International Journal of Research in Circuits, Devices and Systems

E-ISSN: 2708-454X
P-ISSN: 2708-4531
IJRCDS 2024; 4(1): 01-10

www.circuitsjournal.com
Received: 02-11-2023
Accepted: 09-12-2023

Navtej Kumar Singh
Department of CSE, Apex Institute of Technology, Chandigarh University, Mohali, Punjab, India

Divyansh Khandelwal
Department of CSE, Apex Institute of Technology, Chandigarh University, Mohali, Punjab, India

Corresponding Author:
Navtej Kumar Singh
Department of CSE, Apex Institute of Technology, Chandigarh University, Mohali, Punjab, India

Sentiment analysis for amazon music instruments review

Navtej Kumar Singh and Divyansh Khandelwal

DOI: https://doi.org/10.22271/27084531.2024.v5.i1a.59

Abstract
This paper presents an approach for sentiment analysis to mine sentiments of user reviews by classifying them as per their corresponding ratings along with a graphical visualization of the tests and results. Our proposed solution is based on a convolutional neural network method for classification of the review text. The implementation makes use of Python and its libraries for Natural Language Processing and Data Visualization. The solution also contains a simple predictor tool for user reviews given as input at run-time. The results of our work can be used to better understand customer feedback by e- commerce companies to mitigate the factors that lead to low ratings and to optimize the factors that customers appreciate. The novelty of this research lies mainly in the usage of neural networks and Python package SpaCy for sentiment analysis. We put forward our approach along with its implementation, evaluation, visualization and conclusion.

Keywords: Sentiment Analysis, SpaCy, Text Mining, NLP

Introduction
Customer reviews and ratings on ecommerce platforms like Amazon [1] have a substantial influence on the product reputation as they act as drivers for prospective buyers before they decide to make purchases. Text mining methods like Sentiment Analysis can be leveraged to uncover customer opinions and understand the general customer sentiment about a product. Amazon gives its customers the option of providing ratings and comments as feedback for its products. The ratings are numeric, ranging from 1 to 5, while the feedback is in the form of text. While ratings can be useful, looking at the overall performance of a product, it has been observed that the actual sentiment of the customers is better reflected in their comments. Product review is the third most important factor that buyers consider while buying products on Amazon, after price and shipping cost [2]. Capturing the actual sentiment through comments can provide the company with an understanding of how particular products are performing in the market, which products need to be changed or upgraded, which products can be priced at higher rates, which products can be sold at discounted prices etc. Using manual labor to go through the feedback comments is certainly a tedious task, especially on a website with humongous traffic such as Amazon. A sentiment analyzer can be used in such a case, that can mine the sentiment of the comments given by a customer. This paper elaborates the implementation of text mining methods for sentiment analysis of user reviews. The goal is to predict an overall sentiment, either positive or negative, of a customer review for around 10,000 musical instruments listed on Amazon. Amazon Musical Instruments dataset on Kaggle [3] has been taken as the source data. The dataset contains 10,262 rows and 9 columns. Table 1. lists the columns along with a short description.

Table 1: List of columns and their description contained in the dataset used in the implementation

Attribute	Description
reviewerID	Unique identifier for reviewer
asin	Unique identifier for product
reviewerName	Full name of the reviewer
helpful	Helpfulness rating of the review
reviewText	Text submitted in the review by the user
overall	Rating of the product given by the user
summary	Summary or title of the review given by the user
unixReviewTime	Unix timestamp [4] for the Date and Time at which review was submitted
reviewTime	Date and Time at which review was submitted

The paper is structured in the following manner. Section II provides an overview of the related work of analyzing sentiment analysis approaches used before. Section III briefs our proposed approach of correlating reviews to product ratings and categorization of reviews. Section IV summarizes our evaluation of classification and categorization. Section V encapsulates the results of our analysis and conclusions drawn. Section VI concludes the paper with sources and recommendations.

Related Work

A Combined Method for Usage of NLP Libraries Towards Analyzing Software Documents [5] gives an insight into selection of NLP libraries for implementation. The paper compares four different NLP libraries generally used for analysis of software documents and proposes a combined approach that utilizes the strengths of all four libraries. The paper reveals an experimental study where spaCy [6] outperforms other NLP libraries (NLTK [7], Stanford CoreNLP [8] and OpenNLP [9]) on part-of-speech tagging. Part-of-speech tagging forms the base for Sentiment Analysis as the 'adjectives' in the text perform an important role in determining the overall sentiment behind the review.

CNN Based Framework for Sentiment Analysis of Tweets [10] shows a comparison of different techniques for Sentiment Analysis for tweets from Twitter [11]. It compares Naïve Bayes, Decision Trees, Support Vector Machine and Convolutional Neural Networks. Convolution Neural Network (CNN) gives the best results on the dataset. The accuracy achieved by CNN is approximately 80.2% which very high as compare to other classifiers applied to the dataset. The insights proved useful for our selection of technique as well as NLP library.

The work in [12] presents usage of the SpaCy package of python to preprocess the data before, each individual review has been tokenized, lemmatized, filtered for stop words and vectorized in order to prepare the data viable for the machine learning model.

Rebecca Williams, Nikita Jindal and Anurag Batra present an analysis of tweets posted on Triple Talaq from the year 2002 to 2019 [13]. The paper enlists the basic operations required for processing tweet data and the steps required for creating the Natural Language Processing pipeline and building the model using SpaCy and python.

	SPACY	NLTK	CORENLP
Programming language	Python	Python	Java / Python
Neural network models	✓	✗	✓
Integrated word vectors	✓	✗	✗
Multi-language support	✓	✓	✓
Tokenization	✓	✓	✓
Part-of-speech tagging	✓	✓	✓
Sentence segmentation	✓	✓	✓
Dependency parsing	✓	✗	✓
Entity recognition	✓	✓	✓
Entity linking	✓	✗	✗
Coreference resolution	✗	✗	✓

Fig 1: Comparison of functionalities offered by SpaCy, NLTK and CoreNLP [14]

Proposed Approach

We have created a sentiment analysis machine learning model using natural language processing techniques and neural networks with SpaCy. SpaCy is a free, open-source library for advanced Natural Language Processing in Python. The package SpaCy provides out-of-the-box models that contain information about vocabularies, trained vectors, syntaxes and entities. These models are loaded to create a pipeline that outputs a wide range of document properties such as - tokens, token's reference index, part of speech tags, entities, vectors and sentiment.

By enabling the use of the SpaCy NLP pipeline, we were able to build and train a convolutional neural network (CNN) for classifying text data.

Convolutional Neural Networks

Neural networks are set of algorithms that recognize patterns. The patterns are vectors containing numbers that get translated from any real-world data that could be in the form of images, sound, or as in our case, text. A convolutional neural network is a neural network that applies convolutional layers to local features. In Natural Language Processing, CNNs take inputs in the form of words represented by vectors. Applying a number of filters on the word vectors creates a Convolutional Feature Map. The maximum valued result from each vector is then used where it gets transformed from 1x4 into 1x3. The image in Figure 2 shows the process for a sample input text "I love my new phone:)"

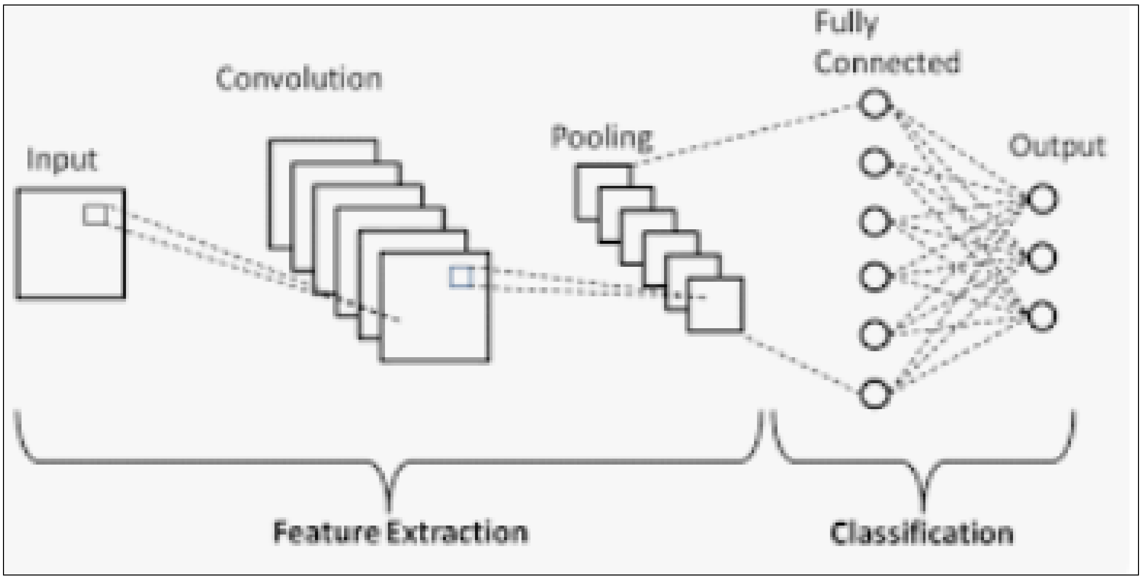

Fig 2: Convolutional Neural Network for NLP. Image source [15]

In order to train a multi-label CNN text classifier for the text reviews, SpaCpeline of tasks undertaken in SpaCy for language processing where raw text is input and a spaCy Doc object is output [16]. The components in the pipeline can be customized to meet the needs of specific applications (e.g., adding custom pattern matchers) and the order in which tasks are executed can be changey's *TextCategorizer* component was used. Figure 3. shows the typical pid.

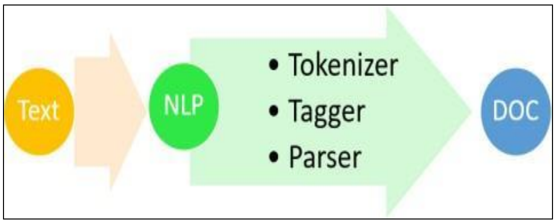

Fig 3: The SpaCy Natural Language Processing pipeline

The SpaCy pipeline provides a number of document properties such as - tokens, part-of-speech (POS) tags, vectors, sentiment etc. The text-preprocessing that involves these functionalities are provided out-of-the-box by SpaCy's NLP pipeline with the *nlp()* constructor call [17].

Tokenization - A document in SpaCy gets tokenized into sentences and tokens. These can be accessed by iterating the document.

Part-of-Speech (POS) Tagging: Once tokenization is completed, SpaCy parses and tags a document. SpaCy makes use of a statistical model. We have used the *en_core_web_sm* which is a pre-trained statistical model for English in SpaCy. This model enables SpaCy to make a prediction of which tag or label most likely applies in this context [18].

Apart from SpaCy, other Python packages used in the implementation were *Pandas*, *MatPlotLib* and *scikit-learn*. *Pandas DataFrame* was used to load the dataset into a two-dimensional tabular data structure with labeled axes (rows and columns). *Pandas DataFrame* consists of three principal components, the data, rows, and columns. *MatPlotLib* package from Python was used to plot graphs for data visualization and for plotting the results of accuracy report. The *scikit-learn* library provided APIs for splitting the data into Train and Test sets with configurable options for the split ratio and randomization.

```
--------------------- DATASET ANALYSIS ------------------------
Number of unique customers in the dataset : 1429
Number of unique products that were reviewed : 900

Columns having blank values:
reviewerID          0
asin                0
reviewerName       27
helpful             0
reviewText          7
overall             0
summary             0
unixReviewTime      0
reviewTime          0
```

Fig 4: Analysis of the dataset to find out number of unique customers and products and columns with blank values

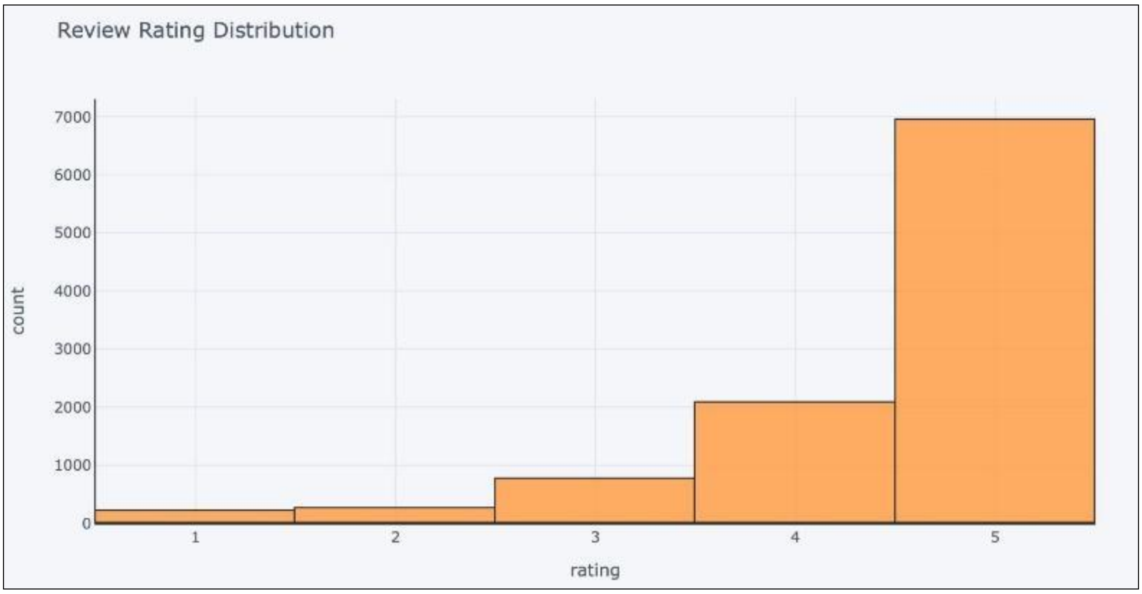

Fig 5: Histogram showing rating distribution

Data Transformation

Table 2 lists the data transformation operations that were done on the dataset along with their purpose.

Implementation

Data Preprocessing

As the first step for any data mining method, dataset was evaluated and preprocessing techniques were applied to prepare the data for mining. The Dataset contained 10,262 records. These records were analyzed for their distribution of unique values for products and customers. The dataset was also analyzed for columns having blank values. These findings have been shown in Figure 4.

The dataset was also analyzed for distribution of ratings. Histograms for each value of rating from the column 'overall' were plotted as shown in Figure 5.

Table 2: List of data transformations applied on the dataset

Operation	Purpose
The columns 'summary' and 'reviewText' were concatenated and stored into single new column 'Review'.	This was done for cases where users did not fill one of the two fields.
All the columns in the dataset except 'Review' and 'overall' were dropped.	The columns that were not required for the data mining were dropped. 'Review' was used as the source of text for the text mining, while 'overall' column containing the Rating was to be used as the classifier in order to train the data.
The column values for 'overall' ratings were changed as follows – Ratings 4, 5 => 1 Ratings 1,2,3 => 0	Since the goal was to classify reviews as either positive or negative, we needed the classifier to have binary values. Hence, ratings that were 4 and 5 were changed to value '1' and ratings of 1,2 or 3 were changed to value '0'.

Histograms for binary values of rating from the column 'overall' plotted are shown in Figure 6.

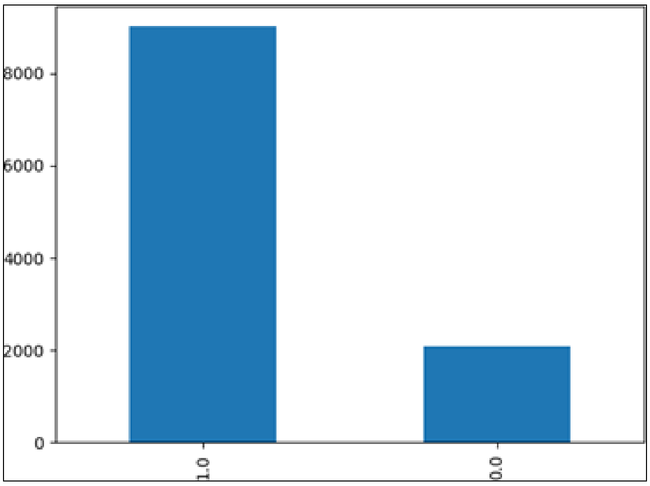

Fig 6: Histogram showing distribution of ratings from column 'overall' post transformation

As observed from the Histogram plots, it was observed that records with rating '1' were very high in number compared to number of records with '0' ratings. Training the dataset with a skewed distribution could lead to biased results. Hence, to balance the classes, equal samples from each class were selected.

WordCloud Analysis

WordCloud was implemented as part of analysis of the reviews to get an idea about the words most frequently found in the best and worst ratings. Figure 7 shows the WordCloud generated for the text reviews with bad ratings (Ratings of 1,2 or 3).

Similarly, Figure 8 shows the WordCloud generated for reviews with good ratings (Ratings 4 and 5).

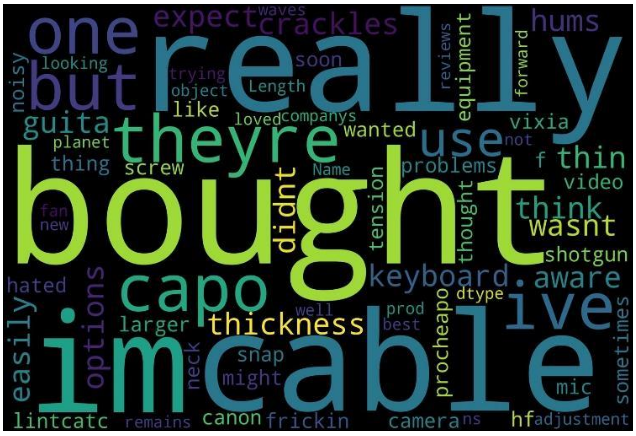

Fig 7: WordCloud generated for reviews with Ratings 1,2 and 3

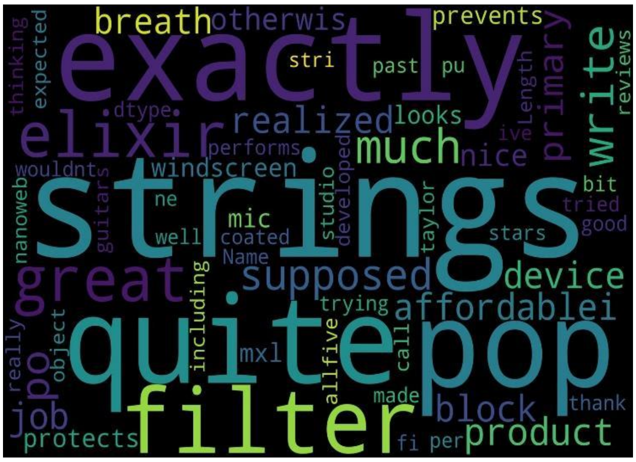

Fig 8: WordCloud generated for reviews with Ratings 4 and 5

Preparing Train and Test Sets

After selecting samples from each class, the dataset was split into Train and Test data with a split ratio of 0.8, i.e., 80 percent of the data was used for training and 20 percent for testing. The data was shuffled while doing the split so that any possible bias could be eliminated from the order in which the training data would be loaded.

Training the model

The built-in en_core_web_sm NLP pipeline has been used in the implementation. The TextCategorizer component (*textcat*) is used as the classifier by adding labels for the TextCategorizer using *add_label()*. The method call *nlp.begin_training()* begins the training. This method returns the initial optimizer function which is later used in *nlp.update()* to update the weights of the underlying model. In the implementation, the training was carried out on batches of data. In each batch of data, the text and labels were separated, and fed into the *nlp.update()* along with the optimizer function. The method *nlp.update()* also uses a dictionary called *loss* to return the loss at the end of each training iteration. To avoid overfitting, some portion of the training data was skipped over by adding the 'dropout' parameter in nlp.update(). We have currently used 25 iterations for the training with a dropout of 0.2.

Evaluating the model

The performance of the model was evaluated after each iteration of the training. The model was evaluated with test data, which was created during the dataset split. Figure 9 shows a snapshot of iterations of training and model evaluation listing the Loss, Precision, Recall and F-Score.

```
Training iteration 1
1.8437582568185462  0.824999999965625   0.7983870967420005  0.8114754098028084
Training iteration 2
0.5585616164880776  0.8134920634597821  0.8266129031924753  0.8199999999672001
Training iteration 3
0.21620931033885427 0.8188976377630355  0.8387096773855359  0.8286852589311281
Training iteration 4
0.0795434789436058  0.8108108107795053  0.846774193514243   0.8284023668312268
Training iteration 5
0.041733352995606765    0.8139534883405444  0.846774193514243   0.8300395256588917
Training iteration 6
0.019665230083887764    0.8085937499684143  0.8346774193211823  0.821428571395975
Training iteration 7
0.007986918516166952    0.8063241106400663  0.8225806451281217  0.8143712574525201
Training iteration 8
0.006263510082220591    0.8102766798098705  0.8266129031924753  0.8183632734204246
Training iteration 9
0.0034601143256054456   0.81599999996736    0.8225806451281217  0.8192771084008322
Training iteration 10
0.001889269846572006    0.8145161289994146  0.8145161289994146  0.8145161289994147
Training iteration 11
0.0013025951520262424   0.805555555523589   0.8185483870637682  0.8119999999675199
Training iteration 12
0.0011349536584788211   0.8063241106400663  0.8225806451281217  0.8143712574525201
Training iteration 13
0.0008947911216807825   0.805555555523589   0.8185483870637682  0.8119999999675199
Training iteration 14
```

Fig 9: Snapshot of model training with loss, precision, recall and F-score values

The results were generated and plotted into graphs the measures of loss, precision, and recall and the F-score for each training iteration. Figure 10 shows the values of Loss over the subsequent iterations.

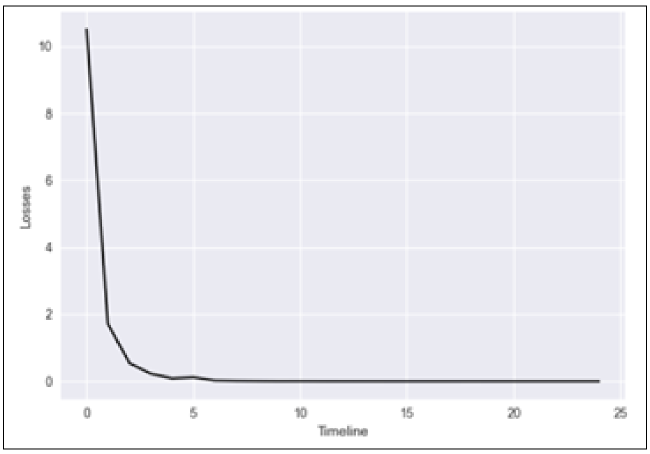

Fig 10: Values of Loss as the iterations progressed

Figure 4.8 shows the values of Precision, Loss and F-Score over 25 iterations. The loss showed a steep decrease as the iterations progressed. Precision, recall and F-score showed a gradual increase and were stable after the first few training iterations.

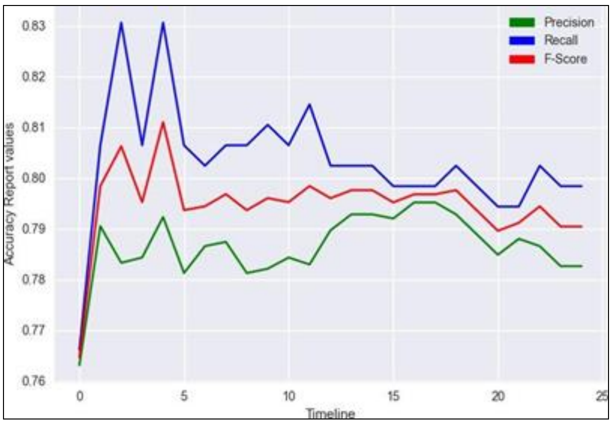

Fig 11: Precision, Recall and F-Score

Testing with varying iterations

The training model was tested for iterations 10, 20, 25 and 100 to determine the optimum value of iterations. For 100 iterations, it was observed that the values of Precision, Recall and F-scores declined over time and the model was overfitting on some data. Figure 4.9 and 4.10 show the results of 100 iterations for training the model.

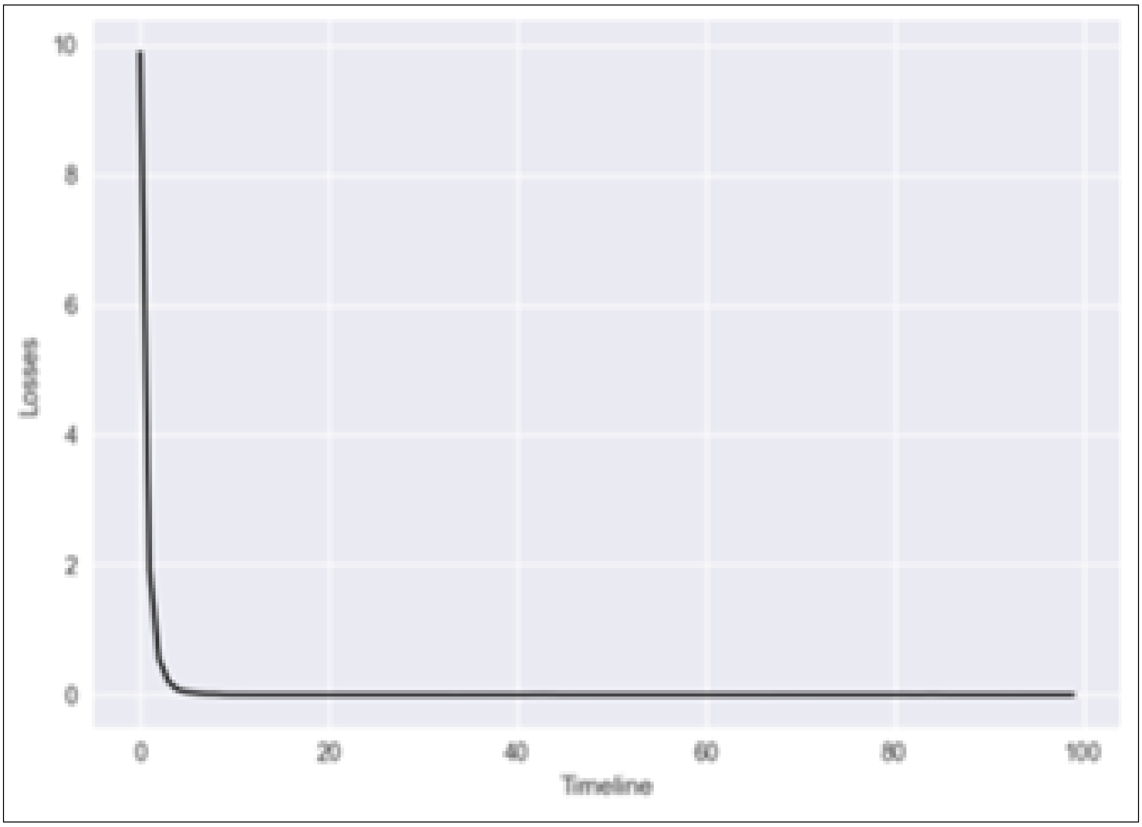

Fig 12: Graph plotting Loss for 100 iterations

Fig 13: Graph plotting Precision, Recall and F-Scores for 100 iterations

Conclusion and Results

Accuracy Report Values

After the model training, the final scores for the accuracy report of the model are listed as below:

```
#creating the objects
logreg_cv = LogisticRegression(random_state=0)
dt_cv=DecisionTreeClassifier()
knn_cv=KNeighborsClassifier()
svc_cv=SVC()
nb_cv=BernoulliNB()
cv_dict = {0: 'Logistic Regression', 1: 'Decision Tree',2:'KNN',3:'SVC',4:'Naive Bayes'}
cv_models=[logreg_cv,dt_cv,knn_cv,svc_cv,nb_cv]


for i,model in enumerate(cv_models):
    print("{} Test Accuracy: {}".format(cv_dict[i],cross_val_score(model, X, y, cv=10, scoring ='accuracy').mean()))

Logistic Regression Test Accuracy: 0.8810059200798708
Decision Tree Test Accuracy: 0.8114224894704574
KNN Test Accuracy: 0.8790565074375868
SVC Test Accuracy: 0.8795439317757772
Naive Bayes Test Accuracy: 0.8038184420263036
```

Testing for sample reviews

We tested the model on test data as well as some sample reviews given as input to the model. The results obtained were the sentiment prediction and the score of that prediction. Figure5.1 to 5.4 show the prediction from the model for single review given as input. This was done to evaluate the prediction and accuracy generated by the model.


```
#creating the objects
logreg_cv = LogisticRegression(random_state=0)
dt_cv=DecisionTreeClassifier()
knn_cv=KNeighborsClassifier()
svc_cv=SVC()
nb_cv=BernoulliNB()
cv_dict = {0: 'Logistic Regression', 1: 'Decision Tree',2:'KNN',3:'SVC',4:'Naive Bayes'}
cv_models=[logreg_cv,dt_cv,knn_cv,svc_cv,nb_cv]


for i,model in enumerate(cv_models):
    print("{} Test Accuracy: {}".format(cv_dict[i],cross_val_score(model, X, y, cv=10, scoring ='accuracy').mean()))

Logistic Regression Test Accuracy: 0.8810059200798708
Decision Tree Test Accuracy: 0.8114224894704574
KNN Test Accuracy: 0.8790565074375868
SVC Test Accuracy: 0.8795439317757772
Naive Bayes Test Accuracy: 0.8038184420263036
```


Fig 14: Accuracy char

```
print("Classification Report:\n",classification_report(y_test, y_pred))

Classification Report:
               precision    recall  f1-score   support

           0       0.93      1.00      0.96      2326
           1       0.91      0.98      0.94      2232
           2       1.00      0.84      0.91      2209

    accuracy                           0.94      6767
   macro avg       0.95      0.94      0.94      6767
weighted avg       0.94      0.94      0.94      6767
```

Fig 15: Classification Report

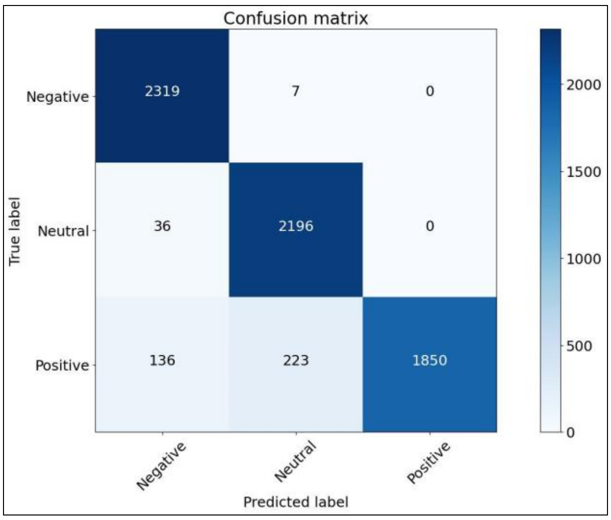

Fig 16: Confusion matrix

Conclusion

The model was trained to generate ‘Positive’ or ‘Negative’ sentiment analysis for a set of around 10,000 user reviews on Amazon for musical instruments. It was trained using 1200 positive and 1200 negative reviews with the overall Rating asthe classifier. The model was trained and after evaluation showed an overall precision of 0.78 and F-score of 0.79.

References

1. Amazon is registered trademark (https://amazon.com)
2. “How Consumers Feel about Amazon” The most important factors driving U.S. buyer decisions on Amazon are price (82%), low shipping costs (70%), positive product reviews (57%), and flexible return policy (49%) Source: https://www.nchannel.com/blog/amazon-statistics/
3. “Amazon Musical Instruments Reviews” Dataset for

musical instruments review and ratings in Kaggle (https://www.kaggle.com/eswarchandt/amazon-music-reviews)

4. What is the unix time stamp?' Source: https://www.unixtimestamp.com/index.php
5. Xinyun Cheng and Xianglong Kong "A Combined Method for Usage of NLP Libraries Towards Analyzing Software Documents" Conference paper. First Online: 03 June 2020 at International Conference on Advanced Information Systems Engineering [https://link.springer.com/chapter/10.1007/978-3-030-49435-3_32]
6. spaCy - Industrial strength Natural Language Processing (https://spacy.io/)
7. Natural Language Toolkit https://www.nltk.org/
8. CoreNLP https://stanfordnlp.github.io/CoreNLP/
9. Apache OpenNLP (https://opennlp.apache.org/)
10. "CNN Based Framework for Sentiment Analysis of Tweets" by Vijay Kumar. International Journal of Innovative Technology and Exploring Engineering [https://www.researchgate.net/publication/343163047_CNN_Based_Framework_for_Sentiment_Analysis_of_Tweets]
11. Twitter (https://twitter.com/)
12. Spacy for NLP Source: https://mahadev001.github.io/Mahadev-Upadhyayula/Sentiment%20Analysis%20via%20NLP/Sentiment%20Analysis%20using%20NLP%20with%20Spacy%20and%20%20SVM.html
13. Rebecca Williams, Nikita Jindal and Anurag Batra Sentiment Analysis using Lexicon based Approach, r, Institute of Information Technology & Managemen https://iitmjanakpuri.com/journals/Volume_10_Issue_1_January-June_2019.pdf#page=70
14. Feature comparison "comparison of the functionalities offered by spaCy, NLTK and CoreNLP https://spacy.io/usage/facts-figures
15. Convolutional Neural Network in Natural Language Processing [Source: https://towardsdatascience.com/convolutional-neural-network-in-natural-language-processing-96d67f91275c]
16. The typical pipeline The typical pipeline of tasks undertaken in spaCy during the NLP process [https://www.peakindicators.com/blog/state-of- the-art-natural-language-processing-with-python-and-spacy#:~:text=The%20typical%20pipeline,level%20NLP%20tasks%20typically%20include%3A&text=Sentence%20segmentation%3A%20breaking%20a%20text%20into%20sentences%20to%20process.]
17. Using Natural Language Processing to Preprocess and Clean Text Data (https://realpython.com/sentiment-analysis-python/#tokenizing)
18. English - Available pretrained statistical models for English [https://spacy.io/models/en]
19. Natural Language Processing and Computational Linguistics: A practical guide pp. 33-50
20. https://books.google.com/books?hl=en&lr=&id=48RiDwAAQBAJ&oi=fnd&pg=PP1&dq=spacy+sentiment+analysis&ots=R3u9NbkZa6&sig=ZIMWc2rNoiqE08FpWkYqk9see8U#v=onepage&q=spacy%20sentiment%20analysis&f=false
21. Chopra Y, Kaushik P, Rathore SPS, Kaur P. Uncovering Semantic Inconsistencies and Deceptive Language in False News Using Deep Learning and NLP Techniques for Effective Management. International Journal on Recent and Innovation Trends in Computing and Communication. 2023;11(8s):681-692. https://doi.org/10.17762/ijritcc.v11i8s.7256
22. Kaushik P. Role and Application of Artificial Intelligence in Business Analytics: A Critical Evaluation. International Journal for Global Academic & Scientific Research. 2022;1(3),01-11. https://doi.org/10.55938/ijgasr.v1i3.15
23. Kaushik P. Deep Learning Unveils Hidden Insights: Advancing Brain Tumor Diagnosis. International Journal for Global Academic & Scientific Research. 2023;2(2):01-22. https://doi.org/10.55938/ijgasr.v2i2.45
24. Kaushik P. Unleashing the Power of Multi-Agent Deep Learning: Cyber-Attack Detection in IoT. International Journal for Global Academic & Scientific Research. 2023;2(2):23-45. https://doi.org/10.55938/ijgasr.v2i2.46
25. Kaushik P, Rathore SPS. Deep Learning Multi-Agent Model for Phishing Cyber-attack Detection. International Journal on Recent and Innovation Trends in Computing and Communication. 2023;11(9s):680-686. https://doi.org/10.17762/ijritcc.v11i9s.7674
26. Kaushik P, Miglani S, Shandilya I, Singh A, Saini D, Singh A. HR Functions Productivity Boost by using AI. International Journal on Recent and Innovation Trends in Computing and Communication. 2023;11(8s):701-713. https://doi.org/10.17762/ijritcc.v11i8s.7672
27. Kaushik P, Singh Rathore SP, Kaur P, Kumar H, Tyagi N. Leveraging Multiscale Adaptive Object Detection and Contrastive Feature Learning for Customer Behavior Analysis in Retail Settings. International Journal on Recent and Innovation Trends in Computing and Communication. 2023;11(6s):326-343. Available from: https://doi.org/10.17762/ijritcc.v11i6s.6938
28. Kaushik P, Yadav R. Reliability design protocol and block chain locating technique for mobile agent. Journal of Advances in Science and Technology (JAST). 2017;14(1):136-141. Available from: https://doi.org/10.29070/JAST
29. Kaushik P, Yadav R. Deployment of Location Management Protocol and Fault Tolerant Technique for Mobile Agents. Journal of Advances and Scholarly Researches in Allied Education [JASRAE]. 2018;15(6):590-595. Available from: https://doi.org/10.29070/JASRAE
30. Kaushik P, Yadav R. Mobile Image Vision and Image Processing Reliability Design for Fault-Free Tolerance in Traffic Jam. Journal of Advances and Scholarly Researches in Allied Education (JASRAE). 2018;15(6):606-611. Available from: https://doi.org/10.29070/JASRAE
31. Ka~~ushik P, Yadav R. Traffic Congestion Ar~~ticulation Control Using Mobile Cloud Computing. Journal of Advances and Scholarly Researches in Allied Education (JASRAE). 2018;15(1):1439-1442. Available from: https://doi.org/10.29070/JASRAE
32. Pratap Singh Rathore S. Analysing the efficacy of training strategies in enhancing productivity and advancement in profession: theoretical analysis in Indian context. International Journal for Global Academic & Scientific Research. 2023;2(2):56-77. Available from: https://doi.org/10.55938/ijgasr.v2i2.49
33. Pratap Singh Rathore S. The Impact of AI on Recruitment and Selection Processes: Analysing the

role of AI in automating and enhancing recruitment and selection procedures. International Journal for Global Academic & Scientific Research. 2023;2(2):78-93. Available from: https://doi.org/10.55938/ijgasr.v2i2.50

34. Rathore R. Application of Assignment Problem and Traffic Intensity in Minimization of Traffic Congestion. IJRST. Jul-Sep 2021;11(3):25-34. Available from: http://doi.org/10.37648/ijrst.v11i03.003
35. Rathore R. A Review on Study of application of queueing models in Hospital sector. International Journal for Global Academic & Scientific Research. 2022;1(2):01-05. Available from: https://doi.org/10.55938/ijgasr.v1i2.11
36. Rathore R. A Study on Application of Stochastic Queuing Models for Control of Congestion and Crowding. International Journal for Global Academic & Scientific Research. 2022;1(1):01-07. Available from: https://doi.org/10.55938/ijgasr.v1i1.6
37. Rathore R. A Study Of Bed Occupancy Management In The Healthcare System Using The M/M/C Queue And Probability. International Journal for Global Academic & Scientific Research. 2023;2(1):01-09. Available from: https://doi.org/10.55938/ijgasr.v2i1.36
38. Sharma T, Kaushik P. Leveraging Sentiment Analysis for Twitter Data to Uncover User Opinions and Emotions. International Journal on Recent and Innovation Trends in Computing and Communication. 2023;11(8s):162-169. Available from: https://doi.org/10.17762/ijritcc.v11i8s.7186
39. Sharma V. A Study on Data Scaling Methods for Machine Learning. International Journal for Global Academic & Scientific Research. 2022;1(1):23-33. Available from: https://doi.org/10.55938/ijgasr.v1i1.4
40. Yadav M, Kakkar M, Kaushik P. Harnessing Artificial Intelligence to Empower HR Processes and Drive Enhanced Efficiency in the Workplace to Boost Productivity. International Journal on Recent and Innovation Trends in Computing and Communication. 2023;11(8s):381-390. Available from: https://doi.org/10.17762/ijritcc.v11i8s.7218