PRODUCT REVIEW ANALYSIS FOR GENUINE RATING

Submitted to:

Sagar Pande

Assistant Professor- Lpu

Subject:

Artificial Intelligence

INT-404

Project By:

Name	*Registration No*	*Roll No*	*Section*
B. Mary Yavanika	***11802373***	***22***	***K18GX***
Madiha Javed	***11802295***	***31***	***K18GX***

TABLE OF CONTENTS:

INTRODUCTION

- **ABSTRACT**

This system is used to help a user to get a product at its best choice and best price. The Admin and the user part both are an Android Application thus the front end uses Android Studio and SQL Server as its Backend. This system allows the user to look into a product with different reviews and rating rated by other users and for his simplicity the system shows the overall average rating of that product. The System also allows the user to compare 2 products of the same kind or same category and to rate and review the product as he wishes too but limiting to only once per product.

The System is meant to give a rough as well as a much detailed idea of whether the user should go for a product. The System gives a list of sellers and its price offered which is added by the admin for the user's convenience and preference. The Users role is to check out for the product using all the resources offered by the system in finding the best product and give his rating and review. The User is also allowed to give his feedback. The Admin is responsible to add update or delete a product while adding seller's information as well and likewise view users and their feedbacks.

- **MOTIVATION**

Today E-commerce have become an important part of our day to day life and people are getting dependent on these website products. The user reviews too, are becoming important for customers.

So, through this project we are building a chrome plug-in, which rates the E-commerce products based on sentiment analysis of user reviews. With this shopping for customers will become very convenient as well as time saving from reviewing large set of users reviews.

- **AIM**

Our motive is to generate future wise rating of the product. Comments of users will contribute to generate overall rating of the product.

The goal is to generate rating for products based on customer reviews. Main focus of our project is textually data mining of user comments based on sentiment analysis. We will achieve this using Naive Baye's algorithm as classifier, NLP, opinion word, opinion target and opinion analysis for excluding some basic limitation of sentiment analysis

- **FUTURE SCOPE**

Service based rating of vendors and Easy product recommendations directly from website.

- **OBJECTIVE**

To extract and analyse opinions from online reviews, it is unsatisfactory to merely obtain the overall sentiment about a product. In the most cases, customers expect to find fine-grained sentiments about an aspect or feature of a product that is reviewed.

We focuses on positive and negative sentiments about each topic of the product that are useful for the customers as well as focuses on how to improve aspect-level opinion mining for online customer reviews.

- **PROPOSED SYSTEM**

We propose an advanced Products Review analysis system which provides a platform to registered users to rate a particular or multiple products using this system. The system uses product review analysis in order to achieve desired functionality. Product review analysis is a web application which consist multiple products added by admin to review to rate and review them. The System takes reviews of various users, based on their personal opinion, system will specify whether the posted product is good, bad, or worst. We use a database of sentimentbased keywords along with positivity or negativity weight in database and then based on these sentiment keywords mined in user review is ranked.

Once the user login to the system he views multiple products and gives review about the product. User can view product description, price and links to buy the product. System will use database and will match the review with the keywords in database and will rank the review. The role of the admin is to add new products, their description and also provide link to buy the product into database. Admin can also view added products, view registered users and view system related feedbacks from the registered users. This application is also useful for the users who want to buy new product. This system helps to find out good product based on multiple user's positive reviews.

ADVANTAGES:

- The user has to Login to make use of the system keeping the data secure.

- The system helps the user to get the best product with many resources.
- Minimizes user's time.
- User can rate and review a product.
- User can post their own review about the product.
- People can easily decide whether the product is good or bad by using this application.
- Since system ranks the feedback based on the weight age of the keywords in database, so the result is appropriate.

DISADVANTAGES:

- It requires active internet connection else error may occur.
- Wrong reviews and ratings will affect the overall ratings of a product.
- System will match the opinion with those keywords which are in database rest of the words are ignored by the system.

APPLICATION:

- This system can be used to by Ecommerce Institution to help their users to get the best product.
- Increase the sales of reviewed products
- Add SEO-friendly review scores to your Google search results

REFERENCES:

- http://www.takeoffprojects.com/
- https://nevonprojects.com/

RELATED WORK:

- What is product review?

In electronic commerce, product reviews are used on shopping sites to give customers an opportunity to rate and comment on products they have purchased, right on the product page. Other consumers can read these when making a purchase decision. Often, the company will include a URL on printed literature or e-mail marketing to invite customers to review their service after a transaction has been completed.

- The Importance of Product Reviews

Amazon, Wal-mart, and Target aren't just popular retailers — their websites also serve as resources. Thousands of people flock to these sites daily to research products by way of customer reviews. Many big stores like these are used as search sources when others don't include enough information for shoppers to make an educated decision.

Product reviews are an essential part of an online store's branding and marketing. They help build trust and loyalty, and typically describe what sets your products apart from others.Today we're going to learn how reviews help your store increase sales, and how you can convince customers to leave more of them. Read on to get started.

- How Reviews help the store?

Savvy shoppers almost never purchase a product without knowing how it's going to work for them. They read the good, the not-so-good, and the downright ugly to make the all-important decision: should I pull out my wallet and take the plunge?

The immediate benefit of reviews is that they can make your future customers feel that much more confident. The more reviews you have, the more convinced a shopper will be that they're making the right decision.

Reviews can help increase a store's online presence, too. Since customer feedback appears on each product's page, reviews can help pages be found on search engines via unique keywords.

Additionally, if customers share products they've reviewed across their social networks, it's more exposure for those products and your brand.

More reviews = more exposure for your store, long-term.

Reviews can also help you better understand your products. You can spend hours using a product, but chances are customers will notice things you never would. That means your customers can give you great feedback, ideas for improvements, or even incredible marketing ideas!

Having said all this, unless you currently sell a product that's going viral or has been seen on Shark Tank, garnering customer reviews is not easy. There are, however, many simple and affordable ways to entice customers to tell the world what they think of the products they've bought.

- Screenshots of the code

1.

```
import os
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'

import json
import numpy as np
import keras.backend as K
from keras.utils import to_categorical
from keras.preprocessing.text import Tokenizer
from keras.preprocessing import sequence
from keras.models import Sequential
from keras.layers import Dense, Dropout, Embedding, LSTM, Bidirectional


# 1. Loading the data
print("loading data...")

pos_file_name = "pos_amazon_cell_phone_reviews.json"
neg_file_name = "neg_amazon_cell_phone_reviews.json"
pos_file = open(pos_file_name, "r")
neg_file = open(neg_file_name, "r")
pos_data = json.loads(pos_file.read())['root']
neg_data = json.loads(neg_file.read())['root']
print("Posititve data loaded. ", len(pos_data), "entries")
print("Negative data loaded. ", len(neg_data), "entries")

print("done loading data...")

plabels = []
nlabels = []

# 2.Process reviews into sentences
pos_sentences, neg_sentences = [], []
for entry in pos_data :
    pos_sentences.append(entry['summary'] + " . " + entry['text'])
    plabels.append(1)
for entry in neg_data :
    nlabels.append(0)
    neg_sentences.append(entry['summary'] + " . " + entry['text'])
print(len(pos_sentences))
```

2.

```
# 2.Process reviews into sentences
pos_sentences, neg_sentences = [], []
for entry in pos_data :
    pos_sentences.append(entry['summary'] + " . " + entry['text'])
    plabels.append(1)
for entry in neg_data :
    nlabels.append(0)
    neg_sentences.append(entry['summary'] + " . " + entry['text'])
print(len(pos_sentences))
print(len(neg_sentences))

texts = pos_sentences + neg_sentences
labels = [1]*len(pos_sentences) + [0]*len(neg_sentences)

# print("after app", labels)

#print(type(pos_sentences), pos_sentences.shape, type(neg_sentences), neg_sentences.shape)
#print(type(texts), texts.shape, type(labels), labels.shape)

# 3. Tokenize
tokenizer = Tokenizer()
tokenizer.fit_on_texts(texts)
sequences = tokenizer.texts_to_sequences(texts)

word_index = tokenizer.word_index
print('Found %s unique tokens.' % len(word_index))

MAX_SEQUENCE_LENGTH = 50

data = sequence.pad_sequences(sequences, maxlen=MAX_SEQUENCE_LENGTH)

# print(labels)

labels = np.array(labels)
print('Shape of data tensor:', data.shape)
print('Shape of label tensor:', labels.shape)

# split the data into a training set and a validation set

```

3.

```
indices = np.arange(data.shape[0])
np.random.shuffle(indices)
data = data[indices]
labels = labels[indices]

rest_data = data[3000:]
rest_labels = labels[3000:]


data = data[:5000]
labels = labels[:5000]

VALIDATION_SPLIT = 0.2
nb_validation_samples = int(VALIDATION_SPLIT * data.shape[0])

print(data.shape, labels.shape, nb_validation_samples)

print(labels)

x_train = data[:-nb_validation_samples]
y_train = labels[:-nb_validation_samples]
x_val = data[-nb_validation_samples:]
y_val = labels[-nb_validation_samples:]

print(len(x_train), len(y_train))

#GloVe
embeddings_index = {}
f = open('glove.6B/glove.6B.50d.txt', 'r', encoding = 'utf-8')
for line in f:
    values = line.split()
    word = values[0]
    coefs = np.asarray(values[1:], dtype='float32')
    embeddings_index[word] = coefs
f.close()

print('Found %s word vectors.' % len(embeddings_index))

EMBEDDING_DIM = MAX_SEQUENCE_LENGTH

```

4.

```
EMBEDDING_DIM = MAX_SEQUENCE_LENGTH

embedding_matrix = np.zeros((len(word_index) + 1, EMBEDDING_DIM))
for word, i in word_index.items():
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        # words not found in embedding index will be all-zeros.
        embedding_matrix[i] = embedding_vector


from keras.layers import Embedding

embedding_layer = Embedding(len(word_index) + 1,
                            EMBEDDING_DIM,
                            weights=[embedding_matrix],
                            input_length=MAX_SEQUENCE_LENGTH,
                            trainable=False)

def precision(y_true, y_pred):
    true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
    predicted_positives = K.sum(K.round(K.clip(y_pred, 0, 1)))
    precision = true_positives / (predicted_positives + K.epsilon())
    return precision

def recall(y_true, y_pred):
    true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
    possible_positives = K.sum(K.round(K.clip(y_true, 0, 1)))
    recall = true_positives / (possible_positives + K.epsilon())
    return recall


# Training the LSTM model

batch_size = 128

model = Sequential()

model.add(embedding_layer)

```

5.

```
    return precision

def recall(y_true, y_pred):
    true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
    possible_positives = K.sum(K.round(K.clip(y_true, 0, 1)))
    recall = true_positives / (possible_positives + K.epsilon())
    return recall


# Training the LSTM model

batch_size = 128

model = Sequential()

model.add(embedding_layer)

model.add(LSTM(64))

model.add(Dropout(0.50))

model.add(Dense(1, activation='sigmoid'))

# try using different optimizers and different optimizer configs

model.compile('adam', 'binary_crossentropy', metrics=['accuracy', precision, recall])

print('Train...')

model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=16,
          validation_data=[x_val, y_val])

x = model.evaluate(rest_data[:5000], rest_labels[:5000])
print("Loss: ", x[0])
print("Accuracy: ", x[1])
print("Precision: ", x[2])
print("Recall: ", x[3])
```