
String[] posterUrls = newString[MoviesList.size()];
for (inti = 0; i <MoviesList.size(); i++) {
                posterUrls[i] = (MoviesList.get(i).getPosterUrl());
            }
//After movie details have been collected, call function to download the
posters
RecommendationFragment.DownloadImageFromInternetimageDownload = new
RecommendationFragment.DownloadImageFromInternet();
imageDownload.execute(posterUrls);
        }

    }

publicvoidsendMoviePoster(List<Bitmap>result) {
for (inti = 0; i <result.size(); i++) {
Bitmapb = result.get(i);
            b = Bitmap.createScaledBitmap(b, (b.getWidth() / 3),
(b.getHeight() / 3), false);
MoviesList.get(i).setPoster(b);
        }
        mAdapter = newMovieAdapter(getActivity(), MoviesList);
movieGridView.setAdapter(mAdapter);
progressBar.setVisibility(View.GONE);
recommendationFragmentView.setVisibility(View.VISIBLE);
```

```
    }

publicclassRecommendationAsyncTaskextendsAsyncTask<String, Void, String> {
        @Override
protectedvoidonPreExecute() {
super.onPreExecute();
progressBar.setVisibility(View.VISIBLE);
        }

        @Override
protectedStringdoInBackground(String... strings) {
if (isNetworkAvailable()) {
Stringres = "str";
try {
URLurl = newURL(strings[0]);
Log.v("IMDB URL:", "\n\t\t\t====================" + strings[0]);
InputStreaminputStream;
HttpsURLConnectionrequestConnection = (HttpsURLConnection)
url.openConnection();
requestConnection.setReadTimeout(200000);
requestConnection.setConnectTimeout(100000);
requestConnection.setRequestMethod("GET");
requestConnection.connect();
                    inputStream = requestConnection.getInputStream();
                    res = readFromStream(inputStream);
requestConnection.disconnect();
return res;
                } catch (Exceptione) {
Log.v("Stack Track SSSS:", "\n\t\t\t====================" +
e.getStackTrace());
e.printStackTrace();
                }
            }
return"Error";
        }

        @Override
protectedvoidonPostExecute(Stringresult) {
doActions(result);
        }

privateStringreadFromStream(InputStreaminputStream) throwsIOException {
StringBuilderoutput = newStringBuilder();
if (inputStream != null) {
InputStreamReaderinputStreamReader = newInputStreamReader(inputStream,
Charset.forName("UTF-8"));
BufferedReaderreader = newBufferedReader(inputStreamReader);
Stringline = reader.readLine();
```

```
while (line != null) {
output.append(line);
                    line = reader.readLine();
                }
            }
returnoutput.toString();
        }
    }

/*AsyncTask to download posters from urls passed as a list in background*/
privateclassDownloadImageFromInternetextendsAsyncTask<String, Void,
List<Bitmap>> {
protectedList<Bitmap>doInBackground(String... urls) {
List<Bitmap>bitmaps = newArrayList<Bitmap>();
//String imageURL = urls[0];
Bitmapbimage = null;
for (Stringurl: urls) {
try {
InputStreamin = new java.net.URL(url).openStream();
                    bimage = BitmapFactory.decodeStream(in);

                } catch (Exceptione) {
Log.e("Error Message", e.getMessage());
e.printStackTrace();
                }
bitmaps.add(bimage);
            }
return bitmaps;
        }
protectedvoidonPostExecute(List<Bitmap>result) {
sendMoviePoster(result);
        }
    }
}
```

11. SentimentPrediction.java

```
package com.example.majorproject;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import org.json.JSONException;
import org.json.JSONObject;
```

```
/*Sentiment Prediction activity to call the Server on typed text*/
publicclassSentimentPredictionextendsAppCompatActivity {
Stringsentiment_response;
TextViewresult;
CircleDisplaycd;
floatscore;

/*By default method called by Android when new view gets created*/
    @Override
protectedvoidonCreate(BundlesavedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_sentiment_prediction);
        cd = findViewById(R.id.circleDisplay);
Intenti = getIntent();

//Parse the response received from the parent activity(Sentiment Prediction
Fragment)
        sentiment_response = i.getStringExtra("response");
        result = findViewById(R.id.textViewResult);

//call method to show result
showResult();
    }

/*Show the prediction result with a Circular animation on a scale of 1-100*/
voidshowResult() {
try {
JSONObjectroot = newJSONObject(sentiment_response);
Strings = null;
            s = root.getString("result ");
Log.v("Result SSSS:", "\n\t\t\t====================" + s);
String[] temp = s.split(" ");
            score = Float.parseFloat(temp[0]);
result.setText(s);
showCircle();
        } catch (JSONExceptione) {
e.printStackTrace();
        }
    }

voidshowCircle() {
cd.setAnimDuration(3000);
cd.setValueWidthPercent(55f);
cd.setTextSize(36f);
        score = score * 100;
cd.setColor(getResources().getColor(R.color.PrimaryPurple));
cd.setDrawText(true);
```

```
cd.setDrawInnerCircle(true);
cd.setFormatDigits(1);
cd.setTouchEnabled(false);
cd.setUnit("%");
cd.setStepSize(0.5f);
cd.showValue(score, 100f, true);
    }
}
```

12. SentimentPredictionFragment.java

```
package com.example.majorproject;

import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.ProgressBar;

import androidx.appcompat.app.AlertDialog;
import androidx.fragment.app.Fragment;

import com.github.ybq.android.spinkit.style.DoubleBounce;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.nio.charset.Charset;

import javax.net.ssl.HttpsURLConnection;

publicclassSentimentPredictionFragmentextendsFragmentimplementsView.OnClickLis
tener {
Stringaddress;
EditTextinputBox;
ProgressBarprogressBar;
LinearLayoutdetails;
```

```
ButtonbuttonSumbitQueryText;

publicViewonCreateView(LayoutInflaterinflater, ViewGroupcontainer,
BundlesavedInstanceState) {
// Inflate the layout for this fragment
Viewview = inflater.inflate(R.layout.fragment_predict_sentiment, container,
false);

        progressBar = view.findViewById(R.id.spin_kit);
        inputBox = view.findViewById(R.id.editTextQueryBox);
        details = view.findViewById(R.id.details);
DoubleBouncemyProgressBar = newDoubleBounce();
progressBar.setIndeterminateDrawable(myProgressBar);
        buttonSumbitQueryText = view.findViewById(R.id.buttonSentimentQuery);
buttonSumbitQueryText.setOnClickListener(this);
return view;
    }

/*If fragment is resumed from a paused state call super and set the title to
current "Sentiment Prediction"
     */
publicvoidonResume() {
super.onResume();
        ((MainActivity) getActivity())
                .setActionBarTitle("Sentiment Prediction");
    }


    @Override
publicvoidonClick(Viewview) {
//if query button is touched then call the API function to perform the request
if (view == buttonSumbitQueryText) {
String[] input = inputBox.getText().toString().split(" ");
Stringreview = "";
for (inti = 0; i <input.length - 1; i++) {
                review += input[i] + "+";
            }
            review += input[input.length - 1];

//append the query string at the end of address
            address = "https://major-project-final-
246818.appspot.com/sentiment/";

SentimentAsyncTasktask = newSentimentAsyncTask();
Stringurl = address;
            url = url + review;
task.execute(url);
        }
    }
```

```
/*Function to check if network is available*/
publicbooleanisNetworkAvailable() {
try {
ConnectivityManagermConnectivityManager = (ConnectivityManager)
getActivity().getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfomNetworkInfo = mConnectivityManager.getActiveNetworkInfo();
return mNetworkInfo != null;

        } catch (NullPointerExceptione) {
returnfalse;

        }
    }


publicvoiddoActions(Stringresult) {
if (!result.equals("Error")) {
//if there was no error then call the result activity to show prediction
result
try {
IntentresultActivity = newIntent(getActivity(), SentimentPrediction.class);
resultActivity.putExtra("response", result);
startActivity(resultActivity);
            } catch (Exceptione) {
e.printStackTrace();
            }
        } else {
//Else show the server error on alert dialog
showAlertDialog("Error", "Server Error!");
progressBar.setVisibility(View.INVISIBLE);
details.setVisibility(View.VISIBLE);
        }
    }

/*Function to show the alert dialog, to be used for showing error or
warnings*/
publicvoidshowAlertDialog(StringTITLE, StringMESSAGE) {
new AlertDialog.Builder(getActivity())
                .setTitle(TITLE)
                .setMessage(MESSAGE)
                .setNegativeButton("OK", null)
                .setIcon(android.R.drawable.ic_dialog_alert)
                .show();
    }

/*Inherited from AsyncTask to perform background network request without
affecting main OS thread*/
publicclassSentimentAsyncTaskextendsAsyncTask<String, Void, String> {
```

```
        @Override
protectedvoidonPreExecute() {
super.onPreExecute();
progressBar.setVisibility(View.VISIBLE);
details.setVisibility(View.INVISIBLE);
        }

        @Override
protectedStringdoInBackground(String... strings) {
if (isNetworkAvailable()) {
Stringres = "str";
try {
URLurl = newURL(strings[0]);
InputStreaminputStream;
HttpsURLConnectionrequestConnection = (HttpsURLConnection)
url.openConnection();
requestConnection.setReadTimeout(200000);
requestConnection.setConnectTimeout(100000);
requestConnection.setRequestMethod("GET");
requestConnection.connect();
                    inputStream = requestConnection.getInputStream();
                    res = readFromStream(inputStream);
requestConnection.disconnect();
Log.v("Result:", "\n\t\t\t====================" + res);
return res;
                } catch (Exceptione) {
e.printStackTrace();
                }
            }
return"Error";
        }

        @Override
protectedvoidonPostExecute(Stringresult) {
doActions(result);
progressBar.setVisibility(View.INVISIBLE);
details.setVisibility(View.VISIBLE);
        }

privateStringreadFromStream(InputStreaminputStream) throwsIOException {
StringBuilderoutput = newStringBuilder();
if (inputStream != null) {
InputStreamReaderinputStreamReader = newInputStreamReader(inputStream,
Charset.forName("UTF-8"));
BufferedReaderreader = newBufferedReader(inputStreamReader);
Stringline = reader.readLine();
while (line != null) {
output.append(line);
```

```
                    line = reader.readLine();
                }
            }
returnoutput.toString();
        }
    }
}
```

13. Signup.java

```
package com.example.majorproject;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

publicclassSignupextendsAppCompatActivityimplementsView.OnClickListener {
//Layout holders
EditTexteditTextEmail;
EditTexteditTextPassword;
ButtonbuttonSignup;
ButtonbuttonLogin;
LinearLayoutdetails;

//Firebase object to perform the registration
FirebaseAuthfirebaseAuth;

    @Override
protectedvoidonCreate(BundlesavedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_signup);
        firebaseAuth = FirebaseAuth.getInstance();

//initialize views
        editTextEmail = findViewById(R.id.editTextSignUpEmail);
        editTextPassword = findViewById(R.id.editTextSignUpPassword);
```

```
        buttonSignup = findViewById(R.id.buttonSignUp);
        buttonLogin = findViewById(R.id.buttonStartLoginScreen);
        details = findViewById(R.id.details);

//add onclick listener
buttonSignup.setOnClickListener(this);
buttonLogin.setOnClickListener(this);

//if user is already logged in then start main activity screen and kill this
present activity
if (firebaseAuth.getCurrentUser() != null) {
showToast("User is already logged in");
finish();
IntentmainActivity = newIntent(Signup.this, MainActivity.class);
startActivity(mainActivity);
        }

    }

/*Method to register user using Firebase authentication using email address
and password*/
voidregisterUser() {
Stringemail = editTextEmail.getText().toString();
Stringpassword = editTextPassword.getText().toString();
if (email.equals("") || password.equals("")) {
showToast("Invalid data");
        } else {
firebaseAuth.createUserWithEmailAndPassword(email, password)
                    .addOnCompleteListener(this,
newOnCompleteListener<AuthResult>() {
                        @Override
publicvoidonComplete(@NonNullTask<AuthResult>task) {
if (task.isSuccessful()) {
//successfully registered
showToast("Registered");
IntentmainActivity = newIntent(Signup.this, MainActivity.class);
startActivity(mainActivity);
                            } else {
//if registration failed then show error
showToast("Registration Failed");
                            }
                        }
                    });
        }
    }

/*Perform necessary actions based on views clicked on screen*/
    @Override
```

```
publicvoidonClick(Viewview) {
if (view == buttonSignup) {
registerUser();
        }
//if login button is pressed then call the login activity and kill current
activity
if (view == buttonLogin) {
IntentloginActivity = newIntent(Signup.this, Login.class);
startActivity(loginActivity);
        }
    }

/*Method to show short time toast messages*/
publicvoidshowToast(StringtoastMessage) {
Toast.makeText(Signup.this, toastMessage,
Toast.LENGTH_LONG).show();
    }
}
```

14. TwitterAnalyzerFragment.java

```
package com.example.majorproject;

import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.ProgressBar;

import androidx.appcompat.app.AlertDialog;
import androidx.fragment.app.Fragment;

import com.github.ybq.android.spinkit.style.DoubleBounce;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.nio.charset.Charset;
```

```
import javax.net.ssl.HttpsURLConnection;

publicclassTwitterAnalyzerFragmentextendsFragmentimplementsView.OnClickListene
r {
ButtonbuttonTwitterQuery;
Stringaddress;
ProgressBarprogressBar;
LinearLayoutdetails;
EditTextinputBox;

publicViewonCreateView(LayoutInflaterinflater, ViewGroupcontainer,
BundlesavedInstanceState) {
// Inflate the layout for this fragment
Viewview = inflater.inflate(R.layout.fragment_twitter_analyzer, container,
false);

        details = view.findViewById(R.id.details);
        progressBar = view.findViewById(R.id.spin_kit);
DoubleBouncemyProgressBar = newDoubleBounce();
progressBar.setIndeterminateDrawable(myProgressBar);

        inputBox = view.findViewById(R.id.editTextTwitterQuery);
        buttonTwitterQuery = view.findViewById(R.id.buttonTwitterAnalyzer);
buttonTwitterQuery.setOnClickListener(this);
return view;
    }

/*If fragment is resumed from a paused state call super and set the title to
current "Twitter Analyzer"
     */
publicvoidonResume() {
super.onResume();
        ((MainActivity) getActivity())
                .setActionBarTitle("Twitter Analyzer");
    }

/*For each click on views, call the necessary functions*/
    @Override
publicvoidonClick(Viewview) {
if (view == buttonTwitterQuery) {
String[] input = inputBox.getText().toString().split(" ");
Stringquery = "";
for (inti = 0; i <input.length - 1; i++) {
                query += input[i] + "+";
            }
            query += input[input.length - 1];
// API address + SearchString is final API function call address
```

```
            address = "https://major-project-final-
246818.appspot.com/analyze/";

//Call background thread to analyze thread
TwitterAnalyzerFragment.TwitterAsyncTasktask = new
TwitterAnalyzerFragment.TwitterAsyncTask();
Stringurl = address;
            url = url + query;
task.execute(url);
        }
    }

/*Method to check if network is available*/
publicbooleanisNetworkAvailable() {
try {
ConnectivityManagermConnectivityManager = (ConnectivityManager)
getActivity().getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfomNetworkInfo = mConnectivityManager.getActiveNetworkInfo();
return mNetworkInfo != null;

        } catch (NullPointerExceptione) {
returnfalse;

        }
    }

/*Perform retrieve of score from result if there is any else show error*/
publicvoiddoActions(Stringresult) {
if (!result.equals("Error")) {
try {
IntentresultActivity = newIntent(getActivity(), TwitterAnalyzerResult.class);
resultActivity.putExtra("response", result);
startActivity(resultActivity);
            } catch (Exceptione) {
e.printStackTrace();
            }
        } else {
showAlertDialog("Error", "Server Error!");
//hide progress bar and show the fragment again
progressBar.setVisibility(View.INVISIBLE);
details.setVisibility(View.VISIBLE);
        }
    }

/*Alert dialog to show the warning, errors and warning messages*/
publicvoidshowAlertDialog(StringTITLE, StringMESSAGE) {
new AlertDialog.Builder(getActivity())
                .setTitle(TITLE)
```

```
                .setMessage(MESSAGE)
                .setNegativeButton("OK", null)
                .setIcon(android.R.drawable.ic_dialog_alert)
                .show();
    }

/*Inherited AsyncTask class to peform network request in background without
affecting OS thread*/
publicclassTwitterAsyncTaskextendsAsyncTask<String, Void, String> {
        @Override
protectedvoidonPreExecute() {
super.onPreExecute();
progressBar.setVisibility(View.VISIBLE);
details.setVisibility(View.INVISIBLE);
        }

        @Override
protectedStringdoInBackground(String... strings) {
if (isNetworkAvailable()) {
Stringres = "str";
try {
URLurl = newURL(strings[0]);
InputStreaminputStream;
HttpsURLConnectionrequestConnection = (HttpsURLConnection)
url.openConnection();
//set timeouts for maximum wait before thread disconnects form server
requestConnection.setReadTimeout(2000000);
requestConnection.setConnectTimeout(1000000);
requestConnection.setRequestMethod("GET");
requestConnection.connect();
                    inputStream = requestConnection.getInputStream();
                    res = readFromStream(inputStream);
requestConnection.disconnect();
return res;
                } catch (Exceptione) {
e.printStackTrace();
                }
            }
return"Error";
        }

        @Override
protectedvoidonPostExecute(Stringresult) {
doActions(result);
progressBar.setVisibility(View.INVISIBLE);
details.setVisibility(View.VISIBLE);
        }
```

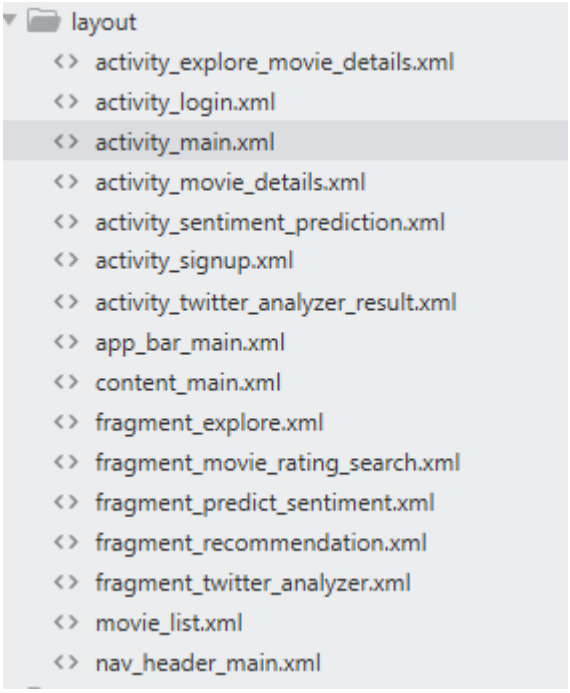
```
privateStringreadFromStream(InputStreaminputStream) throwsIOException {
StringBuilderoutput = newStringBuilder();
if (inputStream != null) {
InputStreamReaderinputStreamReader = newInputStreamReader(inputStream,
Charset.forName("UTF-8"));
BufferedReaderreader = newBufferedReader(inputStreamReader);
Stringline = reader.readLine();
while (line != null) {
output.append(line);
                    line = reader.readLine();
                }
            }
returnoutput.toString();
        }
    }
}
```

15. TwitterAnalyzerResult.java

```
package com.example.majorproject;

import android.content.Intent;
import android.os.Bundle;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import org.json.JSONException;
import org.json.JSONObject;

publicclassTwitterAnalyzerResultextendsAppCompatActivity {
//Holders for values and layouts to view result of twitter analysis
Stringtwitter_analyzer_response;
TextViewresult;
CircleDisplaycd;
floatscore;

    @Override
protectedvoidonCreate(BundlesavedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_twitter_analyzer_result);

        cd = findViewById(R.id.circleDisplay);
Intenti = getIntent();

//Get the result from parent activity(Twitter analyzer fragement)
        twitter_analyzer_response = i.getStringExtra("response");
        result = findViewById(R.id.textViewResultTwitterAnalyzer);
```

```
showResult();
    }

voidshowResult() {
try {
JSONObjectroot = newJSONObject(twitter_analyzer_response);
Strings = null;
            s = root.getString("result ");
            score = Float.parseFloat(s);
Stringtemp = "Positivity : " + s + " %";
result.setText(temp);
showCircle();
        } catch (JSONExceptione) {
e.printStackTrace();
        }
    }

voidshowCircle() {
cd.setAnimDuration(3000);
cd.setValueWidthPercent(55f);
cd.setTextSize(36f);
cd.setColor(getResources().getColor(R.color.PrimaryPurple));
cd.setDrawText(true);
cd.setDrawInnerCircle(true);
cd.setFormatDigits(1);
cd.setTouchEnabled(false);
cd.setUnit("%");
cd.setStepSize(0.5f);
cd.showValue(score, 100f, true);
    }
}
```

List of Layout Files:



1. activity_explore_movie_detalis.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollViewxmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical">

<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
tools:context=".ExploreMovieDetails">

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="horizontal">

<androidx.cardview.widget.CardView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:layout_margin="16dp"
android:elevation="16dp"
android:outlineAmbientShadowColor="@color/Black"
android:outlineSpotShadowColor="@color/Black"
android:translationZ="16dp"
```

```
app:cardCornerRadius="8dp">

<ImageView
android:id="@+id/imageViewPoster"
android:layout_width="fill_parent"
android:layout_height="150dp"
android:adjustViewBounds="true"/>
</androidx.cardview.widget.CardView>

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_weight="1"
android:orientation="vertical">

<TextView
android:id="@+id/textViewMovieTitle"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="8dp"
android:layout_weight="1"
android:padding="8dp"
android:textSize="16sp"/>

<TextView
android:id="@+id/textViewMovieOriginalTitle"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="8dp"
android:layout_weight="1"
android:padding="8dp"
android:textSize="16sp"/>

<TextView
android:id="@+id/textViewReleaseDate"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="8dp"
android:layout_weight="1"
android:padding="8dp"
android:textSize="16sp"/>
</LinearLayout>
</LinearLayout>

<TextView
```

```
android:id="@+id/textViewOverview"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="8dp"
android:padding="8dp"
android:textSize="16sp"/>

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:gravity="center"
android:orientation="horizontal">

<Button
android:id="@+id/buttonLike"
android:layout_width="53dp"
android:layout_height="53dp"
android:layout_margin="16dp"
android:background="@drawable/ic_like"
android:padding="8dp"/>

<Button
android:id="@+id/buttonDislike"
android:layout_width="50dp"
android:layout_height="50dp"
android:layout_margin="16dp"
android:background="@drawable/ic_dislike"
android:padding="2dp"/>

<Button
android:id="@+id/buttonLaunchInternet"
android:layout_width="wrap_content"
android:layout_height="50dp"
android:layout_gravity="center"
android:layout_margin="16dp"
android:background="@drawable/ripple_button"
android:gravity="center"
android:padding="8dp"
android:text="Find out more"/>
</LinearLayout>
</LinearLayout>
</ScrollView>
```

2. activity_login.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
```

```
android:layout_width="match_parent"
android:layout_height="match_parent"
android:gravity="center"
tools:context=".Login">

<LinearLayout
android:id="@+id/details"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="vertical">

<EditText
android:id="@+id/editTextSignInEmail"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="16dp"
android:hint="Enter the email ID"
android:textSize="24sp"/>

<EditText
android:id="@+id/editTextSignInPassword"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="16dp"
android:hint="Enter the password"
android:password="true"
android:textSize="24sp"/>

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="8dp"
android:orientation="horizontal">

<Button
android:id="@+id/buttonLogin"
style="@style/buttonStyle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_margin="8dp"
android:layout_weight="1"
android:text="Login"/>

<Button
android:id="@+id/buttonBack"
style="@style/buttonStyle"
```

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_margin="8dp"
android:layout_weight="1"
android:text="Back"/>
</LinearLayout>
</LinearLayout>
</RelativeLayout>
```

3. activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayoutxmlns:android="http://schemas.androi
d.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/drawer_layout"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:fitsSystemWindows="true"
tools:openDrawer="start">

<FrameLayout
android:id="@+id/fragment_container"
android:layout_width="match_parent"
android:layout_height="match_parent"/>

<include
layout="@layout/app_bar_main"
android:layout_width="match_parent"
android:layout_height="match_parent"/>

<com.google.android.material.navigation.NavigationView
android:id="@+id/nav_view"
android:layout_width="wrap_content"
android:layout_height="match_parent"
android:layout_gravity="start"
android:fitsSystemWindows="true"
app:headerLayout="@layout/nav_header_main"
app:menu="@menu/activity_main_drawer"/>


</androidx.drawerlayout.widget.DrawerLayout>
```

4. activity_movie_details.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"
```

```
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
tools:context=".MovieDetailsActivity">

<TextView
android:id="@+id/movieTitleTextView"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="60dp"
android:layout_margin="16dp"
android:textSize="24sp"/>

<androidx.cardview.widget.CardView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:layout_margin="16dp"
android:elevation="16dp"
android:outlineAmbientShadowColor="@color/Black"
android:outlineSpotShadowColor="@color/Black"
android:translationZ="16dp"
app:cardCornerRadius="8dp">

<ImageView
android:id="@+id/image_view"
android:layout_width="fill_parent"
android:layout_height="200dp"
android:adjustViewBounds="true"/>
</androidx.cardview.widget.CardView>

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:gravity="center"
android:orientation="horizontal">

<Button
android:id="@+id/buttonLike"
android:layout_width="53dp"
android:layout_height="53dp"
android:layout_margin="16dp"
android:background="@drawable/ic_like"
android:padding="8dp"/>

<Button
```

```
android:id="@+id/buttonDislike"
android:layout_width="50dp"
android:layout_height="50dp"
android:layout_margin="16dp"
android:background="@drawable/ic_dislike"
android:padding="2dp"/>

<Button
android:id="@+id/buttonLaunchInternet"
android:layout_width="wrap_content"
android:layout_height="50dp"
android:layout_gravity="center"
android:layout_margin="16dp"
android:background="@drawable/ripple_button"
android:gravity="center"
android:padding="8dp"
android:text="Find out more"/>
</LinearLayout>

<LinearLayout
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_gravity="center"
android:orientation="horizontal">

<TextView
android:id="@+id/textViewMovieRating"
style="@style/textBoxStyle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:layout_margin="16dp"
android:padding="16dp"
android:textSize="24sp"/>

<com.example.majorproject.CircleDisplay
android:id="@+id/circleDisplay"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_margin="16dp"
android:outlineAmbientShadowColor="@color/Black"
android:outlineSpotShadowColor="@color/Black"/>

</LinearLayout>

</LinearLayout>
```

5. activity_sentiment_prediction.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
tools:context=".SentimentPrediction">

<TextView
android:id="@+id/textViewResult"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:layout_margin="26dp"
android:textSize="24sp"/>

<com.example.majorproject.CircleDisplay
android:id="@+id/circleDisplay"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_margin="36dp"
android:outlineAmbientShadowColor="@color/Black"
android:outlineSpotShadowColor="@color/Black"/>
</LinearLayout>
```

6. activity_signup.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:gravity="center"
tools:context=".Signup">

<LinearLayout
android:id="@+id/details"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="vertical">

<EditText
android:id="@+id/editTextSignUpEmail"
style="@style/textBoxStyle"
```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="16dp"
android:hint="Enter the email ID"
android:textSize="24sp"/>

<EditText
android:id="@+id/editTextSignUpPassword"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="16dp"
android:hint="Enter the password"
android:password="true"
android:textSize="24sp"/>

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="8dp"
android:orientation="horizontal">

<Button
android:id="@+id/buttonSignUp"
style="@style/buttonStyle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_margin="8dp"
android:layout_weight="1"
android:text="Register"/>

<Button
android:id="@+id/buttonStartLoginScreen"
style="@style/buttonStyle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_margin="8dp"
android:layout_weight="1"
android:text="Already registered? Login Here"/>
</LinearLayout>

</LinearLayout>
</RelativeLayout>
```

7. activity_twitter_analyzer_result.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
tools:context=".TwitterAnalyzerResult">

<TextView
android:id="@+id/textViewResultTwitterAnalyzer"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:layout_margin="26dp"
android:textSize="24sp"/>

<com.example.majorproject.CircleDisplay
android:id="@+id/circleDisplay"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_margin="36dp"
android:outlineAmbientShadowColor="@color/Black"
android:outlineSpotShadowColor="@color/Black"/>
</LinearLayout>
```

8. app_bar_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.coordinatorlayout.widget.CoordinatorLayoutxmlns:android="http://sche
mas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity">

<com.google.android.material.appbar.AppBarLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:theme="@style/AppTheme.AppBarOverlay">

<androidx.appcompat.widget.Toolbar
android:id="@+id/toolbar"
android:layout_width="match_parent"
android:layout_height="?attr/actionBarSize"
android:background="?attr/colorPrimary"
app:popupTheme="@style/AppTheme.PopupOverlay"/>
```

```
</com.google.android.material.appbar.AppBarLayout>

<includelayout="@layout/content_main"/>

</androidx.coordinatorlayout.widget.CoordinatorLayout>
```

9. content_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayoutxmlns:android="http://schema
s.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
app:layout_behavior="@string/appbar_scrolling_view_behavior"
tools:context=".MainActivity"
tools:showIn="@layout/app_bar_main">

<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintLeft_toLeftOf="parent"
app:layout_constraintRight_toRightOf="parent"
app:layout_constraintTop_toTopOf="parent"/>

</androidx.constraintlayout.widget.ConstraintLayout>
```

10. fragment_explore.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.swiperefreshlayout.widget.SwipeRefreshLayoutxmlns:android="http://sc
hemas.android.com/apk/res/android"
android:id="@+id/pullToRefresh"
android:layout_width="match_parent"
android:layout_height="wrap_content">

<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:gravity="center">

<LinearLayout
android:id="@+id/linearLayoutExploreFragment"
android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
android:orientation="vertical"
android:visibility="gone">

<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:layout_margin="24dp"
android:gravity="center"
android:text="Hello World!"/>

<GridView
android:id="@+id/listViewMovies"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:columnWidth="100dp"
android:gravity="center"
android:numColumns="auto_fit"
android:stretchMode="columnWidth"></GridView>

</LinearLayout>

<com.github.ybq.android.spinkit.SpinKitViewxmlns:app="http://schemas.android.c
om/apk/res-auto"
android:id="@+id/spin_kit"
style="@style/SpinKitView.Large.Circle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerInParent="true"
android:layout_gravity="center"
android:elevation="16dp"
android:outlineAmbientShadowColor="@color/Black"
android:outlineSpotShadowColor="@color/Black"
android:visibility="invisible"
app:SpinKit_Color="@color/PrimaryPurple"/>

<TextView
android:id="@+id/textViewExploreStatusWarning"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="16dp"
android:gravity="center"
android:text="Loading! Please wait..."
android:textColor="@color/PrimaryPurple"
android:textSize="24sp"/>
</RelativeLayout>
</androidx.swiperefreshlayout.widget.SwipeRefreshLayout>
```

11. fragment_movie_rating_search.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent">

<LinearLayout
android:id="@+id/details"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:layout_centerVertical="true"
android:gravity="center"
android:orientation="vertical">

<EditText
android:id="@+id/editTextMovieName"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="60dp"
android:layout_gravity="center"
android:layout_margin="16dp"
android:hint="@string/enter_movie_name"

android:outlineProvider="bounds"/>

<Button
android:id="@+id/buttonMovieSearch"
style="@style/buttonStyle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_margin="16dp"
android:padding="16dp"
android:text="@string/search"/>
</LinearLayout>

<com.github.ybq.android.spinkit.SpinKitViewxmlns:app="http://schemas.android.c
om/apk/res-auto"
android:id="@+id/spin_kit"
style="@style/SpinKitView.Large.Circle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerInParent="true"
android:layout_gravity="center"
android:elevation="16dp"
android:outlineAmbientShadowColor="@color/Black"
android:outlineSpotShadowColor="@color/Black"
```

```
android:visibility="invisible"
app:SpinKit_Color="@color/PrimaryPurple"/>
</RelativeLayout>
```

12. fragment_predict_sentiment.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent">

<LinearLayout
android:id="@+id/details"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:gravity="center"
android:orientation="vertical">

<EditText
android:id="@+id/editTextQueryBox"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="60dp"
android:layout_gravity="center"
android:layout_margin="16dp"
android:hint="Enter review"
android:textSize="24sp"/>

<Button
android:id="@+id/buttonSentimentQuery"
style="@style/buttonStyle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:layout_margin="16dp"
android:onClick="buttonGetSentiment"
android:text="@string/Submit"/>
</LinearLayout>

<com.github.ybq.android.spinkit.SpinKitViewxmlns:app="http://schemas.android.c
om/apk/res-auto"
android:id="@+id/spin_kit"
style="@style/SpinKitView.Large.Circle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerInParent="true"
android:layout_gravity="center"
```

```
android:elevation="16dp"
android:outlineAmbientShadowColor="@color/Black"
android:outlineSpotShadowColor="@color/Black"
android:visibility="invisible"
app:SpinKit_Color="@color/PrimaryPurple"/>
</RelativeLayout>
```

13. fragment_recommendation.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent">

<LinearLayout
android:id="@+id/linearLayoutRecommendationFragment"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
android:visibility="gone">

<GridView
android:layout_marginTop="20dp"
android:id="@+id/listViewRecommendedMovies"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:columnWidth="100dp"
android:gravity="center"
android:numColumns="auto_fit"
android:stretchMode="columnWidth"></GridView>
</LinearLayout>

<com.github.ybq.android.spinkit.SpinKitViewxmlns:app="http://schemas.android.c
om/apk/res-auto"
android:id="@+id/spin_kit"
style="@style/SpinKitView.Large.Circle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerInParent="true"
android:layout_gravity="center"
android:elevation="16dp"
android:outlineAmbientShadowColor="@color/Black"
android:outlineSpotShadowColor="@color/Black"
android:visibility="invisible"
app:SpinKit_Color="@color/PrimaryPurple"/>

<TextView
```

```
android:id="@+id/textViewExploreStatusWarning"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_centerInParent="true"
android:layout_margin="16dp"
android:gravity="center"
android:text="Loading! Please wait..."
android:textColor="@color/PrimaryPurple"
android:textSize="24sp"/>

</RelativeLayout>
```

14. fragment_twitter_analyzer.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:background="@drawable/twitterbackgroung"
android:scaleType="centerCrop">

<LinearLayout
android:id="@+id/details"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:layout_centerVertical="true"
android:gravity="center"
android:orientation="vertical">

<EditText
android:id="@+id/editTextTwitterQuery"
style="@style/textBoxStyle"
android:layout_width="match_parent"
android:layout_height="60dp"
android:layout_gravity="center"
android:layout_margin="16dp"
android:hint="Enter search query"

android:outlineProvider="bounds"/>

<Button
android:id="@+id/buttonTwitterAnalyzer"
style="@style/buttonStyle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_margin="16dp"
```

```
android:padding="16dp"
android:text="@string/search"/>
</LinearLayout>

<com.github.ybq.android.spinkit.SpinKitViewxmlns:app="http://schemas.android.c
om/apk/res-auto"
android:id="@+id/spin_kit"
style="@style/SpinKitView.Large.Circle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerInParent="true"
android:layout_gravity="center"
android:elevation="16dp"
android:outlineAmbientShadowColor="@color/Black"
android:outlineSpotShadowColor="@color/Black"
android:visibility="invisible"
app:SpinKit_Color="@color/PrimaryPurple"/>
</RelativeLayout>
```

15. movie_list.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical">

<LinearLayout
android:layout_width="match_parent"
android:layout_height="250dp"
android:orientation="vertical">

<androidx.cardview.widget.CardView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:elevation="8dp"
android:outlineAmbientShadowColor="@color/Black"
android:outlineSpotShadowColor="@color/Black"
android:translationZ="16dp"
app:cardCornerRadius="8dp">

<ImageView
android:id="@+id/imageViewPoster"
android:layout_width="fill_parent"
android:layout_height="150dp"
```

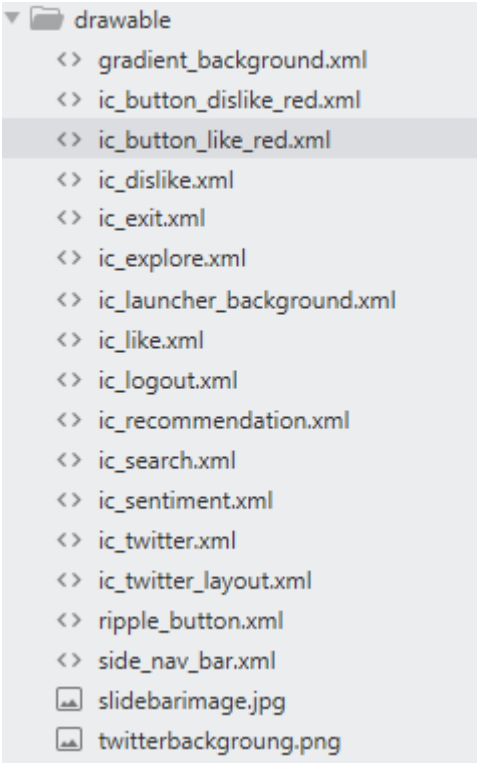
```
android:adjustViewBounds="true"/>
</androidx.cardview.widget.CardView>

<TextView
android:id="@+id/textViewMovieTitle"
android:layout_width="100dp"
android:layout_height="match_parent"
android:layout_gravity="center"
android:gravity="center"
android:text="Title"
android:textColor="@color/Black"
android:textSize="12sp"/>
</LinearLayout>
</LinearLayout>
```

16. nav_header_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="@dimen/nav_header_height"
android:background="@drawable/slidebarimage"
android:gravity="bottom"
android:orientation="vertical"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingBottom="@dimen/activity_vertical_margin"
android:theme="@style/ThemeOverlay.AppCompat.Dark">

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="horizontal">

<ImageView
android:id="@+id/imageView"
android:layout_width="80dp"
android:layout_height="80dp"
android:layout_weight="1"
android:contentDescription="@string/nav_header_desc"
android:paddingTop="@dimen/nav_header_vertical_spacing"
android:src="@mipmap/ic_logo_university"/>

<ImageView
android:id="@+id/imageViewAsutoshCollgeLogo"
android:layout_width="80dp"
```

```
android:layout_height="80dp"
android:layout_weight="1"
android:contentDescription="@string/nav_header_desc"
android:paddingTop="@dimen/nav_header_vertical_spacing"
android:src="@mipmap/ic_asutosh"/>
</LinearLayout>

<TextView
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:fontFamily="@font/almendra_sc"
android:gravity="center"
android:paddingTop="2dp"
android:text="Asutosh College Kolkata"
android:textAppearance="@style/TextAppearance.AppCompat.Body1"
android:textColor="@color/TextWhite"
android:textSize="20dp"
android:textStyle="bold"/>

<TextView
android:id="@+id/textViewUserEmailDisplay"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:gravity="center"
android:text="No user signed in"
android:textColor="@color/TextWhite"/>
</LinearLayout>
```

List of Drawable Resource Files:



1. ripple_button.xml

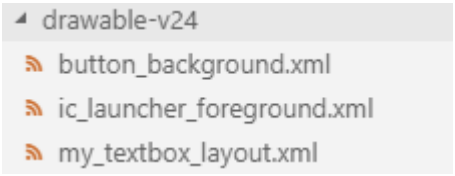
```
<?xml version="1.0" encoding="utf-8"?>
<ripplexmlns:android="http://schemas.android.com/apk/res/android"
android:color="#FFFFFF">
<itemandroid:id="@android:id/mask">
<shapeandroid:shape="rectangle">
<solidandroid:color="#FFFFFF"/>
<cornersandroid:radius="4dp"/>
</shape>
</item>

<itemandroid:id="@android:id/background">
<shapeandroid:shape="rectangle">
<gradient
android:angle="90"
android:endColor="@color/PrimaryPurple"
android:startColor="@color/PrimaryPurple"
android:type="linear"/>
<cornersandroid:radius="4dp"/>
</shape>
</item>
</ripple>
```

2. side_nav_bar.xml

```
<shapexmlns:android="http://schemas.android.com/apk/res/android"
```

```
android:shape="rectangle">
<gradient
android:angle="135"
android:centerColor="#009688"
android:endColor="#00695C"
android:startColor="#4DB6AC"
android:type="linear"/>

</shape>
```

List of drawable-v24 Files:



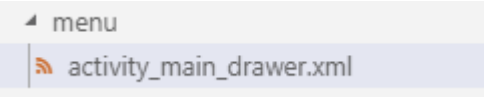
1. button_background.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ripplexmlns:android="http://schemas.android.com/apk/res/android"
android:color="#FFFFFF">
<itemandroid:id="@android:id/mask">
<shapeandroid:shape="rectangle">
<solidandroid:color="#FFFFFF"/>
<size
android:width="40dp"
android:height="40dp"/>
<cornersandroid:radius="6dp"/>
</shape>
</item>

<itemandroid:id="@android:id/background">
<shapeandroid:shape="rectangle">
<gradient
android:angle="90"
android:endColor="@color/PrimaryPurple"
android:startColor="@color/PrimaryPurple"
android:type="linear"/>
<size
android:width="40dp"
android:height="40dp"/>
<cornersandroid:radius="6dp"></corners>
</shape>
</item>
</ripple>
```

2. my_textbox_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shapexmlns:android="http://schemas.android.com/apk/res/android"
android:shape="rectangle">
<solidandroid:color="@color/TextWhite"/>
<stroke
android:width="1dp"
android:color="@color/PrimaryPurple"/>
<size
android:width="40dp"
android:height="40dp"/>
<cornersandroid:radius="10dp"></corners>
</shape>
```

List of menu items:

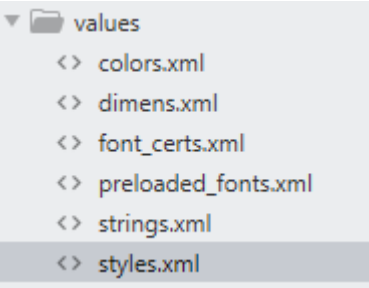


1. activity_main_drawer.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menuxmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
tools:showIn="navigation_view">

<groupandroid:checkableBehavior="single">
<item
android:id="@+id/nav_sentiment_prediction"
android:icon="@drawable/ic_sentiment"
android:title="Predict Sentiment"/>
<item
android:id="@+id/nav_movie_rating_search"
android:icon="@drawable/ic_search"
android:title="Search Movie Rating"/>
<item
android:id="@+id/nav_twitter_analyzer"
android:icon="@drawable/ic_twitter"
android:title="Analyzer Twitter Sentiment"/>
<item
android:id="@+id/nav_explore"
android:icon="@drawable/ic_explore"
android:title="Explore Content"/>
<item
android:id="@+id/nav_recommendation"
android:icon="@drawable/ic_recommendation"
android:title="Recommended Movies"/>
```

```
</group>

<itemandroid:title="User Profile Commands">
<menu>
<item
android:id="@+id/nav_logout"
android:icon="@drawable/ic_logout"
android:title="Logout"/>
<item
android:id="@+id/nav_exit"
android:icon="@drawable/ic_exit"
android:title="Exit"/>
</menu>
</item>
</menu>
```

List of Resource Values Files:



1. colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
<colorname="colorPrimary">#008577</color>
<colorname="colorPrimaryDark">#00574B</color>
<colorname="colorAccent">#D81B60</color>

<colorname="themeBlack">#757575</color>
<colorname="boxBlack">#424242</color>
<colorname="TextWhite">#FFFFFF</color>
<colorname='PrimaryPurple'>#6200EE</color>
<colorname='PurpleMiddle'>#3700B3</color>
<colorname="Black">#000000</color>
<colorname="LightPurple">#9575CD</color>


<colorname="DarkStatusBar">#000000</color>
<colorname="DarkAppBar">#212121</color>
<colorname="DarkBackground">#303030</color>
<colorname="DarkGrey">#424242</color>
<colorname="DarkPrimaryText">#FFFFFFFF</color>
```

```
<colorname="DarkSecondaryText">#B3FFFFFF</color>
<colorname="DarkHintText">#80FFFFFF</color>
<colorname="DarkDividers">#1FFFFFFF</color>
</resources>
```

2. dimens.xml

```
<resources>
<!-- Default screen margins, per the Android Design guidelines. -->
<dimenname="activity_horizontal_margin">16dp</dimen>
<dimenname="activity_vertical_margin">16dp</dimen>
<dimenname="nav_header_vertical_spacing">8dp</dimen>
<dimenname="nav_header_height">176dp</dimen>
<dimenname="fab_margin">16dp</dimen>
</resources>
```

3. strings.xml

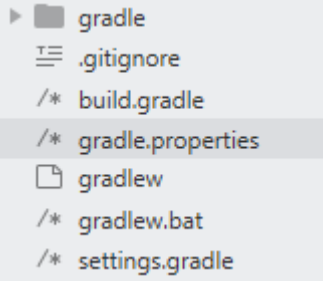
```
<resources>
<stringname="app_name">Major Project</string>
<stringname="navigation_drawer_open">Open navigation drawer</string>
<stringname="navigation_drawer_close">Close navigation drawer</string>
<stringname="nav_header_title">Android Studio</string>
<stringname="nav_header_subtitle">android.studio@android.com</string>
<stringname="nav_header_desc">Navigation header</string>
<stringname="action_settings">Settings</string>


<stringname="server_error">Server Error</string>
<stringname="sentiment_activity_result">Sentiment Prediction Result</string>
<stringname="Submit">Submit</string>
<stringname="Sentiment_activity">Sentiment Prediction</string>
<stringname="enter_movie_name">Enter movie name</string>
<stringname="movie_rating_activity">Search movie rating</string>
<stringname="search">Search</string>

<stringname="selection_menu">Home</string>
<stringname="movie_rating_menu">Movie Rating Lookup</string>
<stringname="movie_rating_result">Rating Result</string>
<stringname="title_activity_rating_error">RatingErrorActivity</string>
<stringname="rating_error_title">Error</string>
</resources>
```

4. styles.xml

```
<resources>

<!-- Base application theme. -->
```

```
<stylename="AppTheme"parent="Theme.AppCompat.Light.DarkActionBar">
<!-- Customize your theme here. -->
<itemname="colorPrimary">@color/PrimaryPurple</item>
<itemname="colorPrimaryDark">@color/PurpleMiddle</item>
<itemname="colorAccent">@color/colorAccent</item>
</style>

<stylename="AppThemeNavigation"parent="Theme.AppCompat.Light.DarkActionBar">
<!-- Customize your theme here. -->
<itemname="colorPrimary">@color/PrimaryPurple</item>
<itemname="colorPrimaryDark">@color/PurpleMiddle</item>
<itemname="colorAccent">@color/colorAccent</item>
<itemname="windowNoTitle">true</item>
<itemname="windowActionBar">false</item>

</style>

<stylename="AppTheme.NoActionBar">
<itemname="windowActionBar">false</item>
<itemname="windowNoTitle">true</item>
</style>

<stylename="AppTheme.AppBarOverlay"parent="ThemeOverlay.AppCompat.Dark.ActionB
ar"/>

<stylename="AppTheme.PopupOverlay"parent="ThemeOverlay.AppCompat.Light"/>

<stylename="textBoxStyle">
<itemname="android:elevation">8dp</item>
<itemname="android:textColor">@color/Black</item>
<itemname="android:background">@drawable/my_textbox_layout</item>
<itemname="android:shadowColor">@color/Black</item>
<itemname="android:gravity">center</item>
</style>

<stylename="buttonStyle">
<itemname="android:elevation">8dp</item>
<itemname="android:textSize">16sp</item>
<itemname="android:background">@drawable/button_background</item>
<itemname="android:shadowColor">@color/Black</item>
<itemname="android:gravity">center</item>
<itemname="android:textColor">@color/TextWhite</item>
<itemname="android:textAllCaps">false</item>
</style>
</resources>
```

List of Gradle Build Files:



1. build.gradle

```
// Top-level build file where you can add configuration options common to all
sub-projects/modules.

buildscript {
    repositories {
        google()
        jcenter()

    }
    dependencies {
        classpath 'com.android.tools.build:gradle:3.4.2'
        classpath 'com.google.gms:google-services:4.2.0'
// NOTE: Do not place your application dependencies here; they belong
// in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        google()
        jcenter()

    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

2. app build.gradle

```
apply plugin: 'com.android.application'

android {
```

```
    compileSdkVersion 29
    buildToolsVersion "29.0.0"
    defaultConfig {
        applicationId "com.example.majorproject"
        minSdkVersion 23
        targetSdkVersion 29
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-
optimize.txt'), 'proguard-rules.pro'
        }
//From android N
        repositories {
            maven { url "https://jitpack.io" }
            mavenLocal()
            jcenter()
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'androidx.appcompat:appcompat:1.0.2'
    implementation 'androidx.legacy:legacy-support-v4:1.0.0'
    implementation 'com.google.android.material:material:1.0.0'
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test:runner:1.2.0'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.2.0'

//From android N
    implementation 'com.github.ybq:Android-SpinKit:1.2.0'

//From firebase
//implementation 'com.google.firebase:firebase-core:16.0.9'
    implementation 'com.google.firebase:firebase-core:17.0.0'
    implementation 'com.google.firebase:firebase-auth:18.0.0'
}
apply plugin: 'com.google.gms.google-services'
```