ABSTRACT

The rise in E — commerce has brought a significant rise in the importance of customer reviews. There are hundreds of review sites online and massive amounts of reviews for every product. Customers have changed their way of shopping and according to a recent survey, 70 percent of customers say that they use rating filters to filter out low rated items in their searches.

The ability to successfully decide whether a review will be helpful to other customers and thus give the product more exposure is vital to companies that support these reviews, companies like Google, Amazon and Yelp!

There are two main methods to approach this problem. The first one is based on review text content analysis and uses the principles of natural language process (the NLP method). This method lacks the insights that can be drawn from the relationship between costumers and items. The second one is based on recommender systems, specifically on collaborative filtering, and focuses on the reviewer's point of view. Use of the user's similarity matrix and applying neighbors' analysis are all part of this method. This method ignores any information from the review text content analysis.

We have created an artificial intelligence software that can make a rating prediction based on the review you have written using the Semi Supervised Learning method and the RC algorithm. We used very simple codes to semi accurately predict the user rating.

DATASET DESCRIPTION

This dataset consists of a few thousand Amazon customer reviews (input text) and star ratings (output labels) for learning how to train fast text for sentiment analysis.

The idea here is a dataset is more than a toy - real business data on a reasonable scale - but can be trained in minutes on a modest laptop.

The dataset is taken from kaggle and the link for the same is as below:

https://www.kaggle.com/datasets/bittlingmayer/amazonreviews.

The dataset contains number of reviews from hundreds of users. These reviews range from reviews about the product or bad reviews about the products. It also contains the ratings of the products along with the reviews to train the machine so as to predict the rating of the reviews entered by the users.

A snippet of the dataset is given below:

	Unnamed: 0	Date	URL	Review_Title	Author	Rating	Review_text	Review_helpful	Sentiment	Subjectivity
0	0	Reviewed in India on 10 August 2018	https://www.amazon.in/Logitech-Prodigy-G213-Ga...	It s a logitech Definitely worth buying it	Aqib Mehmood	5	A really awesome keyboard i was actually go...	77	0.359722	0.552778
1	1	Reviewed in India on 27 March 2018	https://www.amazon.in/Logitech-Prodigy-G213-Ga...	Great deal got it for With Lightening ...	Chauhan	4	I know its costly but its worth your money ...	55	0.233125	0.468542
2	2	Reviewed in India on 19 December 2018	https://www.amazon.in/Logitech-Prodigy-G213-Ga...	Loved it	Smok3y	5	I had been contemplating to buy this for a l...	18	0.338750	0.568750
3	3	Reviewed in India on 25 June 2020	https://www.amazon.in/Logitech-Prodigy-G213-Ga...	Not a good purchase please read description	Kumar Saharsh	1	SO Very very small keys For fast typers w...	11	0.139083	0.507840
4	4	Reviewed in India on 27 October 2018	https://www.amazon.in/Logitech-Prodigy-G213-Ga...	Good only when new	Amazon Customer	2	Good to use keyboard while it is new but it...	15	0.165427	0.698140

ALGORITHM DESCRIPTION

The algorithms used in the development of the program are:

→*Logistic Regression:* Logistic regression is a Machine Learning classification algorithm that is used to predict the probability of certain classes based on some dependent variables. In short, the logistic regression model computes a sum of the input features (in most cases, there is a bias term), and calculates the logistic of the result.

→*K Nearest Neighbours:* The k-nearest neighbours algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point.

→*Support Vector Classifier:* The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyper plane.

→*Decision Tree Classifier:* The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data).

→*Random Forest Classifier:* Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.

→*Multinomial Naive Bayes:* Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of email or newspaper article.

RESULT AND INFERENCE

The snapshot of the codes and the result are given below:

Load dataset from kaggle

```
import pandas as pd
import numpy as npS
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer

df = pd.read_csv('finalReview.csv')

df.head(10)
```

Importing all the libraries and training it based on text review and rating

```
from sklearn.linear_model import RidgeClassifier
from sklearn.semi_supervised import SelfTrainingClassifier
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer


X_train, X_test, y_train, y_test = train_test_split(df.Review_text,
                                                    df.Rating,
                                                    test_size=0.3,
                                                    random_state=25,
                                                    shuffle=True)


X_CountVectorizer = CountVectorizer(stop_words='english')
X_train_counts = X_CountVectorizer.fit_transform(X_train)

X_TfidfTransformer = TfidfTransformer()
X_train_tfidf = X_TfidfTransformer.fit_transform(X_train_counts)


model_semi = SelfTrainingClassifier(RidgeClassifier())
model_semi.fit(X_train_tfidf, y_train)
```

Rating prediction given based on the review entered by the user

```
text = input()

text = [text]

text_counts = X_CountVectorizer.transform(text)

#Prediction Processing
prediction = model_semi.predict(text_counts)

f"Predicted rating is {prediction[0]}"
```

```
not so good
'Predicted rating is 5'
```

From this project, we draw the inference that upon using the machine learning algorithms in accordance with various python libraries, the rating from (1-5) of the user can be predicted based on the reviews of the user and it can be used in various industries to semi accurately predict the rating.