Amazon Reviews Sentiment Analysis, Segmentation, Classification and Prediction leveraging Multi Class Multi Output Classification

A thesis
Submitted to Department of Computer Science & Engineering

In the partial fulfillment of the requirements for

The award of the degree of

MASTER OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

By

G. YOGESH SINGH (20567T7606)

Under the Guidance of

DR. RAMANA NAGAVELLI

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
KU COLLEGE OF ENGINEERING & TECHNOLOGY,
KAKATIYA UNIVERSITY CAMPUS, VIDYARANYAPURI,
WARANGAL – 506 009, TELANGANA
NOVEMBER 2022

KU COLLEGE OF ENGINEERING & TECHNOLOGY

KAKATIYA UNIVERSITY CAMPUS, VIDYARANYAPURI,

WARANGAL - 506 009

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Certificate

This is to certify that this thesis entitled **"Amazon Reviews Sentiment Analysis, Segmentation, Classification and Prediction leveraging Multi Class Multi Output Classification"** that is being submitted by Mr. **G. YOGESH SINGH bearing Roll no. 20567T7606** in the partial fulfillment for the award of the degree of **Master of Technology** in **Computer Science and Engineering** to the **KAKATIYA UNIVERSITY** is a record of work carried out during the academic year 2021-2022 under our guidance and supervision.

Supervisor Head of the Department Principal

Internal Examiner External Examiner

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I adequately cited and referenced the original sources. I declare that the work presented in this project report is original and carried out in the department of Computer Science & Engineering, KU College of Engineering & Technology, Warangal, and Telangana, and has not been submitted elsewhere for any graduate in part or in full.

G. YOGESH SINGH (20567T7606)

ACKNOWLEDGEMENT

There are many people who have helped me directly or indirectly in the successful completion of my project. I would like to take this opportunity to thank one and all.

I would like to express my gratitude to my supervisor **Dr. Ramana Nagavelli** (Associate Professor) for his guidance, kindness, motivation, suggestions, and insight throughout this project. Without him, this thesis would not exist.

I express my deep gratitude to **Mrs. E. Rajeshwari**, project coordinator for her immense support in the completion of the project.

I would also be thankful to my HOD **Dr. V. Mahender** for his support, direction, incentives, suggestions, and insight throughout this project.

I am also grateful to **Prof. P. Malla Reddy,** Principal of KU Engineering College for his valuable guidance during my project.

I would like to express my deep gratitude towards **my teaching and non-teaching staff** for giving their valuable suggestions and cooperation in doing my project.

I would like to thank my friends for their help and constructive criticism during my project period.

Finally, I am very much indebted to my parents for their moral support and encouragement to achieve higher goals. I have no words to express my gratitude and still I am very thankful to my parents who have shown me this world and for every support they gave me.

ABSTRACT

Most users provide their reviews on the assorted products on the Amazon website. The reviews provided by users are most often compact. Hence it becomes a loaded source for sentiment analysis. Sentiment Analysis is a substantially employed method for locating and obtaining the appropriate polarity of text sources. This project centers on a contrastive study of machine learning techniques for classifying the emotions of the considered product reviews dataset into Positive polarity, Neutral polarity, and Negative polarity, segmenting into Product, Delivery, and Packaging categories. This can be helpful for consumers who want to look at the reviews of products before purchase and for companies who wish to look at the public's reaction to their products. In this project, we correlate the performance of algorithms that support multi-class multi-output classifications, the accuracy of classifying the sentiment of an unknown review, insight on sentiment analysis, segmentation, and therefore the comparison of the performance of the considered algorithms for the classification of the sentiments supported by several performance metrics.

Keywords: Amazon Reviews, Segmentation, Opinion Mining, Sentiment Analysis, Machine Learning Classification.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The recent breakthrough in Natural Language Processing (NLP) and text mining, along with advancements in information technologies, have led to the development of many new applications [1]. New methods in artificial intelligence and an increase in the amount and variety of textual data produced on a daily basis allow for a great deal of research to be carried out on real-life problems.

Text classification is one of the most fundamental topics in NLP and text mining. The text classification problem can be defined as associating relevant text with pre-existing labels. The structure of datasets plays an important role in such labeling. Depending on the status of the problem, each text can be represented by one or more labels. This has led to variations in text classification practices. Text classification types resulting from this diversity of approaches are shown in Table 1.1.

Table 1.1: Text Classification Types

Classification Type	Task	Labels
Binary	Spam Filter	Ham, Spam
Multi-Class	Sentiment Analysis	Positive, Neutral, Negative
Multi-Label	Toxic Comment Detection	Threat, Toxic, Obscene, Insult

Binary classification is a popular and relatively simple type of text classification based on structure. Each text is classified such that it is represented by only one of two labels. Fake news detection [2], spam e-mail detection [3], spam review detection [4], and authorship verification [5] are examples of binary text classification applications. Unlike binary classification, in multi-class multi-output text classification there are more than two labels. What multi-class text classification has in common with binary text classification is that in both models each text is represented by only one label.

Sentiment analysis [7], topic modeling [8] and synonym extraction [9] are examples of multi-class text classification tasks. Although both binary and multi-class text classification lead to successful results, they do not produce satisfactory outputs in situations where individuals' opinions are needed, such as concerning products and events, as the language we use and the expressions we produce contain more complex meanings in these contexts. Restricting textual expression to a single label often prevents the extraction of more detailed information from text, even though this is possible.

For this reason, with the developments in information technologies, multi-label data are needed in order to meet the expectations of individuals. Multi-label text classification methods are employed to analyze multi-label data. In other words, in multi-label text classification, each text can have either a single label or more than one label.

Developments in information technologies have affected both artificial intelligence applications and human behaviors, lives, and expectations. In particular, in the so-called social digital life caused by digitalization, and in the circumstances of the COVID-19 pandemic, people have started to use the internet intensively to work, shop, have fun, and learn; in short, people have started to use it in their daily routines. This influence can best be witnessed in electronic commerce (E-Commerce). Recent statistics [10] indicate that 93.5 percent of internet users have purchased products online. In the US, 41 percent of customers receive one or two packages from Amazon per week, and that percentage rises to 50 for customers aged 18–25 and 57 for customers aged 26–35 [10]. It is estimated [11] that 95% of all purchases will be made through e-commerce by the year 2040. Product reviews have a high impact on customers' online purchases; 55% of online customers tell friends and family when they are not satisfied with a product or company [10].

Moreover, 90% of customers read online reviews before making a purchase. As a result, the amount of data collected daily and its influence on customers have attracted the attention of researchers. Thus, many studies have been conducted on e-commerce customer reviews. Table 1.2 presents e-commerce customer review classification methods and possible labels.

Table 1.2: E-Commerce Review Classification Methods and Possible Labels.

Binary Classification	Multi-Class Classification	Multi-Label Classification
Spam Review	Positive Review	Fabric Quality
Normal Review	Neutral Review	Express Delivery
	Negative Review	Order Size

As stated in [12], most studies performed in the literature are based on polarity analysis, and multi-label review analysis has not been performed. Unlike the traditional classification techniques shown in Table 2, the model we propose here aims to identify the various labels present in reviews.

Multi-Class, Multi-Label examinations of these data are important because with detailed analyses useful findings can be discovered for both people who buy products and for companies that want to improve their customer relationship management via customer reviews.

For example, say a person is shopping for a product that they want to buy very urgently. In this process, their primary request about the product is for it to be shipped immediately by the seller. It is an advantage for such a person to read reviews by classifying them in line with their priorities.

For this reason, it is important to analyze customer reviews both sentimentally and qualitatively. For these reasons, and due to the absence of such studies on this subject in the literature, we are motivated to classify e-commerce customer reviews through a multi-class multi-output approach.

1.1 OBJECTIVE & SCOPE OF THE PROJECT

1.1.1 SCOPE OF THE PROJECT

The project addresses e-commerce customer review analysis for consumers, making three main contributions.

- Customer reviews are analyzed in an aspect-based manner rather than a polarity-oriented manner to determine the detailed opinions of customers about products.
- Create a new multi-label e-commerce customer review dataset. This data set can allow researchers to compare customer habits. Which includes the intention of the reviews such as delivery, packaging, and product review.
- Carry out a multi-label customer review analysis with various algorithms and diverse measurement techniques and produce a multi-output classifier helping in understanding the text of the reviews in a much better manner.

1.1.2. OBJECTIVE

- To determine the various output labels for review segmentation as product, packaging, and delivery reviews.
- To classify the sentiment of the reviews into various sentiments such as Positive, Negative, and Neutral Reviews.
- To develop a prediction system that can be used to classify an input customer review into various aspects as stated above to further understand the customer intentions on the reviews provided by them.

1.2 ORGANISATION OF PROJECT REPORT

The report has been divided into five major chapters. The First chapter is the introduction to the project's nature, giving a brief overview of the project and its scope of the project.

The Second chapter is about the comparison of the view of the existing system and ideas to better that with the proposed system has been drawn.

The Third chapter deals with the approach to the project, which comprises the design specifications in the form of class diagrams and hardware, and software requirements.

Chapter Four gives brief details about the project processes with the help of flowcharts. The architecture of the system is being discussed along with the modules present in the project.

Chapter Five, the last chapter of this project report comprises testing details of the project done by applying unit tests on the modules. All the parts and sub-parts have been numbered sequentially for better understanding and easy identification.

2. LITERATURE SURVEY

2.1 EXISTING SYSTEMS

Sentiment analysis or opinion mining is the systematic examination of people's attitudes, views, and feelings regarding a given entity [13]. Sentiment analysis of customer reviews has been performed in many studies. Almost all of these studies have focused on polarity analysis.

- Muslim [14] aimed to improve Support Vector Machine (SVM) accuracy for classifying e-commerce customer review datasets using grid search, with uni-gram used for feature extraction. They used datasets consisting of Amazon reviews and Lazada reviews labeled as positive or negative. Their experimental results showed that applying unigram and grid search on the support vector machine (SVM) algorithm could improve the accuracy of Amazon reviews by 26.4% to 80.8% and that of Lazada reviews by 4.26% to 90.13%.

- Vanaja et al. [16] performed aspect-level sentiment analysis on Amazon customer review data. They analyzed whether the reviews were positive, negative, or neutral. They stated that they found 0.9023 accuracy using naive Bayes in their comparative analysis.

- Jabbar et al. [17] presented a real-time sentiment analysis of the product reviews of e-commerce applications. They used SVM to design a model for sentiment analysis of collected review data from Amazon. They labeled reviews as positive or negative. They obtained an F1 score of 0.9354 for the reviews' sentiment analysis using SVM.

- Tripathi et al. [19] examined the textual content of reviews on e-commerce websites with different helpfulness votes to further classify a new review by collecting reviews from e-commerce websites. They stated that the best accuracy was 0.945, obtained with a random forest classifier.

- Guan et al. [22] proposed a deep learning framework for review sentiment analysis. They collected reviews from Amazon and classified sentiment as positive or negative. Their deep learning framework achieved 0.877 accuracy on review sentiment analysis.

- Zhang et al. [12] proposed a directed weighted multi-classification model for e-commerce reviews. They used 10,000 reviews from Amazon Review Data. They used multi-label classification for review sentiment. Their directed weighted model achieved 0.8 average recall.

- Gu et al. [24] proposed a novel sentiment analysis model called MBGCV. In their study, they used 31,107 reviews labeled as positive or negative. Their proposed model achieved 0.94 accuracy on review sentiment analysis.

- Bilen et al. [25] performed LSTM network-based sentiment analysis on Turkish customer reviews. They used two different datasets for sentiment analysis. They collected a new corpus of approximately 7000 reviews for sentiment analysis of Turkish consumer preferences. They classified the data they collected as either positive or negative using an LSTM-based model, finding 0.905 accuracy for binary sentiment analysis.

- Acikalin et al. [27] performed sentiment analysis on Turkish movie and hotel reviews with positive and negative labels using BERT. They stated that the best result they found in their study was 93.3%.

- Santur [28] performed sentiment analysis on Turkish e-commerce customer reviews using Gated Recurrent Unit, classified the reviews as positive, negative, or neutral, and stated their best result as 0.95 accuracy. Ozyurt et al. [29] performed aspect-based sentiment analysis on Turkish reviews using LDA. They collected 1292 user reviews about smartphones and defined nine aspects for smartphones. They found an F-score of 82.39% in their results.

2.1.1 DRAWBACKS IN THE EXISTING SYSTEM

- The above-mentioned classifiers or regressions are all based on either multi-label or multi-class inputs, produce a single output.

- They depend on various labeled data to form predictions thereby increasing the need for large data sets.

- Most of the predictions performed are on minimal data sets containing few reviews which do not contribute to the project.

2.2 MOTIVATION FOR THE PROPOSED SYSTEM

Majority of the e-commerce users usually depend on the reviews stated by various consumers throughout the world for their purchase decisions, so it is quite important to understand the intent of the reviews in if its sarcasm or helpful or negative review.

Most of the reviews are either fake or wrongly segregated which highly impacts both the consumer and the businesses. Thus, the need for proper regulation, segmentation, and classification of the reviews has aroused to help both consumers and business owners in attaining good products and sales easing the process of online commercial business and time for users.

Sentiment classification and Review segmentation aim to determine the overall intention of a written text review which can be of admiration or criticism type. This can be achieved by using Natural Language Processing algorithms. So, the problem that is going to be investigated in the project is to find which machine learning approach performs better in terms of Sentiment classification and Review segmentation on Amazon books products reviews.

2.3 PROPOSED SYSTEM

In our proposed approach we have used Different embedding methods, both frequency-based and prediction-based, and different classification methods are employed throughout the project.

The embedding methods used in this project consist of frequency-based Term Frequency-Inverse Document Frequency (TF-IDF) and prediction-based Global Vectors for Word Representation (GloVe). Afterward, Multiclass-Multioutput Classifiers such as Random Forest (RF), Support Vector Classification (SVC), k-Nearest Neighbor, Decision Tree Classifier, and AdaBoost Classifier are used.

1. We scrape amazon product reviews from websites or datasets.
2. Filter the dataset according to feature requirements and create a new dataset that has attributes according to the analysis to be done.
3. Perform Pre-Processing on the dataset.
4. Split the data into training and testing.
5. Train the model with training data then analyze the testing dataset over the classification algorithm.
6. Finally, you will get results as accuracy metrics.

2.3.1 SYSTEM ARCHITECTURE

A graphical abstraction of the proposed model is shown in Figure 2.1. The proposed model includes data collection, data preprocessing, feature extraction, and testing. The whole procedure is explained in the following subsections.

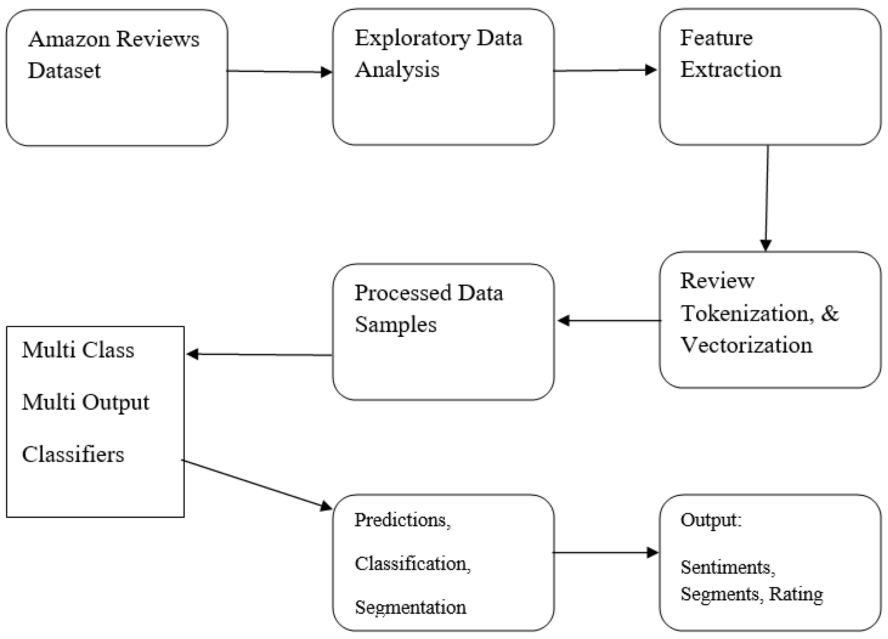

Amazon Reviews Dataset
Exploratory Data Analysis
Feature Extraction
Review Tokenization, & Vectorization
Processed Data Samples
Multi Class Multi Output Classifiers
Predictions, Classification, Segmentation
Output: Sentiments, Segments, Rating

Figure 2.1: Proposed System Graphical Illustration

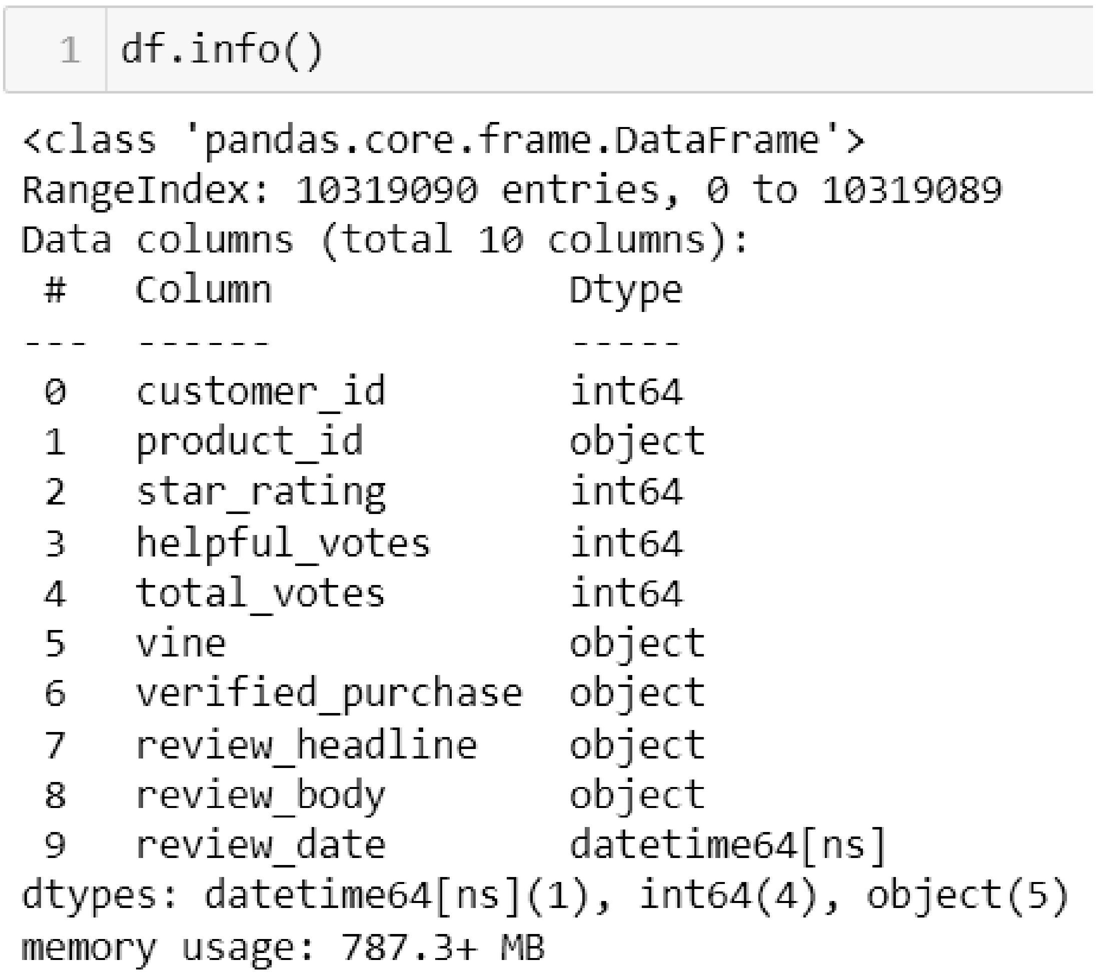
2.3.2 AMAZON REVIEWS DATASET

The following table shows the various data columns present in the dataset

Table 2.1: Amazon Customer Review Data Columns

marketplace	2 letter country code of the marketplace where the review was written.
customer_id	Random identifier that can be used to aggregate reviews written by a single author.
review_id	The unique ID of the review.
product_id	The unique Product ID the review pertains to.
product_parent	Random identifier that can be used to aggregate reviews for the same product.
product_title	Title of the product.
product_category	Broad product category that can be used to group reviews (also used to group the dataset into coherent parts).
star_rating	The 1-5 star rating of the review.
helpful_votes	Number of helpful votes.
total_votes	Number of total votes the review received.
vine	Review was written as part of the Vine program.
verified_purchase	The review is on a verified purchase.
review_headline	The title of the review.
review_body	The review text.
review_date	The date the review was written.

The data format is Tab ('\t') separated text file, without quote or escape characters. The first line in each file is the header; 1 line corresponds to 1 record. Figure 2.2 represents the data set columns and rows information obtained from the jupyter notebook implementation.


```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10319090 entries, 0 to 10319089
Data columns (total 10 columns):
 #   Column             Dtype
---  ------             -----
 0   customer_id        int64
 1   product_id         object
 2   star_rating        int64
 3   helpful_votes      int64
 4   total_votes        int64
 5   vine               object
 6   verified_purchase  object
 7   review_headline    object
 8   review_body        object
 9   review_date        datetime64[ns]
dtypes: datetime64[ns](1), int64(4), object(5)
memory usage: 787.3+ MB
```

Figure 2.2: Description of Amazon Reviews Data on Jupyter Notebook

The various modules involved in the proposed project are as follows:

1. Data Collection
2. Data Pre-Processing
3. Feature Extraction
4. ML & Natural Language Processing
5. Evaluation Model

Data Collection

The dataset used consists of more than 10,000,000 reviews from amazon's books category available at https://s3.amazonaws.com/amazon-reviews-pds/tsv/index.txt. It is in the format shown in above Figure 2.2.

Data Pre-Processing

In Natural Language Processing (NLP), most of the text and documents contain many words that are redundant for text classification, such as stopwords, miss-spellings, slang, etc. In many algorithms like statistical and probabilistic learning methods, noise and unnecessary features can negatively affect the overall performance. So, the elimination of these features is extremely important.

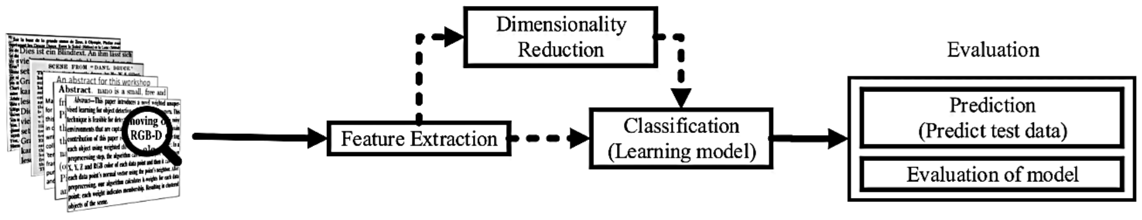


Figure 2.3: Graphical Illustration of Text Processing

Organize your selected data by formatting, cleaning, and sampling from it.

Three common data pre-processing steps are:

1. Formatting
2. Cleaning
3. Sampling

Formatting: The data you have selected may not be in a format that is suitable for you to work with. The data may be in a relational database, and you would like it in a flat file, or the data may be in a proprietary file format, and you would like it in a relational database or a text file.

Cleaning: Cleaning data is the removal or fixing of missing data. There may be data instances that are incomplete and do not carry the data you believe you need to address the problem. These instances may need to be removed. Additionally, there may be sensitive information in some of the attributes and these attributes may need to be anonymized or removed from the data entirely.

Sampling: There may be far more selected data available than you need to work with. More data can result in much longer running times for algorithms and larger computational and memory requirements. You can take a smaller representative sample of the selected data that may be much faster for exploring and prototyping solutions before considering the whole dataset.

2.3.3 FEATURE EXTRACTION

Next thing is to do Feature extraction is an attribute reduction process. Unlike feature selection, which ranks the existing attributes according to their predictive significance, feature extraction transforms the attributes. The transformed attributes, or features, are linear combinations of the original attributes. Finally, our models are trained using the Classifier algorithm. We use classify module on Natural Language Toolkit library on Python.

We use the labeled dataset gathered. The rest of our labeled data will be used to evaluate the models. Some machine learning algorithms were used to classify pre-processed data. The chosen classifiers were Random Forest. These algorithms are very popular in text classification tasks.

2.3.4 NATURAL LANGUAGE PROCESSING

- Remove URLs and email addresses from every single sample — because they won't add meaningful value.

- Remove punctuation signs — otherwise, your model won't understand that "good!" and "good" are meaning the same thing.

- Lowercase all text — because you want to make the input text as generic as possible and avoid that, for example, a "Good" which is at the beginning of a phrase to be understood differently than the "good" in another sample.

- Remove stop-words — stop-words refer to the most common words in a language, such as "I", "have", "are" and so on.

- Stemming/Lemmatization: Stemming and lemmatization are very similar tasks, both look forward to extracting the root words from every word of a sentence of the corpus data.

- Transform dataset (text) into numeric tensors — usually referred to as vectorization.

2.3.5 FEATURE SELECTION AND PRE-PROCESSING

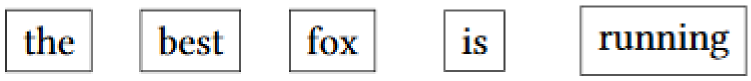
Feature selection and preprocessing are significant tasks in Artificial Intelligence and mainly represent the data preparation step in the CRISP-DM. Especially in NLP, this task does have a tremendous impact on the success of text analysis. This is mostly caused by the unstructured and arbitrary nature of text data. Furthermore, machines need structure and numerical data.

A couple of approaches for this transformation task, e.g. word embedding or the vector space model, exist. This section's scope lies in the theoretical foundation of different preprocessing and feature selection techniques. This section will be accompanied by the English phrase 'the best fox is running ' as an example to illustrate the application of preprocessing. Nevertheless, every routine should be used with care. It is not always the case that a reasonably good preprocessing method leads to better results in every application. The so-called no free lunch theorem makes it necessary to evaluate every suggested method separately in the practical part of this thesis.

Tokenization

For processing a written natural language it is inevitable to split texts into smaller units, which are called tokens. Computers need to distinguish single entities of a text and tokenization is used to create them. Usually, tokens represent simple words, which are the smallest independent units of natural language. Furthermore, token scan consists of idioms or hyphens, e.g., “user-generated”. Tokenization breaks running texts into short text entities and is the very first task in any text preprocessing cycle. Besides the partition of small units, whole sentences can also be the output of a tokenizer.

A simple word tokenizer can be realized in many languages by splitting the text at the occurrences of space symbols. This simple baseline approach does have a couple of downsides, due to the lack of identifying words that semantically belong together. However, a simple tokenizer divides the phrase, which was introduced above, into the following five tokens:



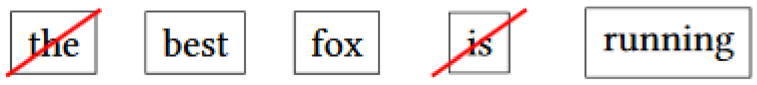
Using tokens, so-called n-grams can be created, which indicate a token set with the length of n. ‘Gramma’ is the Greek word for a letter or token. When talking about a set of n letters in words, it is about character n-grams.

Stop-Word Removal

A very important approach to reduce the huge raw input space in NLP is stop word removal. Most languages have specific words, which do appear more often than others or do not include much information about the content of the text, e.g., auxiliary verbs or articles. Due to this, it often makes sense to exclude these so-called stop words in further analysis. In English such words could be “the”, “a” or “an” and for German typical stop words are the articles “der”, “die” and “das”.

The elimination could be done by checking the words against a standardized stop word list. These lists are available in the literature and are often implemented in different software packages.

In our example, “the” and “is” eliminated. Stop word removal should be used with care, especially in sentiment analysis, which attempts to predict a positive or negative intention of a text. The removal would exclude words that can change a whole statement, such as “not” or “none”.



Stemming

Besides stop-word elimination, stemming is a useful technique to map words to their word stems and further reduce the input dimension. This helps to extract the real meaning of a text and makes the unstructured data better accessible to a machine. The first stemming algorithm based on deleting longest suffixes and spelling exceptions was developed in 1968.

By now, the porter stemming algorithm is a state-of-the-art approach and strips suffixes from words to retain the word stem. While this method performs well in English, there are some drawbacks for the German language, since German words are not usually built by adding suffixes.

However, there is a German equivalent based on Porter's idea and the string processing language Snowball. By using the English Porter Stemmer, the words "best", "fox" and "running" are assigned to the following words:

Best → best, fox →fox, running →run

Lemmatization

Lemmatization is the process of mapping every word in a text to its dictionary type or intended originating structure. Verbs are transformed to their infinite form, a noun is reconstructed to its singular representation, and adverbs or adjectives anticipate their positive format.

The method is based on morphological analysis and often uses a dictionary, for example, WordNet, where the lemma of every modified word form could be retrieved. This preprocessing step is similar to stemming and reduces the input space, by mapping different word forms to their common representation. Natural Language processing lemmatization is supported by dictionary entries; it can map "best" to its lemma "good".

Best → good, fox→ fox, running→ run

Vector Space Model

Besides, preprocessing the words themselves, their representations have to be changed into a machine-readable format. Meanwhile, a couple of different approaches have been developed to transform texts into different kinds of numerical representations. Some of them only represent statistics of a word, such as the one-hot-encoding, and other formats also include the word's context, e.g., word2vec.

The Vector Space Model is an approach that transforms a text into one vector. It is based on the one-hot encoding of words. Given a set of textual documents (corpus), it is possible to create a vocabulary with the length of N. The one-hot-encoded word vector represents a word by 1 at the corresponding vocabulary entry.

2.3.6 MACHINE LEARNING ALGORITHMS

Machine learning algorithms are classified into 4 types:

- Supervised
- Unsupervised Learning
- Semi-supervised Learning
- Reinforcement Learning

However, these four types of ml algorithms are further classified into more types as shown below in Figure 2.4.

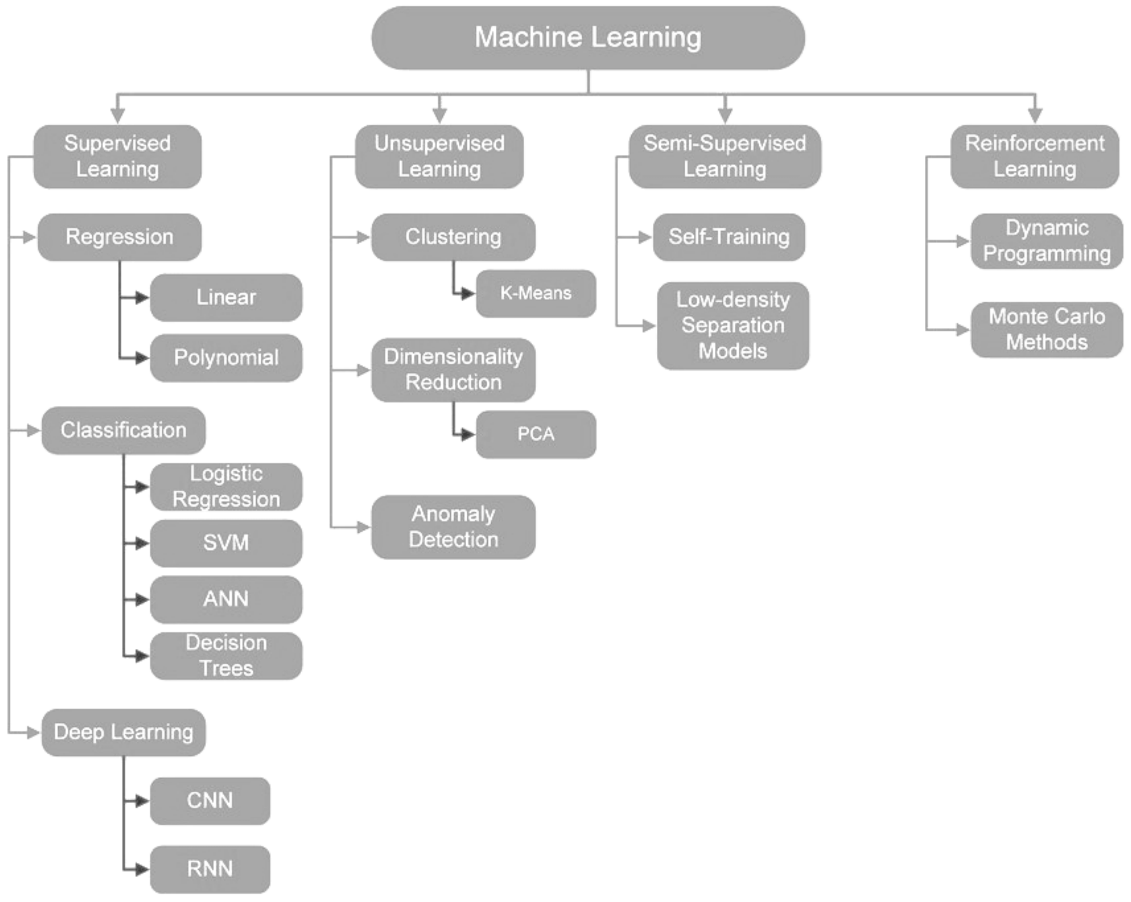


Figure 2.4: Machine Learning Algorithms

Supervised Learning

This algorithm consists of a target/outcome variable (or dependent variable) which is to be predicted from a given set of predictors (independent variables). Using this set of variables, we generate a function that map inputs to desired outputs. The training process continues until the model achieves the desired level of accuracy on the training data. Examples of Supervised Learning: Regression, Decision Tree, Random Forest, KNN, Logistic Regression, etc.

Unsupervised Learning

In this algorithm, we do not have any target or outcome variable to predict or estimate. It is used for clustering populations in different groups, which is widely used for segmenting customers into different groups for specific interventions. Examples of Unsupervised Learning: Apriori algorithm, K-means.

Reinforcement Learning

Using this algorithm, the machine is trained to make specific decisions. It works this way: the machine is exposed to an environment where it trains itself continually using trial and error. This machine learns from experience and tries to capture the best possible knowledge to make accurate business decisions. Example of Reinforcement Learning: Markov Decision Process.

2.3.7 TEXT CLASSIFICATION ALGORITHMS

Support Vector Machines (SVM)

Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression purposes. For the text classification process, the SVM algorithm categorizes the classes of a given dataset by determining the best hyperplane or boundary line that divides the given text data into predefined groups.

The SVM algorithm creates multiple hyperplanes, but the objective is to find the best hyperplane that accurately divides both classes. The best hyperplane is selected by selecting the hyperplane with the maximum distance from the data points of both classes. The vectors or data points nearer to the hyperplane are called support vectors, which highly influence the position and distance of the optimal hyperplane. For instance, using SVM, you can create a classifier for detecting hate speech. You will be required to label or assign two sets of words to various sentences in the dataset that would represent hate speech or neutral speech.

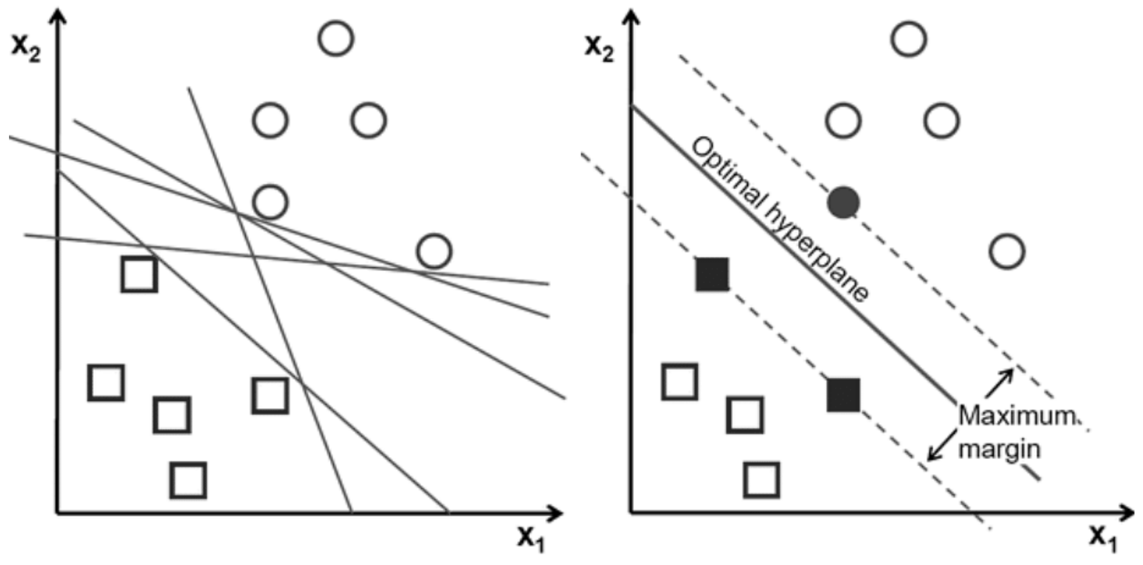


Figure 2.5: Support Vector Machine (SVM) illustration

Consider the above images, where the blue circle represents hate speech, and the red box represents neutral speech. The first image shows all possible hyperplanes that separate two classes of data, such as hate speech and neutral speech, and the second image shows the optimal and best hyperplane that classifies hate speech and neutral speech by the highest distance or maximum margin from the data points.

By selecting the best possible hyperplane, the SVM model is trained to classify hate and neutral speech. Now, whenever the new set of data is passed through this machine learning model, it matches the new dataset with the previously trained set of data, and based on that, it can classify or categorize whether the speech is hateful or neutral.

Adaboost Classifier

Ada-boost or Adaptive Boosting is one of the ensemble boosting classifiers proposed by Yoav Freund and Robert Schapire in 1996. It combines multiple classifiers to increase the accuracy of classifiers. AdaBoost is an iterative ensemble method.

AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. The basic concept behind Adaboost is to set the weights of classifiers and train the data sample in each iteration such that it ensures accurate predictions of unusual observations. Any machine learning algorithm can be used as a base classifier if it accepts weights on the training set. Adaboost should meet two conditions:

- The classifier should be trained interactively on various weighed training examples.
- In each iteration, it tries to provide an excellent fit for these examples by minimizing training errors.

AdaBoost algorithm work in the following steps

1. Initially, Adaboost selects a training subset randomly.
2. It iteratively trains the AdaBoost machine learning model by selecting the training set based on the accurate prediction of the last training.
3. It assigns a higher weight to wrong-classified observations so that in the next iteration these observations will get a high probability for classification.
4. Also, it assigns the weight to the trained classifier in each iteration according to the accuracy of the classifier. The more accurate classifier will get a high weight.
5. This process iterates until the complete training data fits without any error or until reached the specified maximum number of estimators.
6. To classify, perform a "vote" across all of the learning algorithms you built.

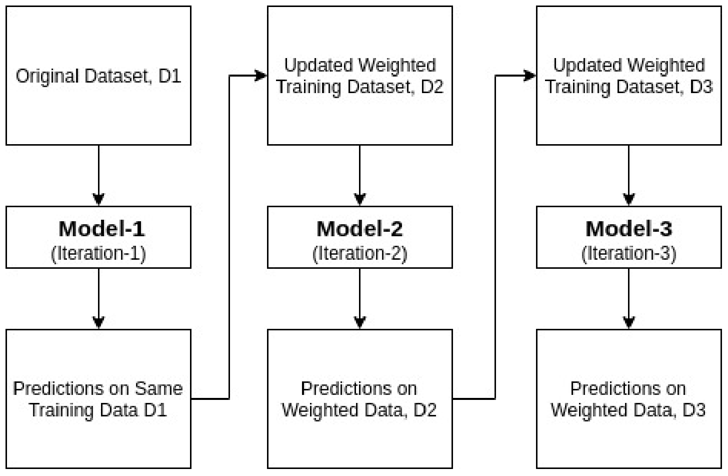

Original Dataset, D1
Model-1 (Iteration-1)
Predictions on Same Training Data D1
Updated Weighted Training Dataset, D2
Model-2 (Iteration-2)
Predictions on Weighted Data, D2
Updated Weighted Training Dataset, D3
Model-3 (Iteration-3)
Predictions on Weighted Data, D3

Figure 2.6: Ada-boost or Adaptive Boosting illustration

Decision Tree

A Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.

It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

The decisions or the test are performed based on features of the given dataset.

It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

It is called a decision tree because similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

To build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.

A decision tree simply asks a question and based on the answer (Yes/No), it further split the tree into subtrees. Figure 2.7 explains the general structure of a decision tree.

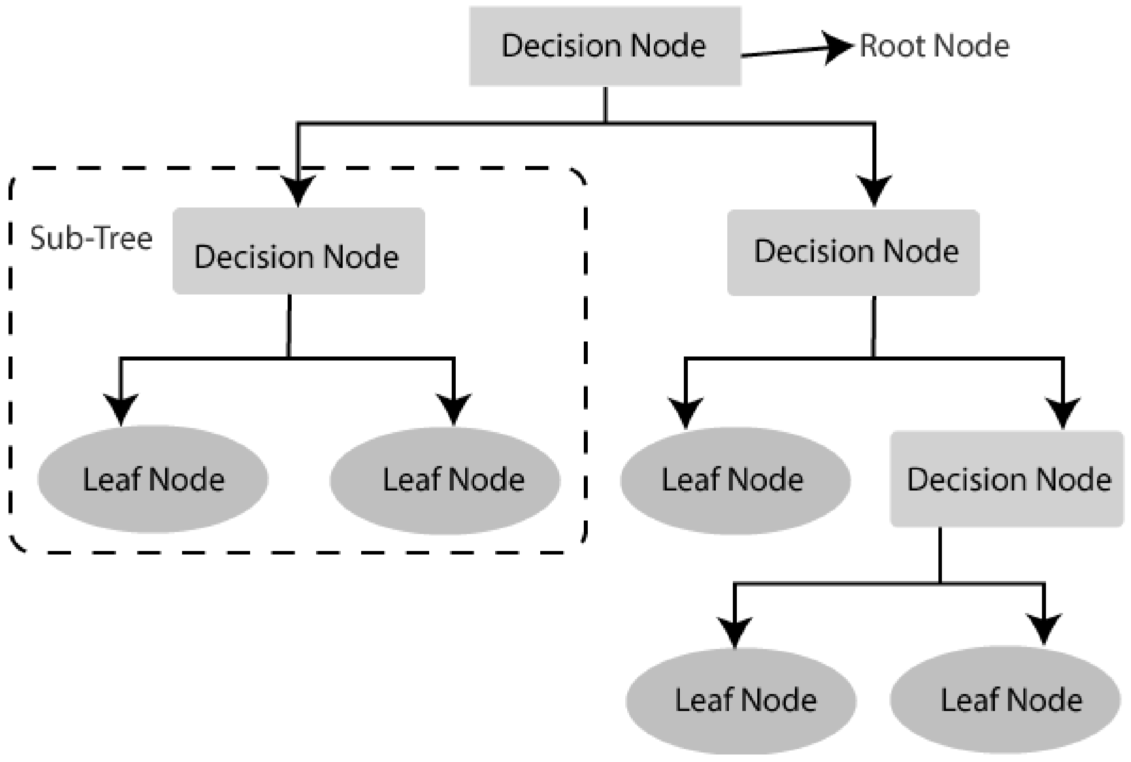


Figure 2.7: General Structure of a Decision Tree

Random Forest Algorithm

A Random Forest Algorithm is a supervised machine learning algorithm that is extremely popular and is used for Classification and Regression problems in Machine Learning.

We know that a forest comprises numerous trees, and the more trees more it will be robust. Similarly, the greater the number of trees in a Random Forest Algorithm, the higher its accuracy and problem-solving ability.

Random Forest is a classifier that contains several decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. It is based on the concept of ensemble learning.

Working of Random Forest Algorithm

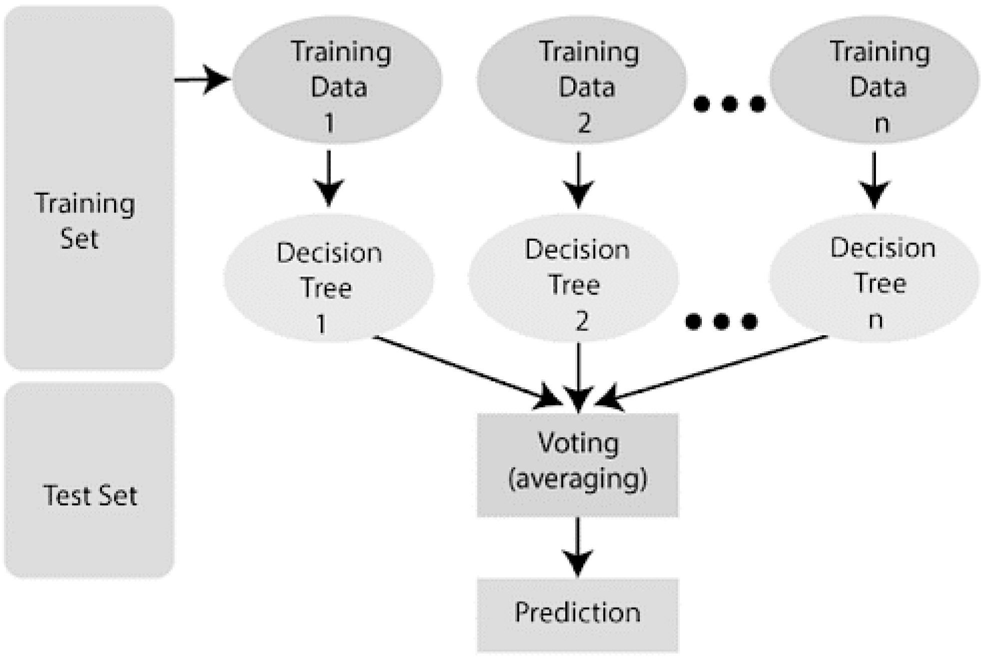


Figure 2.8: Random Forest Algorithm illustration

The following steps explain the working Random Forest Algorithm:

Step 1: Select random samples from a given data or training set.

Step 2: This algorithm will construct a decision tree for every training data.

Step 3: Voting will take place by averaging the decision tree.

Step 4: Finally, select the most voted prediction result as the final prediction result.

This combination of multiple models is called Ensemble. Ensemble uses two methods:

Bagging

Creating a different training subset from sample training data with replacement is called Bagging. The final output is based on majority voting. Example: Random Forest.

Boosting

Combing weak learners into strong learners by creating sequential models such that the final model has the highest accuracy is called Boosting. Example: ADA BOOST, XG BOOST.

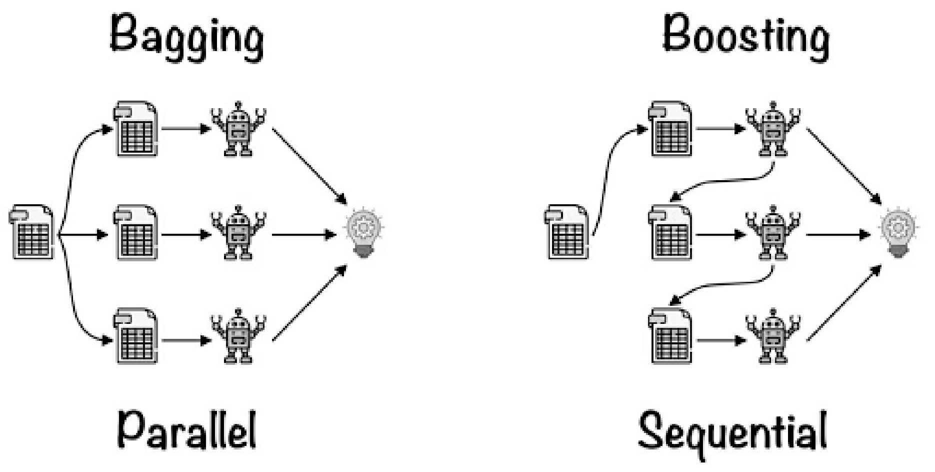


Figure 2.9: Bagging vs Boosting illustration

2.3.8 Metrics and Evaluation

Cross-validation is a common method to evaluate the performance of a text classifier. It works by splitting the training dataset into random, equal-length example sets (e.g., 4 sets with 25% of the data). For each set, a text classifier is trained with the remaining samples (e.g., 75% of the samples). Next, the classifiers make predictions on their respective sets, and the results are compared against the human-annotated tags. This will determine when a prediction was right (true positives and true negatives) and when it made a mistake (false positives, false negatives). With these results, you can build performance metrics that are useful for a quick assessment of how well a classifier works:

- **Accuracy:** the percentage of texts that were categorized with the correct tag.
- **Precision:** the percentage of examples the classifier got right out of the total number of examples that it predicted for a given tag.
- **Recall:** the percentage of examples the classifier predicted for a given tag out of the total number of examples it should have predicted for that given tag.
- **F1 Score:** the harmonic mean of precision and recall.

3 MULTI-CLASS MULTI-OUTPUT CLASSIFICATION

Multi-output learning subsumes many learning problems in multiple disciplines and deals with complex decision-making in many real-world applications. It has a multivariate nature, and the multiple outputs may have complex interactions, architected to be handled by structured inference. The output values have diverse data types, depending on the type of ML problem. For example,

- 0/1-based Binary output values can refer to the multi-label classification problem.
- Nominal output values to the multi-dimensional classification problem
- Ordinal output values to label ranking problem.
- Real-valued outputs to a multi-target regression problem.

Multi-output classification is a type of machine learning that predicts multiple outputs simultaneously. In multi-output classification, the model will give two or more outputs after making any prediction. In other types of classifications, the model usually predicts only a single output.

3.1 USE CASES OF MULTI-OUTPUT LEARNING

Multi-class Classification

Multi-class classification can be categorized as a traditional single-output learning paradigm when the output class is represented by integer encoding. It can also be extended to a multi-output learning scenario if each output class is represented by the one-hot vector.

Fine-grained Classification

In this type of classification, though the vector representation is the same as fine-grained classification outputs to the multi-class classification outputs, the internal structures of the vectors are different. Labels under the same parent tend to have a closer relationship than the ones under different parents in the label hierarchy.

Multi-task Learning

Multi-task learning aims at learning multiple related tasks simultaneously, where each task outputs one single label and learning multiple tasks is similar to learning multiple outputs. It leverages the relatedness between tasks to improve the performance of learning models.

The major difference between multi-task learning and multi-output learning is that different tasks might be trained on different training sets or features in multi-task learning, while output variables usually share the same training data or features in multi-output learning.

In Multi-output pattern recognition problems, each instance in the dataset has two or more output values (nominal or real-valued) i.e., the output value is a vector rather than a scalar. They are solved by any of the following methods:

- By transforming the multi-label (or multi-output) into multiple single-output problems.
- By adopting a pattern recognition algorithm so that it directly handles multi-output data.

However, there are certain pros and cons to the above-mentioned approaches:

- The first approach of training an inductive classifier or regression model can be a time-consuming task particularly so when training data sets are very large.
- When multiple models need to be trained using the same input data but with different output data, the training time is unusually too high making it unsuitable for large datasets. Consequently, this also impacts the processing requirements.
- The second adaption approach enables to create a model that simultaneously predicts a set of two or more classification labels, regression values, or even joint classification-regression outputs from only a single training iteration.
- If the prediction tasks are related (i.e., there is a correlation or covariance between output values), training a coherent multi-output model can potentially bring benefits in the form of increased predictive performance compared to training multiple disjoint models.

3.2 MULTI-CLASS, MULTI-LABEL, MULTI-OUTPUT CLASSIFICATION

Multi-class classification means a classification task with more than two classes; e.g., classify a set of images of fruits which may be oranges, apples, or pears. Multiclass classification assumes that each sample is assigned one and only one label: a fruit can be either an apple or a pear but not both at the same time.

Multi-label classification assigns to each sample a set of target labels. This can be thought of as predicting properties of a data point that are not mutually exclusive, such as topics that are relevant to a document. A text might be about any religion, politics, finance, or education at the same time or none of these.

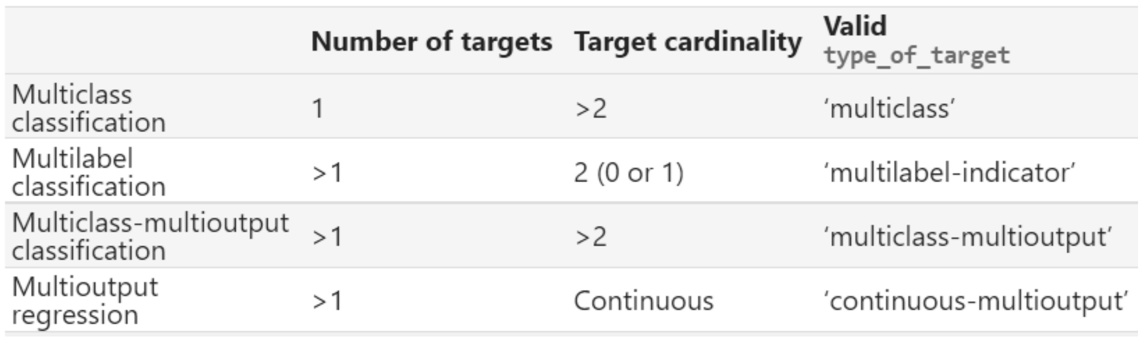
Multi-output regression assigns each sample a set of target values. This can be thought of as predicting several properties for each data point, such as wind direction and magnitude at a certain location.

Multi-output Multi-class classification and **Multi-task classification** mean that a single estimator has to handle several joint classification tasks. This is a generalization of the multi-label classification task, where the set of classification problems is restricted to binary classification, and of the multi-class classification task. The output format is a 2d NumPy array or sparse matrix.

The set of labels can be different for each output variable. For instance, a sample could be assigned "pear" for an output variable that takes possible values in a finite set of species such as "pear", "apple", "orange" and "green" for a second output variable that takes possible values in a finite set of colors such as "green", "red", "orange", "yellow" ...

This means that any classifiers handling multi-output multiclass or multi-task classification task supports the multi-label classification task as a special case.

Multi-task classification is similar to the multi-output classification task with different model formulations. For more information, see Figures 3.1, and 3.2.



	Number of targets	Target cardinality	Valid `type_of_target`
Multiclass classification	1	>2	'multiclass'
Multilabel classification	>1	2 (0 or 1)	'multilabel-indicator'
Multiclass-multioutput classification	>1	>2	'multiclass-multioutput'
Multioutput regression	>1	Continuous	'continuous-multioutput'

Figure 3.1: Difference between Problem Types

Figure 3.2 below demonstrates the problem types that each module is responsible for, and the corresponding meta-estimators that each module provides.

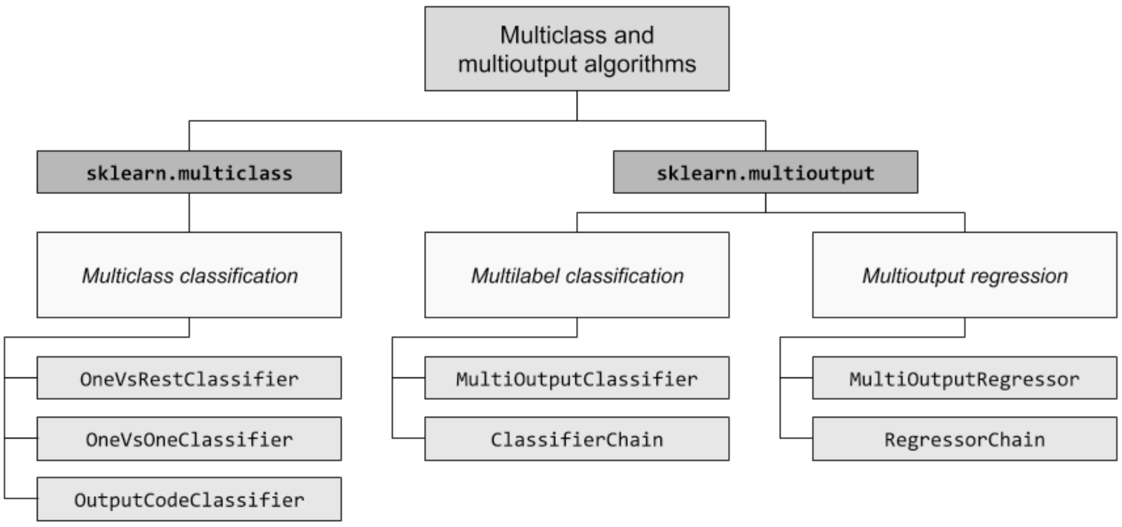


Figure 3.2: Chart Demonstrating Problem Types and their Modules

3.2.1 MULTI-OUTPUT VS MULTI-LABEL CLASSIFICATION

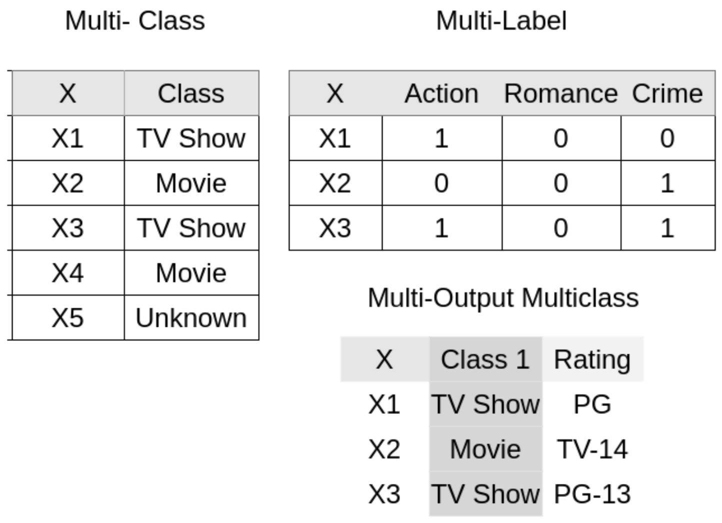
In multi-output classification, the goal is to learn a classification rule whose output is a set, or vector, of labels i.e., y1 belongs to Y1, y2 belongs to Y2, yn belongs to Yn and v, the vector is composed of y1, y2, y3, …yn.

Multi-label classification involves classifying instances into several labels that share semantics, for example, the problem of classifying songs according to their genre — it is possible to classify a dance as either ballet or traditional, but also possible to classify a dance as both (i.e.,' ballet traditional'). Here, pop and rock share semantics: they both relate to the songs' genre and are thus two different values of the same label. There is also no a priori knowledge regarding the size of the output — it is very possible that a song cannot be classified as any previously known genre, or that it is best classified as several different genres.

The problem of multi-output classification is effectively the opposite of multi-label classification, the output values do not share semantics, but the number of outputs is known as priori. In a classification problem where the goal is to simultaneously predict temperature (low, medium, or high) and pressure (low, medium, or high) inside a pressure cooker. In this case, the model is expected to output exactly two values, one value for the temperature label and another for the pressure label.

The machine learning task of solving a multi-output problem thus involves building a predictive model that simultaneously outputs a set of (two or more) labels that measure different concepts, essentially two or more separate (although related) classification problems are solved concurrently within the same model.

A multi-output classification is a multi-task classification that illustrates the fact that a multi-output classification problem is effectively equivalent to multiple simultaneous (multi-tasked) single-label classification problems. Multi-label problems can be transformed into multi-output problems, the opposite is not necessarily true. Figure 3.3 represents the difference between multi outputs and multi-label classification.



Multi- Class

X	Class
X1	TV Show
X2	Movie
X3	TV Show
X4	Movie
X5	Unknown

Multi-Label

X	Action	Romance	Crime
X1	1	0	0
X2	0	0	1
X3	1	0	1

Multi-Output Multiclass

X	Class 1	Rating
X1	TV Show	PG
X2	Movie	TV-14
X3	TV Show	PG-13

Figure 3.3: Example of Multi Class Multi Output Model

3.2.2 Performance Evaluation of Multi-Output Learning

Label identification and evaluation are the primary steps to quantify the quality of labels and label representations. It also plays a key role in the performance of multi-output tasks. Learning models with different multi-outputs can be used to determine and then improve the label quality in terms of different tasks. Labels can be evaluated from three different perspectives.

- To add valid checks to determine whether the annotation has good quality (Step A).
- To determine and infer the possible labels, to conclude how well the chosen label representation can represent the labels (Step B).
- To determine the coverage, to evaluate how the provided label set well covers the dataset (Label Set). After the evaluation, there occurs a human intervention when the human expert explores and addresses the underlying issues, and provides feedback to improve different aspects of labels accordingly.

3.2.3 Real-World Applications of Multi-output Learning

1. Independent Vector: Independent vector is a vector with independent dimensions, where each dimension represents a particular label that does not necessarily depend on other labels. This includes tags, attributes, bag-of-words, bad-of-visual-words, hash codes, etc. of a given data.
2. Distribution: Provides information on a probability distribution for each dimension, like the tag with the largest weight.
3. Ranking: It shows the tags ordered from the most important to the least. Examples of its application are text categorization ranking, question answering, and visual object recognition.
4. Text: Text can be in the form of keywords, sentences, paragraphs, or even documents. Applications for text outputs can be document summarization and paragraph generation.
5. Sequence: Sequence (used in speech recognition, and language translation) is usually a sequence of elements selected from a label set or word set. Each element prediction is dependent on past predicted outputs and present input. An output sequence often corresponds with an input sequence.
6. Tree: The tree is represented as the hierarchical labeled structure to display the outputs. The outputs have a hierarchical internal structure where each output belongs to a label as well as its ancestors in the tree, useful in syntactic parsing.
7. Image: One of the output objects is an image consisting of multiple pixel values. A single pixel is predicted depending on the input and the pixels around it to consider an overall region prediction. Image output applications include super-

resolution construction, text-to-image synthesis, which generates images from natural language descriptions, and face generation.

8. Bounding Box: A bounding box is often used to find the exact locations of the objects that appeared in an image and it is commonly used in object recognition and object detection
9. Link: A partitioned social network with edges representing the friendship of the users, the goal is to predict whether two currently unlinked users will be friends in the future.
10. Graph: A graph made up of a set of nodes and edges and it is used to model the relations between objects, where each object is represented by a node. The connected objects are linked by an edge.
11. Others: Contour and polygons are similar to the bounding box which can be used to localize objects in an image. In information retrieval, the output can be a list of data objects that are similar to the given query. In image segmentation, the output is usually segmentation masks for different objects, used for detecting common saliency on multiple images.

4. IMPLEMENTATION OF AMAZON REVIEW ANALYSIS, CLASSIFICATION AND PREDICTION

4.1 SENTIMENT ANALYSIS

One of the key areas where NLP has been predominantly used is Sentiment analysis. The understanding of customer behavior and needs on a company's products and services is vital for organizations.

Generally, the feedback provided by a customer on a product can be categorized into Positive, Negative, and Neutral. Interpreting customer feedback through product reviews helps companies evaluate how satisfied the customers are with their products/services.

Sentiment analysis is the process of detecting positive or negative sentiments in text. It's often used by businesses to detect sentiment in social data, gauge brand reputation, and understand customers.

4.1.1 TYPES OF SENTIMENT ANALYSIS

Sentiment analysis focuses on the polarity of a text (positive, negative, neutral) but it also goes beyond polarity to detect specific feelings and emotions (angry, happy, sad, etc.), urgency (urgent, not urgent), and even intentions (interested v. not interested).

Depending on how you want to interpret customer feedback and queries, you can define and tailor your categories to meet your sentiment analysis needs. In the meantime, here are some of the most popular types of sentiment analysis:

Graded Sentiment Analysis

If polarity precision is important to your business, you might consider expanding your polarity categories to include different levels of positive and negative:

- Very positive
- Positive
- Neutral
- Negative
- Very negative

This is usually referred to as graded or fine-grained sentiment analysis, and could be used to interpret 5-star ratings in a review, for example:

Very Positive = 5 stars

Very Negative = 1 star

Emotion detection

Emotion detection sentiment analysis allows you to go beyond polarity to detect emotions, such as happiness, frustration, anger, and sadness.

Many emotion detection systems use lexicons (i.e. lists of words and the emotions they convey) or complex machine learning algorithms.

One of the downsides of using lexicons is that people express emotions in different ways. Some words that typically express anger, like bad or kill (e.g. your product is so bad or your customer support is killing me) might also express happiness (e.g. this is a badass or you are killing it).

Aspect-based Sentiment Analysis

Usually, when analyzing sentiments of texts, you'll want to know which aspects or features people are mentioning in a positive, neutral, or negative way.

That's where aspect-based sentiment analysis can help, for example in this product review - "The battery life of this camera is too short", an aspect-based classifier would be able to determine that the sentence expresses a negative opinion about the battery life of the product in question.

Multilingual sentiment analysis

Multilingual sentiment analysis can be difficult. It involves a lot of preprocessing and resources. Most of these resources are available online (e.g., sentiment lexicons), while others need to be created (e.g., translated corpora or noise detection algorithms), but you'll need to know how to code to use them.

Alternatively, you could detect the language in texts automatically with a language classifier, then train a custom sentiment analysis model to classify texts in the language of your choice.

4.1.2 SENTIMENT ANALYSIS IMPORTANCE

Since humans express their thoughts and feelings more openly than ever before, sentiment analysis is fast becoming an essential tool to monitor and understand the sentiment in all types of data.

Automatically analyzing customer feedback, such as opinions in survey responses and social media conversations, allows brands to learn what makes customers happy or frustrated so that they can tailor products and services to meet their customers' needs.

For example, using sentiment analysis to automatically analyze 4,000+ open-ended responses in your customer satisfaction surveys could help you discover why customers are happy or unhappy at each stage of the customer journey.

Maybe you want to track brand sentiment so you can detect disgruntled customers immediately and respond as soon as possible. Maybe you want to compare sentiment from one quarter to the next to see if you need to take action. Then you could dig deeper into your qualitative data to see why sentiment is falling or rising.

The overall benefits of sentiment analysis include:

Sorting Data at Scale

Can you imagine manually sorting through thousands of tweets, customer support conversations, or surveys? There's just too much business data to process manually. Sentiment analysis helps businesses process huge amounts of unstructured data efficiently and cost-effectively.

Real-Time Analysis

Sentiment analysis can identify critical issues in real-time, for example, a PR crisis on social media escalating. Is an angry customer about to churn? Sentiment analysis models can help you immediately identify these kinds of situations, so you can take action right away.

Consistent criteria

It's estimated that people only agree around 60-65% of the time when determining the sentiment of a particular text. Tagging text by sentiment is highly subjective, influenced by personal experiences, thoughts, and beliefs.

By using a centralized sentiment analysis system, companies can apply the same criteria to all of their data, helping them improve accuracy and gain better insights. The applications of sentiment analysis are endless.

4.1.3 SENTIMENT ANALYSIS IN PYTHON

There are many packages available in python which use different methods to do sentiment analysis. In the next section, we shall go through some of the most popular methods and packages.

Rule-based sentiment analysis is one of the very basic approaches to calculating text sentiments. It only requires minimal pre-work and the idea is quite simple, this method does not use any machine learning to figure out the text sentiment. For example, we can figure out the sentiments of a sentence by counting the number of times the user has used the word "sad" in his/her tweet.

VADER Sentiment

Valence aware dictionary for sentiment reasoning (VADER) is another popular rule-based sentiment analyzer. It uses a list of lexical features (e.g., words) which are labeled as positive or negative according to their semantic orientation to calculate the text sentiment. Vader sentiment returns the probability of a given input sentence to be Positive, negative, and neutral.

For example:

“The food was great!”
Positive: 99%
Negative:1%
Neutral: 0%

These three probabilities will add up to 100%.

Let’s see how to use VADER:

Code:

```
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer


analyzer = SentimentIntensityAnalyzer()

sentence = "The food was great!"

vs = analyzer.polarity_scores(sentence)

print ("{:-<65} {}”. format (sentence, str(vs)))


output:
```

```
{'compound': 0.6588, 'neg': 0.0, 'neu': 0.406, 'pos': 0.594}
```

Vader is optimized for social media data and can yield good results when used with data from Twitter, Facebook, etc.

The main drawback of the rule-based approach for sentiment analysis is that the method only cares about individual words and completely ignores the context in which it is used. For example, *“the party was savage”* will be negative when considered by any token-based algorithms.

As shown in the above code snippet the Vader package can estimate the sentiment of the reviews into various percentages of sentiments as Positive, Negative, Neutral, and a compound value whose value lies between o and 1.

Making use of VADER values and the rating values provided in the data set, custom metrics are derived to perform sentiment analysis which is as shown in below flow chart 4.1.

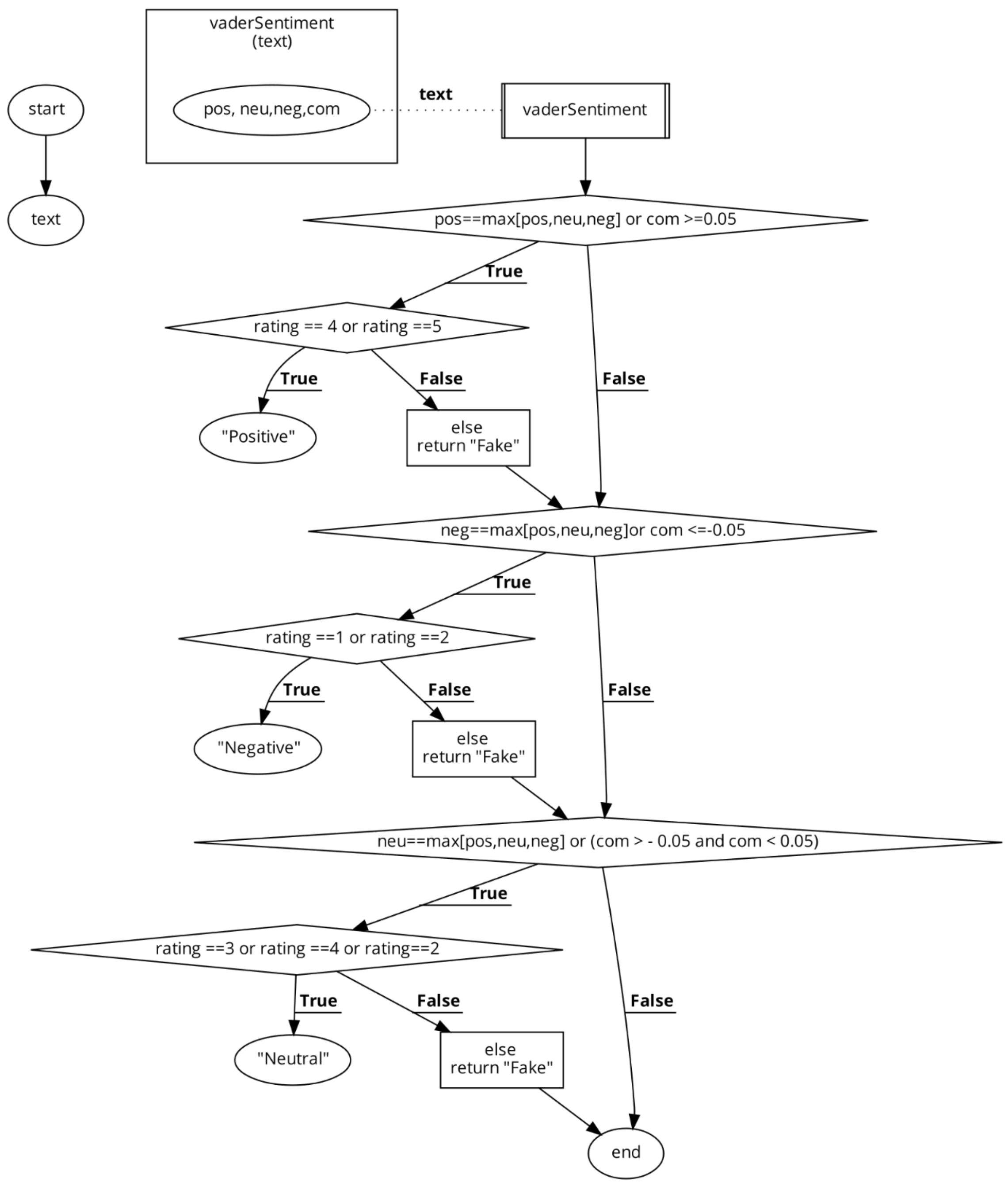


Figure 4.1: Flow Diagram of Sentiment Analysis

With the above flow chart, it’s clear that we can perform not only sentiment analysis in terms of positive, negative, and neutral sentiments but we are also able to understand if the reviews are fake or wrongly marked with ratings.

Thus, we can eliminate the reviews which are mapped to fake as they may be fake reviews or ratings are false as they are not mapping to the sentiment of the reviews by the user.

Thus, we drop these reviews thereby reducing the reviews dataset size which helps in fast computing. Next is classification.

4.2 REVIEWS SEGMENTATION

Before we could go to the classification, first reviews need to process so that the algorithms perform their instructions on them.

4.2.1 REVIEWS / TEXT PRE-PROCESSING

First, import the dataset into pandas and find the null values, in this project there were around 200 null values which are dropped from the data frame and proceeded to tokenization.

Tokenization

Tokenization is essentially splitting a phrase, sentence, paragraph, or entire text document into smaller units, such as individual words or terms. Each of these smaller units is called a token.

There are numerous uses for doing this. We can use this tokenized form to:

- Count the number of words in the text
- Count the frequency of the word, that is, the number of times a particular word is present

Tokenization using NLTK

NLTK, short for Natural Language ToolKit, is a library written in Python for symbolic and statistical Natural Language Processing. NLTK contains a module called tokenize which further classifies into two sub-categories:

Word tokenize: word_tokenize() method to split a sentence into tokens or words

Sentence tokenize: sent_tokenize() method to split a document or paragraph into sentences

Tokenization is a critical step in the overall NLP pipeline. We cannot simply jump into the model-building part without cleaning the text first.

Using the word_tokenize method, the text is transformed into tokens for further processing which includes stop-word removal, pos tagging, Lemmatization, and stemming which are mentioned in the below flow chart figure 4.2.

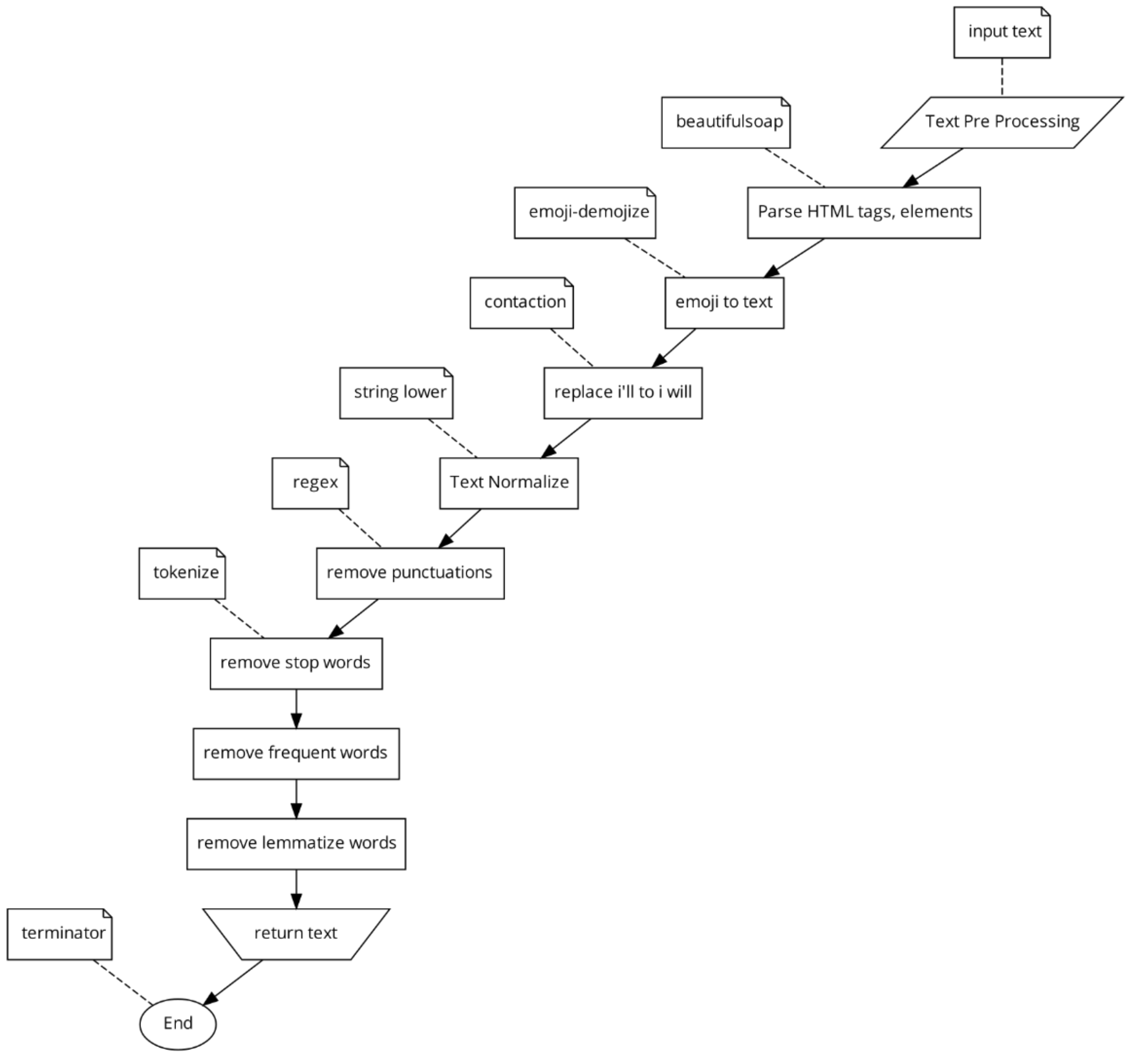


Figure 4.2: Flow Chart Describing the Flow of Text Pre-Processing Steps

Now to classify a review into the product, delivery, and packaging categories in the project had to apply regression to find the text of the reviews which contains delivery or packaging as tokens, if the text contains any of the two categories in the text they will be labeled accordingly.

The flow chart to represent this process is described below in figure 4.3.

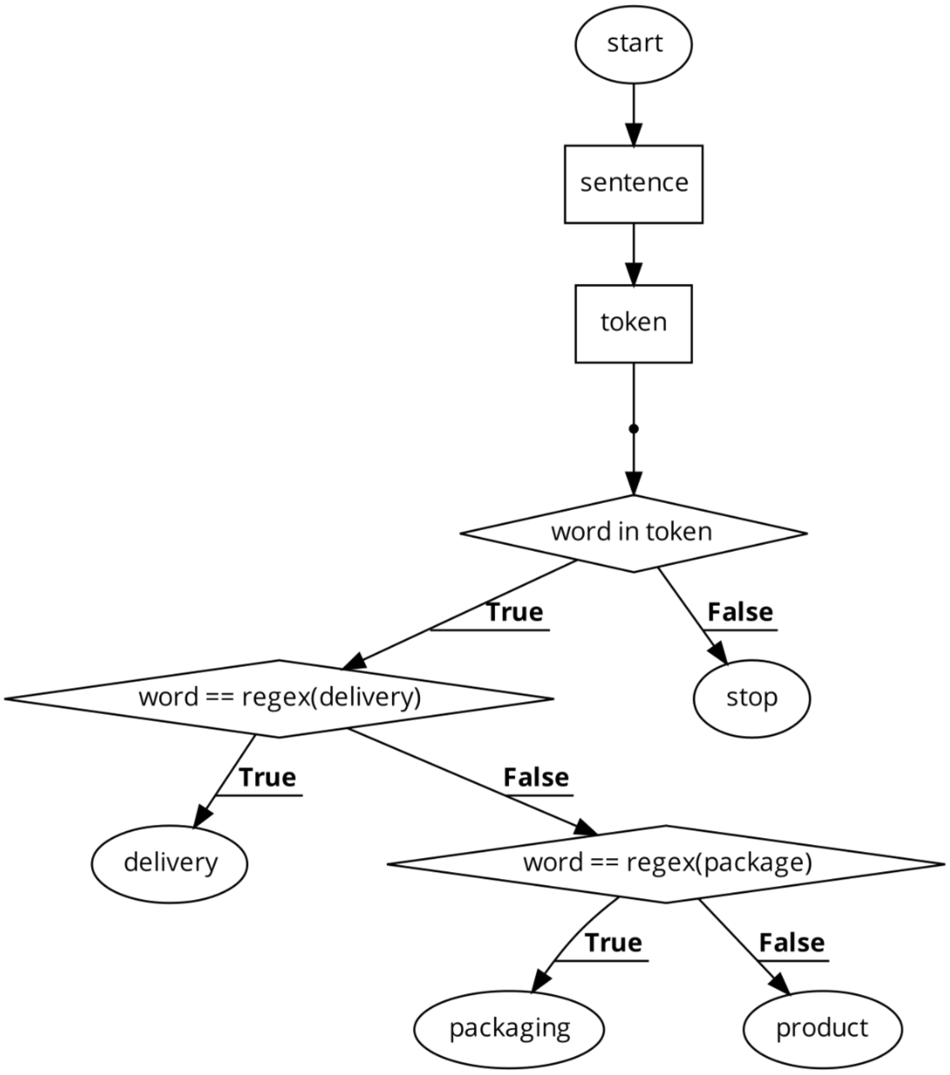


Figure 4.3: Flow Chart Representing Review Segmentation into Products, Packaging, and Delivery.

Now that we have all the required Classes for classification, we can now proceed to reduce the data set to only the required columns namely rating, reviews, sentiments, and category.

We can further reduce the rows such that there is a minimum no of rows in all the required labels, hence need to group all the required labels using pandas and get top rows from such groups whose helping votes counts are high so that the reviews so present in the down-sampled data are useful reviews.

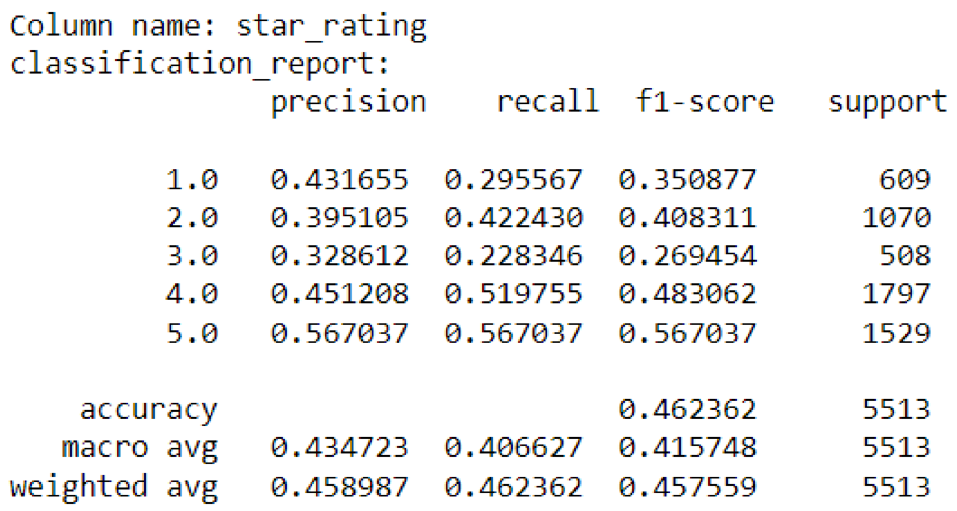
These prepared data sets along with labels are then sent as inputs to the multi-class multi-output classifiers which will help in classifying the reviews into sentiments, and categories and predict the rating.

To perform such classifications, we need to transform the text data into vectors, tfidf transforms, and word embedding and then classifier all in a pipeline along with hyperparameter tuning to fit the classifier with the best parameters.

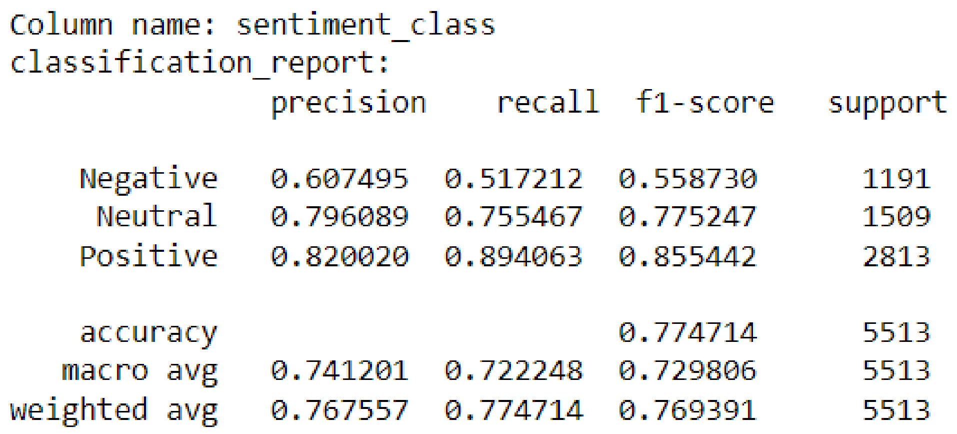
5. RESULTS AND CONCLUSION

5.1 RESULT ANALYSIS

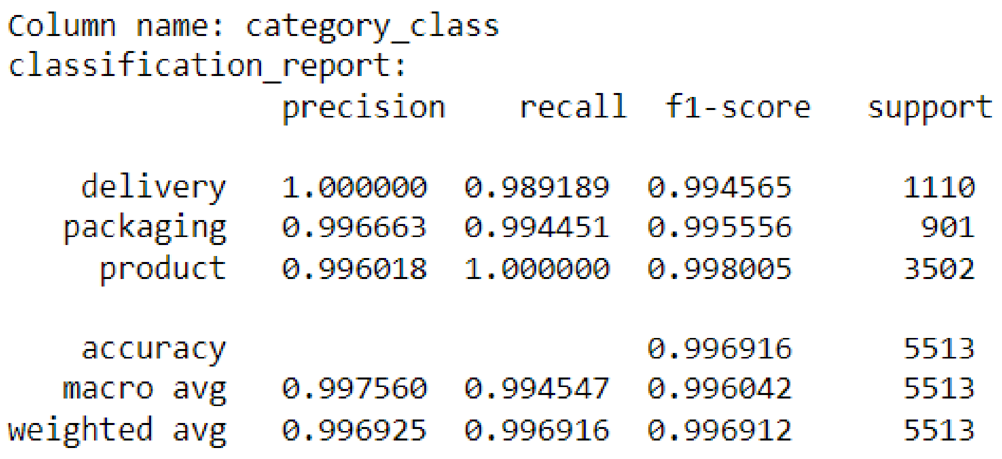
Decision Tree Classifier (Multi-Output) Result:



```
Column name: star_rating
classification_report:
              precision    recall  f1-score   support

         1.0   0.431655  0.295567  0.350877       609
         2.0   0.395105  0.422430  0.408311      1070
         3.0   0.328612  0.228346  0.269454       508
         4.0   0.451208  0.519755  0.483062      1797
         5.0   0.567037  0.567037  0.567037      1529

    accuracy                       0.462362      5513
   macro avg   0.434723  0.406627  0.415748      5513
weighted avg   0.458987  0.462362  0.457559      5513
```



```
Column name: sentiment_class
classification_report:
              precision    recall  f1-score   support

    Negative   0.607495  0.517212  0.558730      1191
     Neutral   0.796089  0.755467  0.775247      1509
    Positive   0.820020  0.894063  0.855442      2813

    accuracy                       0.774714      5513
   macro avg   0.741201  0.722248  0.729806      5513
weighted avg   0.767557  0.774714  0.769391      5513
```



```
Column name: category_class
classification_report:
              precision    recall  f1-score   support

    delivery   1.000000  0.989189  0.994565      1110
   packaging   0.996663  0.994451  0.995556       901
     product   0.996018  1.000000  0.998005      3502

    accuracy                       0.996916      5513
   macro avg   0.997560  0.994547  0.996042      5513
weighted avg   0.996925  0.996916  0.996912      5513
```

The accuracy score of the Star Rating is: 0.46

The accuracy score of the Sentiment Class is: 0.77

The accuracy score of the Category Class is: 0.99

Random Forest Classifier (Multi-Output) Result:

```
Column name: star_rating
classification_report:
              precision    recall  f1-score   support

         1.0   0.658683  0.176565  0.278481       623
         2.0   0.434160  0.441319  0.437710      1031
         3.0   0.430769  0.101449  0.164223       552
         4.0   0.443775  0.614230  0.515272      1799
         5.0   0.577473  0.642573  0.608286      1508

    accuracy                       0.488845      5513
   macro avg   0.508972  0.395227  0.400794      5513
weighted avg   0.501532  0.488845  0.464301      5513

Column name: sentiment_class
classification_report:
              precision    recall  f1-score   support

    Negative   0.865882  0.313725  0.460576      1173
     Neutral   0.789346  0.838046  0.812968      1556
    Positive   0.786088  0.970187  0.868489      2784

    accuracy                       0.793216      5513
   macro avg   0.813772  0.707320  0.714011      5513
weighted avg   0.803986  0.793216  0.766027      5513
```

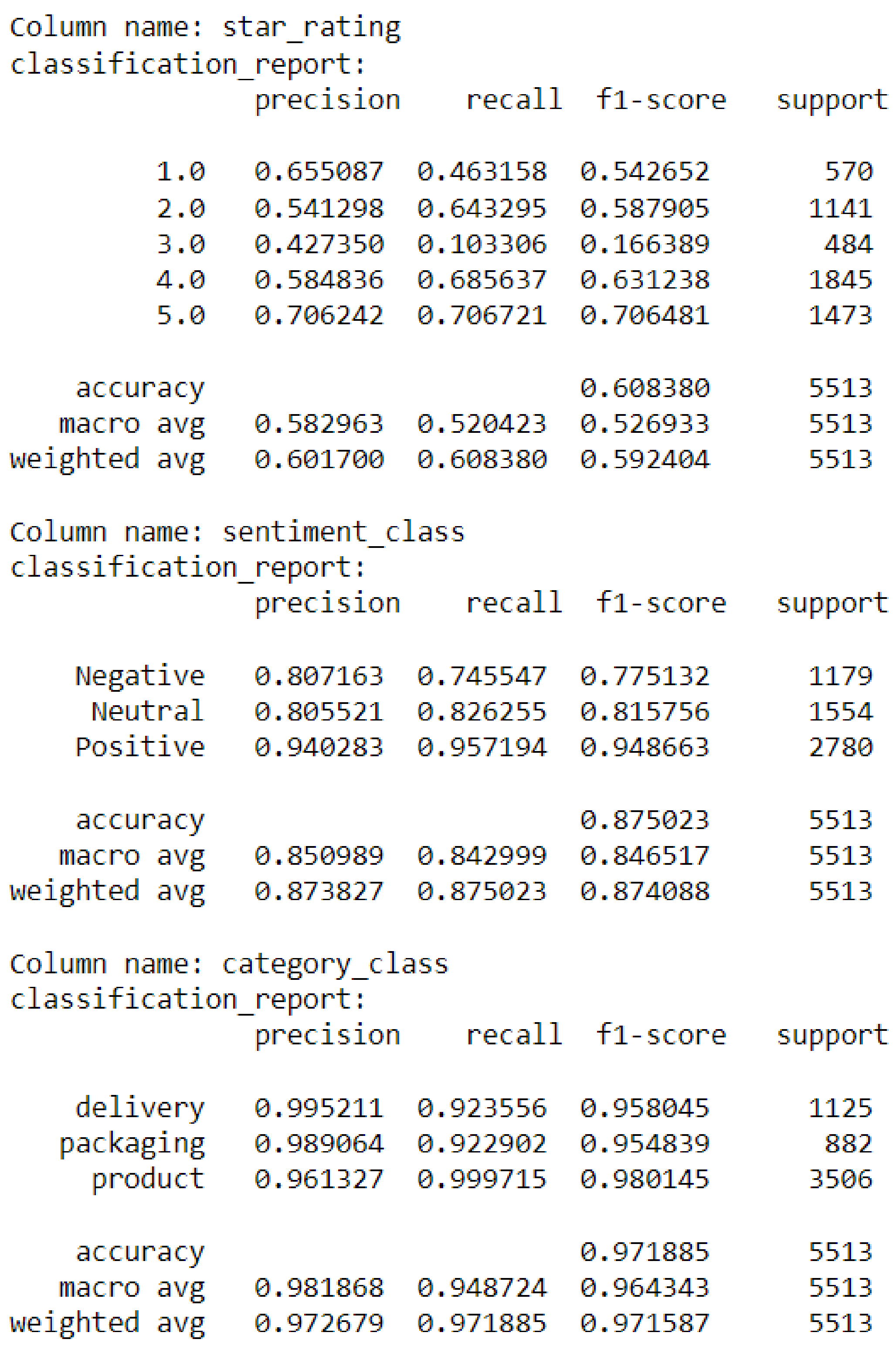
```
Column name: category_class
classification_report:
              precision    recall  f1-score   support

    delivery   0.987775  0.700781  0.819888      1153
   packaging   0.964626  0.787778  0.867278       900
     product   0.870960  0.996821  0.929650      3460

    accuracy                       0.900780      5513
   macro avg   0.941120  0.828460  0.872272      5513
weighted avg   0.910682  0.900780  0.896512      5513
```

The accuracy score of the Star Rating is: 0.48

The accuracy score of the Sentiment Class is: 0.79

The accuracy score of the Category Class is: 0.90

Support Vector Machines (Multi-Output) Result:



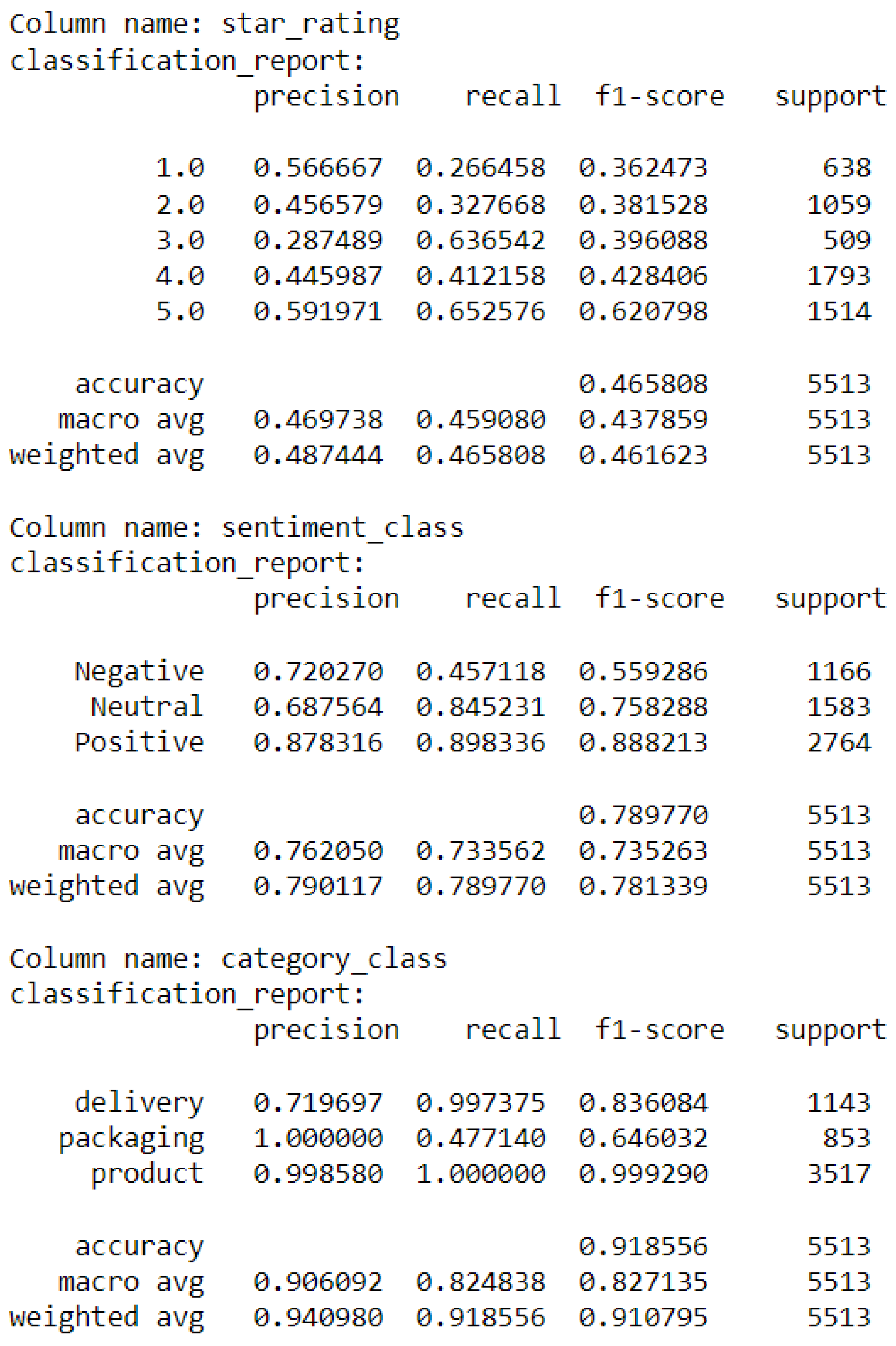
```
Column name: star_rating
classification_report:
               precision    recall  f1-score   support

         1.0    0.655087  0.463158  0.542652       570
         2.0    0.541298  0.643295  0.587905      1141
         3.0    0.427350  0.103306  0.166389       484
         4.0    0.584836  0.685637  0.631238      1845
         5.0    0.706242  0.706721  0.706481      1473

    accuracy                        0.608380      5513
   macro avg    0.582963  0.520423  0.526933      5513
weighted avg    0.601700  0.608380  0.592404      5513

Column name: sentiment_class
classification_report:
               precision    recall  f1-score   support

    Negative    0.807163  0.745547  0.775132      1179
     Neutral    0.805521  0.826255  0.815756      1554
    Positive    0.940283  0.957194  0.948663      2780

    accuracy                        0.875023      5513
   macro avg    0.850989  0.842999  0.846517      5513
weighted avg    0.873827  0.875023  0.874088      5513

Column name: category_class
classification_report:
               precision    recall  f1-score   support

    delivery    0.995211  0.923556  0.958045      1125
   packaging    0.989064  0.922902  0.954839       882
     product    0.961327  0.999715  0.980145      3506

    accuracy                        0.971885      5513
   macro avg    0.981868  0.948724  0.964343      5513
weighted avg    0.972679  0.971885  0.971587      5513
```

The accuracy score of Star Rating is: 0.60

The accuracy score of the Sentiment Class is: 0.87

The accuracy score of the Category Class is: 0.97

Ada Boost Classifier (Multi-Output) Result:



```
Column name: star_rating
classification_report:
               precision    recall  f1-score   support

         1.0    0.566667  0.266458  0.362473       638
         2.0    0.456579  0.327668  0.381528      1059
         3.0    0.287489  0.636542  0.396088       509
         4.0    0.445987  0.412158  0.428406      1793
         5.0    0.591971  0.652576  0.620798      1514

    accuracy                        0.465808      5513
   macro avg    0.469738  0.459080  0.437859      5513
weighted avg    0.487444  0.465808  0.461623      5513

Column name: sentiment_class
classification_report:
               precision    recall  f1-score   support

    Negative    0.720270  0.457118  0.559286      1166
     Neutral    0.687564  0.845231  0.758288      1583
    Positive    0.878316  0.898336  0.888213      2764

    accuracy                        0.789770      5513
   macro avg    0.762050  0.733562  0.735263      5513
weighted avg    0.790117  0.789770  0.781339      5513

Column name: category_class
classification_report:
               precision    recall  f1-score   support

    delivery    0.719697  0.997375  0.836084      1143
   packaging    1.000000  0.477140  0.646032       853
     product    0.998580  1.000000  0.999290      3517

    accuracy                        0.918556      5513
   macro avg    0.906092  0.824838  0.827135      5513
weighted avg    0.940980  0.918556  0.910795      5513
```

The accuracy score of the Star Rating is: 0.46

The accuracy score of the Sentiment Class is: 0.78

The accuracy score of the Category Class is: 0.91

Model Validation Output:

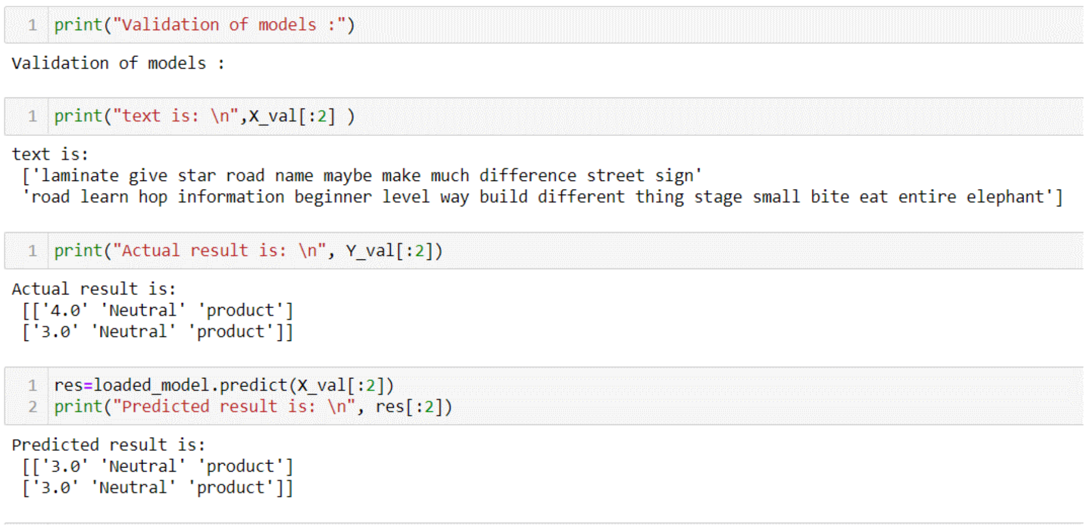

```
print("Validation of models :")
```
```
Validation of models :
```
```
print("text is: \n",X_val[:2] )
```
```
text is:
 ['laminate give star road name maybe make much difference street sign'
 'road learn hop information beginner level way build different thing stage small bite eat entire elephant']
```
```
print("Actual result is: \n", Y_val[:2])
```
```
Actual result is:
 [['4.0' 'Neutral' 'product']
 ['3.0' 'Neutral' 'product']]
```
```
res=loaded_model.predict(X_val[:2])
print("Predicted result is: \n", res[:2])
```
```
Predicted result is:
 [['3.0' 'Neutral' 'product']
 ['3.0' 'Neutral' 'product']]
```

Figure 5.1: Model Validation Output

Overall observing all the models and their performance metrics SVM performed better than all the other models.

5.2 PROJECT DEPLOYMENT

Model Deployment helps you showcase your work to the world and make better decisions with it. But deploying a model can get a little tricky at times. Before deploying the model a lot of things need to be looked into, such as data storage, pre-processing, model building, and monitoring. This can be a bit confusing as the number of tools that perform these model deployment tasks efficiently is few. Enter, Streamlit!

Streamlit is a popular open-source framework used for model deployment by machine learning and data science teams. And the best part is it's free of cost and purely in python.

Introduction to Streamlit

Streamlit lets you create apps for your machine-learning project using simple python scripts. It also supports hot-reloading, so that your app can update live as you edit and save your file. An app can be built in a few lines of code only(as we will see below) using the Streamlit API. Adding a widget is the same as declaring a variable. There is no need to write a backend, define different routes or handle HTTP requests. It is easy to deploy and manage. More information can be found on their website – https://www.streamlit.io/.

Here are some of the key features of Streamlit that I found interesting and useful:

- It quickly turns data scripts into shareable web applications. You just have to pass a running script to the tool and it can convert that to a web app.

- Everything in Python. The best thing about Streamlit is that everything we do is in Python. Starting from loading the model to creating the front end, all can be done using Python.

- All for free. It is open source and hence no cost is involved. You can deploy your apps without paying for them.
- No front-end experience is required. Model deployment generally contains two parts, frontend, and backend. The backend is generally a working model, a machine learning model in our case, which is built-in python. And the front-end part, which generally requires some knowledge of other languages like java scripts, etc.
- Using Streamlit, we can create this front end in Python itself. So, we need not learn any other programming languages or web development techniques. Understanding Python is enough.

Now, let's look at the deployment pipeline if we use Streamlit:

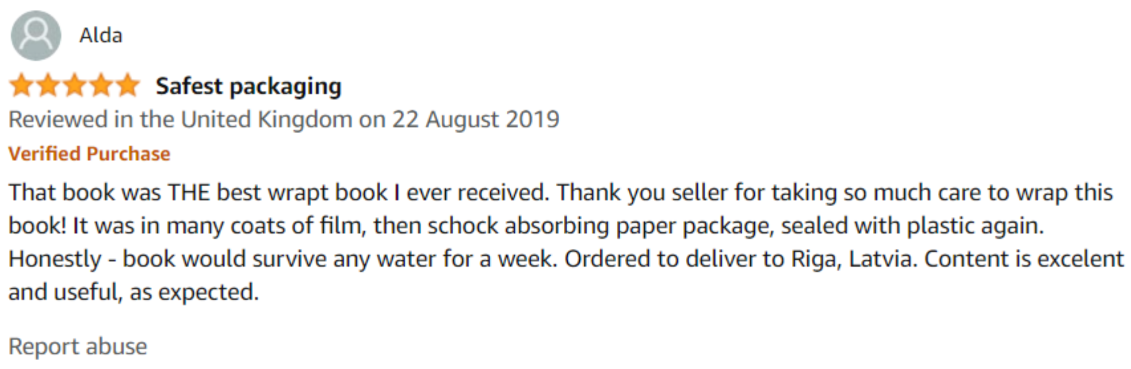
1. Model Building
2. Creating a python script
3. Create front-end: Python
4. Deploy

Here we built the model and created a python script for it. Then we built the front-end for the app which will be in python and finally, we will deploy the model. That's it. Our model is deployed.

Steps for Deployment:

1. So first we will train our model.
2. We will split the dataset into training and testing datasets and will use various Classifiers.
3. Will acquire the best model by comparing the accuracies and choosing the model with the highest accuracy
4. Now, to use this model to predict other unknown data, we need to save it.
5. We can save it by using pickle, which is used for serializing and deserializing a Python object structure.
6. Now we can get down to using Streamlit to deploy the model.
7. Create the python script (app.py) to predict the input text using the saved model and produce the output on the web application using Streamlit.
8. You can run this by the following command in the terminal: "streamlit run app.py ". The website will open in your browser and then you can test it.
9. The code is then pushed to the public GitHub repository and deployed on Streamlit Cloud for public access.

Deployed Project Screenshot:



Alda

★★★★★ **Safest packaging**

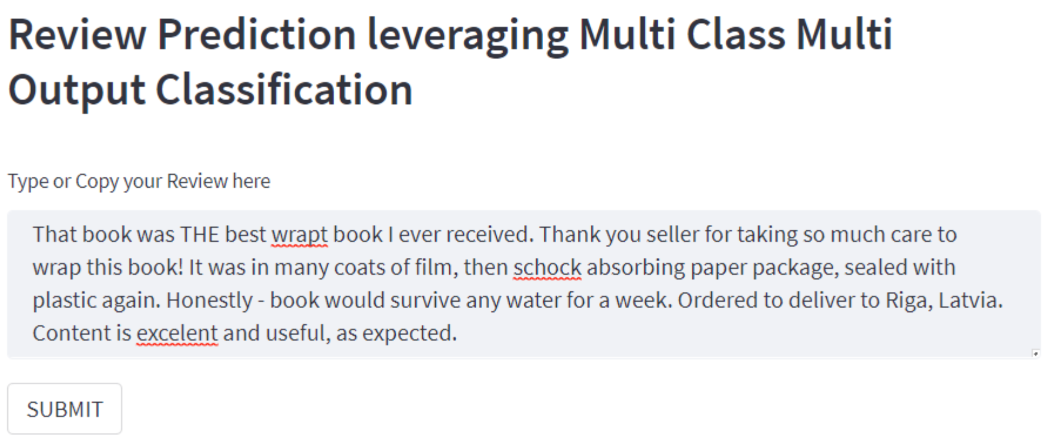
Reviewed in the United Kingdom on 22 August 2019

Verified Purchase

That book was THE best wrapt book I ever received. Thank you seller for taking so much care to wrap this book! It was in many coats of film, then schock absorbing paper package, sealed with plastic again. Honestly - book would survive any water for a week. Ordered to deliver to Riga, Latvia. Content is excelent and useful, as expected.

Report abuse

Figure 5.2: Actual Amazon Review of a Book

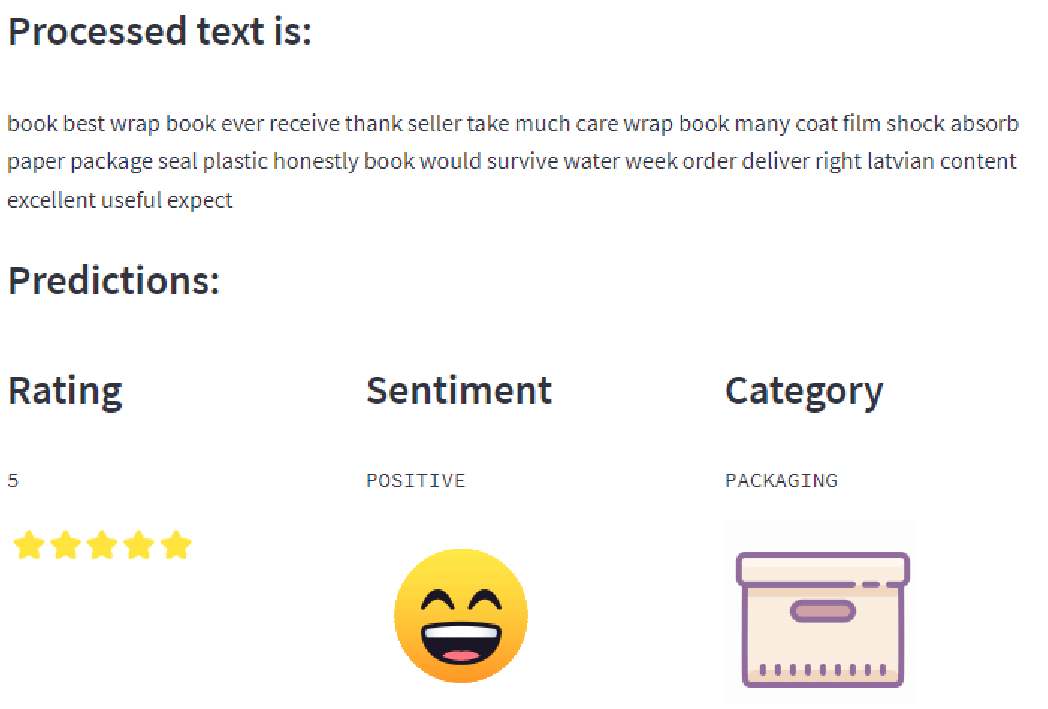


Review Prediction leveraging Multi Class Multi Output Classification

Type or Copy your Review here

That book was THE best wrapt book I ever received. Thank you seller for taking so much care to wrap this book! It was in many coats of film, then schock absorbing paper package, sealed with plastic again. Honestly - book would survive any water for a week. Ordered to deliver to Riga, Latvia. Content is excelent and useful, as expected.

Figure 5.3: Web Application for Review Prediction



Processed text is:

book best wrap book ever receive thank seller take much care wrap book many coat film shock absorb paper package seal plastic honestly book would survive water week order deliver right latvian content excellent useful expect

Predictions:

Rating	Sentiment	Category
5	POSITIVE	PACKAGING

Figure 5.4: Predicted Output on Amazon Review of a Book

Observation:

- From the above screenshots, Figure 5.2 represents the actual review obtained from the amazon website for a book.
- When that same review is fed as input to the model as observed in Figure 5.3 it classified the review as a 5-star rating same as presented by the user, a prediction shown in Figure 5.4.
- The model could also predict the sentiment as positive which is true based on the customer review as there is no sarcasm or fakeness in the text.
- The model also segmented the review into the Packaging Category which is also appropriate based on the text context.

5.3 CONCLUSION

The main goal of this study was to determine which machine learning algorithm methods perform better in the task of text classification. This was accomplished by using Amazon reviews as data set. The classifiers were evaluated by comparing their accuracies in different cases of experiments.

The results from the study showed that in terms of accuracy the SVM approach achieves better results than the other approaches when the data set was used as a training and testing data set.

5.4 FUTURE SCOPE

The data set used for the model evaluation was down-sampled which did have an impact on the model accuracy so need to work on the complete dataset to achieve high accuracy and precision results.

More Multi-Class Multi-Output Classifiers can be employed to further increase the study on multi-output classifiers and their implementation on the amazon reviews dataset.

REFERENCES

[1] Kowsari, K.; Jafari Meimandi, K.; Heidarysafa, M.; Mendu, S.; Barnes, L.; Brown, D. Text classification algorithms: A survey. Information 2019, 10, 150. [CrossRef]

[2] Rusli, A.; Young, J.C.; Iswari, N.M.S. Identifying fake news in Indonesian via supervised binary text classification. In Proceedings of the 2020 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT), Bali, Indonesia, 7–8 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 86–90.

[3] Al-Rawashdeh, G.; Mamat, R.; Abd Rahim, N.H.B. Hybrid water cycle optimization algorithm with simulated annealing for spam e-mail detection. IEEE Access 2019, 7, 143721–143734. [CrossRef]

[4] Shehnepoor, S.; Salehi, M.; Farahbakhsh, R.; Crespi, N. NetSpam: A network-based spam detection framework for reviews in online social media. IEEE Trans. Inf. Forensics Secur. 2017, 12, 1585–1595. [CrossRef]

[5] Alterkavı, S.; Erbay, H. Novel authorship verification model for social media accounts compromised by a human. Multimed. Tools Appl. 2021, 80, 13575–13591. [CrossRef]

[6] Alterkavı, S.; Erbay, H. Design and Analysis of a Novel Authorship Verification Framework for Hijacked Social Media Accounts Compromised by a Human. Secur. Commun. Netw. 2021, 2021, 8869681. [CrossRef]

[7] Liu, S.; Cheng, X.; Li, F.; Li, F. TASC: Topic-adaptive sentiment classification on dynamic tweets. IEEE Trans. Knowl. Data Eng. 2014, 27, 1696–1709. [CrossRef]

[8] Esposito, F.; Corazza, A.; Cutugno, F. Topic Modelling with Word Embeddings. In Proceedings of the Third Italian Conference on Computational Linguistics (CLiC-it 2016) & Fifth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2016), Napoli, Italy, 5–7 December 2016.

[9] Leeuwenberg, A.; Vela, M.; Dehdari, J.; van Genabith, J. A minimally supervised approach for synonym extraction with word embeddings. Prague Bull. Math. Linguist. 2016, 105, 111. [CrossRef]

[10] PForms. 68 Useful eCommerce Statistics You Must Know in 2022. Available online: https://wpforms.com/ecommercestatistics/ (accessed on 4 August 2022).

[11] Nasdaq. UK Online Shopping and E-Commerce Statistics for 2017. Available online: https://www.nasdaq.com/articles/ukonline-shopping-and-e-commerce-statistics-2017-2017-03-14 (accessed on 4 August 2022).

[12] Zhang, S.; Zhang, D.; Zhong, H.; Wang, G. A multiclassification model of sentiment for E-commerce reviews. IEEE Access 2020, 8, 189513–189526. [CrossRef]

[13] Medhat, W.; Hassan, A.; Korashy, H. Sentiment analysis algorithms and applications: A survey. Ain Shams Eng. J. 2014, 5, 1093–1113. [CrossRef]

[14] Muslim, M.A. Support vector machine (svm) optimization using grid search and unigram to improve e-commerce review accuracy. J. Soft Comput. Explor. 2020, 1, 8–15.

[15] Xu, F.; Pan, Z.; Xia, R. E-commerce product review sentiment classification based on a naïve Bayes continuous learning framework. Inf. Process. Manag. 2020, 57, 102221. [CrossRef]

[16] Vanaja, S.; Belwal, M. Aspect-level sentiment analysis on e-commerce data. In Proceedings of the 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 11 July 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1275–1279.

[17] Jabbar, J.; Urooj, I.; JunSheng, W.; Azeem, N. Real-time sentiment analysis on E-commerce application. In Proceedings of the 2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC), Banff, AB, Canada, 9–11 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 391–396.

[18] Parveen, N.; Santhi, M.; Burra, L.R.; Pellakuri, V.; Pellakuri, H. Women's e-commerce clothing sentiment analysis by probabilistic model LDA using R-SPARK. Mater. Today Proc. 2021, in press. [CrossRef]

[19] Tripathi, P.; Singh, S.; Chhajer, P.; Trivedi, M.C.; Singh, V.K. Analysis and prediction of extent of helpfulness of reviews on E-commerce websites. Mater. Today Proc. 2020, 33, 4520–4525. [CrossRef]

[20] Kumar, K.S.; Desai, J.; Majumdar, J. Opinion mining and sentiment analysis on online customer review. In Proceedings of the 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Chennai, India, 15–17 December 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1–4.

[21] Miyoshi, T.; Nakagami, Y. Sentiment classification of customer reviews on electric products. In Proceedings of the 2007 IEEE International Conference on Systems, Man and Cybernetics, Banff, AB, Canada, 5–8 October 2017; IEEE: Piscataway, NJ, USA, 2007; pp. 2028–2033.

[22] Guan, Z.; Chen, L.; Zhao, W.; Zheng, Y.; Tan, S.; Cai, D. Weakly-Supervised Deep Learning for Customer Review Sentiment Classification. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16), New York, NY, USA, 9–15 July 2016; pp. 3719–3725.

[23] Shoja, B.M.; Tabrizi, N. Customer reviews analysis with deep neural networks for e-commerce recommender systems. IEEE Access 2019, 7, 119121–119130. [CrossRef]

[24] Gu, T.; Xu, G.; Luo, J. Sentiment analysis via deep multichannel neural networks with variational information bottleneck. IEEE Access 2020, 8, 121014–121021. [CrossRef]

[25] B˙Ilen, B.; Horasan, F. LSTM network based sentiment analysis for customer reviews. Politek. Derg. 2021. [CrossRef]

[26] Vural, A.G.; Cambazoglu, B.B.; Senkul, P.; Tokgoz, Z.O. A framework for sentiment analysis in turkish: Application to polarity detection of movie reviews in turkish. In Computer and Information Sciences III; Springer: Berlin/Heidelberg, Germany, 2013; pp. 437–445.

[27] Acikalin, U.U.; Bardak, B.; Kutlu, M. Turkish sentiment analysis using bert. In Proceedings of the 2020 28th Signal Processing and Communications Applications Conference (SIU), Gaziantep, Turkey, 5–7 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–4.

[28] Santur, Y. Sentiment analysis based on gated recurrent unit. In Proceedings of the 2019 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, Turkey, 21–22 September 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–5.

[29] Ozyurt, B.; Akcayol, M.A. A new topic modeling based approach for aspect extraction in aspect based sentiment analysis: SS-LDA. Expert Syst. Appl. 2021, 168, 114231. [CrossRef]

[30] Kadhim, A.I. Term weighting for feature extraction on Twitter: A comparison between BM25 and TF-IDF. In Proceedings of the 2019 International Conference on Advanced Science and Engineering (ICOASE), Zakho, Iraq, 2–4 April 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 124–128.