An NLP-Deep Learning approach for Product Rating Prediction Based on Online Reviews and Product Features

Tolou Amirifar

A Thesis

In the Department of

Mechanical, Industrial and Aerospace Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of

Master of Applied Science (Industrial Engineering)

at Concordia University

Montréal, Québec, Canada

April 2022

©Tolou Amirifar, 2022

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Tolou Amirifar**

Entitled: **An NLP-Deep Learning approach for Product Rating Prediction Based on Online Reviews and Product Features**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Industrial Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

Dr. Onur Kuzgunkaya	Chair
Dr. Danielle Morin	External Examiner
Dr. Onur Kuzgunkaya	Examiner
Dr. Masoumeh Kazemi Zanjani	Thesis Supervisor
Dr. Salim Lahmiri	Thesis Supervisor

Approved by

Dr. Martin D. Pugh, Chair
Department of Mechanical, Industrial and Aerospace Engineering

May 5, 2022

Dr. Mourad Debbabi, Dean of Faculty

Abstract

An NLP-Deep Learning approach for Product Rating Prediction Based on Online Reviews and Product Features

Tolou Amirifar

This study focuses on predicting the popularity of a product based on its overall rating score. Unlike previous studies that focus on predicting the review rating based on sentiment analysis and polarity of the reviews, in this thesis, the effect of product features in determining its popularity is directly measured and analyzed in order to predict its overall rating score. To this end, a methodology consisting of three phases is considered. Phase 1 predicts the overall rating by feeding the general product features, extracted from the online product information available on Amazon webpages to a Deep Learning (DL) model. Phase 2 identifies other features that customers care about the most by applying the Named Entity Recognition (NER) algorithm to the customer online reviews; and lastly, Phase 3 feeds the combination of the general and custom features to the DL model to predict the overall rating score of the product.

The experimental results on a dataset of laptop products, collected from Amazon, indicate an impressive performance of the proposed approach, which is mainly attributed to including custom product features to the inputs of the DL algorithm when compared with the existing method. More precisely, the proposed model could achieve the highest accuracy score of 84.01%, 84.68% for recall, 87.63% for precision, and 84.06% for F1 score. Applying this procedure could help businesses identify the specific areas of strengths and weaknesses of their products or services from the perspective of their customers, allowing them to thrive in today's competitive markets.

Acknowledgments

I am extremely thankful to my supervisors Dr. Masoumeh Kazemi Zanjani from the Department of Mechanical, Industrial and Aerospace Engineering (MIAE) and Dr. Salim Lahmiri from the Department of Supply Chain and Business Technology Management at the John Molson School of Business, for their noble supervision, support with full encouragement and enthusiasm, valuable suggestions, and motivating advice. This thesis would not have been possible without your trust in me and your guidance from the initial step in research that allowed me to develop an understanding of the subject.

I am grateful for my parents whose constant love and support keep me motivated and confident. My accomplishments and success are because they believed in me. Finally, I owe my deepest gratitude to my husband, Ali. I am forever thankful for the unconditional love and support throughout the entire thesis process and every day.

Contents

List of Figures

List of Tables

Acronyms

The following abbreviations are used in this manuscript:

DL	Deep Learning
ML	Machine Learning
NLP	Natural Language Processing
UGC	User Generated Content
NER	Named Entity Recognition
ANN	Artificial Neural Network
DNN	Deep Neural Network
PNN	Probabilistic Neural Network
RBFNN	Radial Basis Function Neural Network
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
NN-based	Neural Network Based
KNN	K-Nearest Neighbor
GRU	Gated Recurrent Unit
SVM	Support Vector Machine
IE	Information Extraction
IR	Information Retrieval
Bi-GRU	Bidirectional Gated Recurrent Unit
RS	Recommender Systems
BOW	Bag of Words
TFIDF	Term Frequency-Inverse Document Frequency
PoS	Part-of-Speech
PDF	Probability Distribution Function
NLTK	Natural Language Toolkit
SMOTE	Synthetic Minority Oversampling Technique

1. Introduction

In recent years, the way customers perceive the products and services of the companies has had a huge impact on their financial viability and market growth. Customer reviews help businesses to better understand the customer perception of the product, and brand, generate ideas for quality improvements, reduce complaints/claims, and gather information on introducing enhanced/new features for new products. This User-Generated Content (UGC) plays a significant role in determining purchasing behavior because a consumer is eager to get the opinion of other customers by evaluating their reviews through online shopping websites, forums, social networks, etc. The customers' opinions are generally presented in an unstructured and shapeless (free-text) format and user star-level rating of out of five [1] [2].

It is important to interpret these data correctly, as it exposes everything from purchasing patterns to product defects and offers a major competitive advantage. It will also expand business opportunities to discover consumer preferences, product improvements, and marketing insights [3]. Converting this unstructured online content into structured data requires computational approaches such as Machine Learning (ML), Deep Learning (DL), Natural Language Processing (NLP), and many more that will be elaborated on in this research.

This study aims to apply these methodologies and comes up with a solution that predicts the product rating based on online reviews that customers share on shopping websites such as Amazon. Towards this goal, this research intends to explore the analytical aspects such as NLP and DL that will derive meaningful insights from UGC. In this research, a specific question is going to be analyzed: Is it possible to predict the popularity of a product via its rating score using solely the product features?

1.1. Thesis Objectives and Organization

To tackle the research question, the following objectives are defined, implemented, and accomplished in the upcoming sections:

(1) To create a new dataset of a product (laptop) including its major features obtained from product details and descriptions published on Amazon web pages by the manufacturers, and its average/overall rating score.

(2) To predict the average/overall rating score of the product based on its general features by applying suitable DL benchmark algorithms.

(3) To identify additional features that the majority of customers cited in their reviews by applying NLP techniques to review content.

(4) To create a corpus of reviews, which is obtained from the Amazon website on the same category of product (laptop), to extract custom features.

(5) To detect the most influential features (both general and custom) in shaping product rating scores.

(6) To predict the average/overall rating score of the product based on the influential features by applying DL benchmark algorithms.

(7) To compare the results in benchmark models with each other and comparable research on similar topics, in terms of performance metrics.

The remainder of this manuscript is organized as follows. Section 2 presents related works about rating prediction and product feature extraction. Thesis contributions are discussed in Section 3. Section 4 describes the methodology, comprising of different algorithms employed in this research as well as various performance metrics applied to measure their competence. The data collection and preprocessing are given in Section 5. The baseline algorithms, the experiment setup, and results are provided in Section 6. The discussion and analysis of the results are presented in Section 7, Section 8 concludes this research and finally, Section 9 provides the direction for future studies.

2. Review of related literature

Before delving into the review of related literature, some relevant fundamental concepts will be discussed in order to better comprehend the proposed algorithms explored in each phase of the proposed methodology.

2.1. Machine Learning

ML is referred to as the science and art of programming computers to learn from data [4]. ML term generally describes a variety of computer-based data mining techniques to detect complex patterns, mainly in large and complex datasets, intended to provide insights for prediction, classification, and decision-making purposes [5]. The ability to learn from the environment is the most important aspect of designing a successful ML application. According to the nature of the data labeling, ML algorithms are usually divided into supervised, unsupervised, and semi-supervised learning [6]. These categories are depicted in Figure 1 adapted from [6]. In this research, supervised learning is applied to predict the desired output. In supervised learning, the intended output referred to as labels, is included in the training data that is given to the algorithm. Training data consists of samples that teach the algorithm to accurately predict output when it encounters new data that it has never seen before, which is referred to as test data. The spam filter is an excellent illustration of this: it is trained to identify new emails using a large number of sample emails and their classification (spam or not spam) [4].

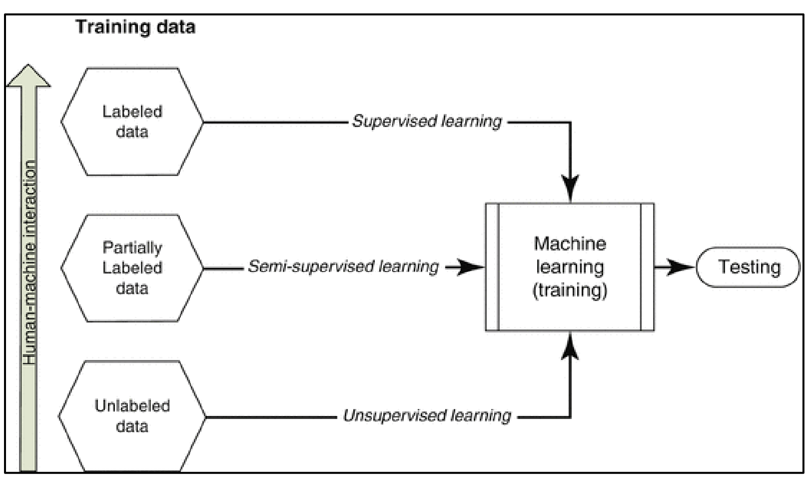

Figure 1. Categories of machine learning algorithms according to training data nature [6]

One of the most important supervised learning algorithms is Neural Networks, also known as artificial neural networks (ANNs), which are a subcategory of machine learning and are at the heart of deep learning algorithms. Deep learning is a branch of machine learning, with neural networks serving as the foundation of deep learning algorithms. Figure 2 illustrates how machine learning, neural networks, and deep earning are related.

ANNs are inspired by the brain's first models of sensory processing. We can train the network to handle a wide range of problems by using algorithms that imitate the activities of actual neurons [44]. An ANN is made up of node layers, which include an input layer, one or more hidden layers, and an output layer. Each node is linked to another and has its own weight and threshold. If the output of any individual node exceeds the defined threshold value, that node is activated and sends data to the next layer of the network and the output of one node becomes the input of the next node. The mathematical logic of this algorithm will be fully explained in the methodology section.

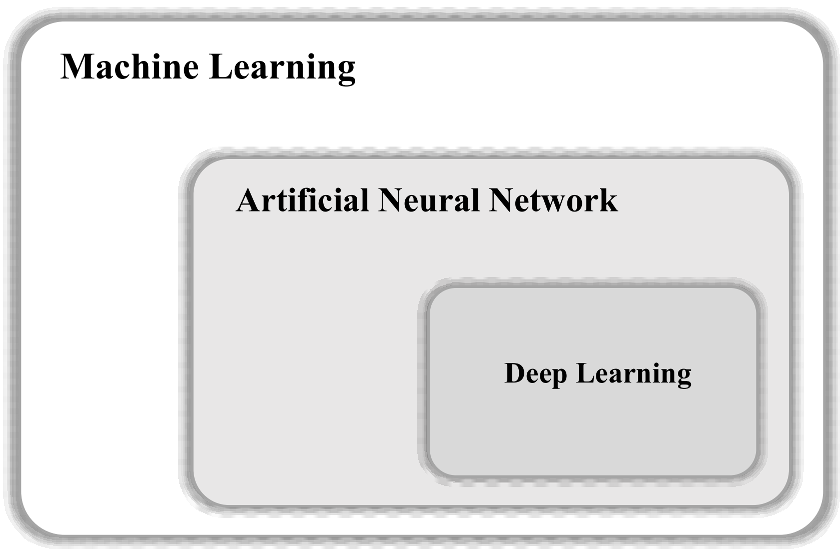

Figure 2. Relations between machine learning, neural networks, and deep learning

2.2. Deep Learning

Deep Learning (DL), which stems from the study of Artificial Neural Networks (ANN), is considered an evolution of ML methods. It combines low-level features to form a more abstract high-level representation. These ANNs are made up of multiple layers, with each layer receiving the output of the previous layer, executing a unique function, and then transferring its output to the next layer [7]. The term "deep" in deep learning refers to the number of layers in a neural network. A deep learning algorithm can be defined as a neural network with more than one hidden layer. Figure 3 displays a Deep Neural Networks (DNN) structure as an illustration. DL also promises improvements in prediction performance as compared to traditional ML models [8]. To examine the potential of DL to predict risky retail investors in a financial risk behavior forecasting case study, Kim et al. [9] employed a DNN for operational risk forecasting which confirmed the feature learning capability of DL algorithms. This procedure provided guidance on designing suitable network architecture and demonstrated the advantage of DL over ML and rule-based benchmarks. DL has been also used for text categorization. Kraus et al. [10] reviewed DL in the area of business analytics and investigated its performance in operations research across different scenarios with real data from entrepreneurs. Based on the empirical results of different cases they suggested that DL is a feasible and effective method, which can considerably and consistently outperform traditional ML counterparts in prediction performance from the family of data-analytic models. Wang et al. [11] proved that when DL methods, such as Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), an ANN architecture, and DNN are compared with traditional methods (Catboost[1], XGboost[2], Lightgbm[3]), for marketing intention detection, the neural network-based (NN-based) approaches are superior to the traditional methods based on all data sets. They also demonstrated that the NN-based models can efficiently construct the semantic representation of the text under investigation. The F1 score of their proposed model based on two test sets was respectively 71% and 73% which are higher than other benchmark models.

1 Open-source gradient boosting on decision trees library [67]
2 Optimized distributed gradient boosting library [68]
3 Gradient boosting framework that uses tree-based learning algorithms [69]

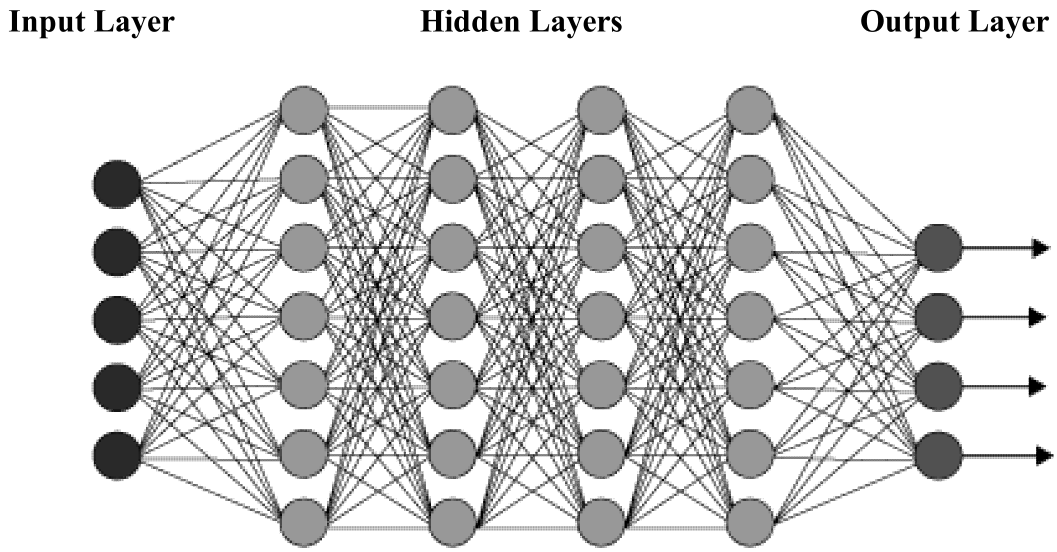

Figure 3. Deep Neural Network Structure

2.3. Natural Language Processing

The two main components of an online review that influence consumer choices are ratings and content reviews. Researchers use a programmed approach to extract useful information from the content [12]. Natural language processing (NLP) refers to a set of computing approaches for the automatic analysis and representation of human languages [13]. NLP [14], basically helps to interpret a text by the computer. The significance of NLP stems from the fact that the World Wide Web (WWW) has a massive quantity of data, at least 20 billion pages, which can be exploited as a valuable resource [13]. NLP applications include but are not limited to a variety of business purposes such as sentiment analysis, market intelligence, neural machine translation, information retrieval (IR), classification of text into categories, information extraction (IE), creditworthiness assessment, and so on. Sentiment analysis is an ongoing field of research that can be used to identify people's views, attitudes, and feelings about an item. This item can be an individual, an event, a topic, or a product and is most likely to be covered by reviews [15]. Other than identifying sentiment polarity (negative, neutral, or positive), NLP applications help to extract a lot of valuable information from reviews. Perceiving the features that customers care about the most about a product and eventually discussing them in their reviews is a good example of NLP that has been explored in this research.

Until choosing to dig deeper and read the text in reviews, almost all shoppers who are reading reviews, pay attention to the rating score. Rating scores are one of the widely available forms of user feedback that has been found to significantly influence the shopping behavior of users, a measure that represents the contents of such reviews numerically [16]. In this regard, Review Rating Prediction (RRP) became a prevalent topic in the area of online review analysis. The predicted rating reflects an estimated user's satisfaction with an item and it is usually presented on a 1–5 scale, where a rating value of 5 means the highest and a rating value of 1 means the lowest satisfactory rate. The existing rating scores represent users' behavior towards products or services which is the basis of review rating prediction [17]. Many researchers apply sentiment analysis to reviews to find out the polarity of the customer's feelings (negative, neutral, or positive) toward the product and use it to predict the rating the customer will assign. The authors in [15] illustrated the process of identifying sentiment polarity as in Figure 4.

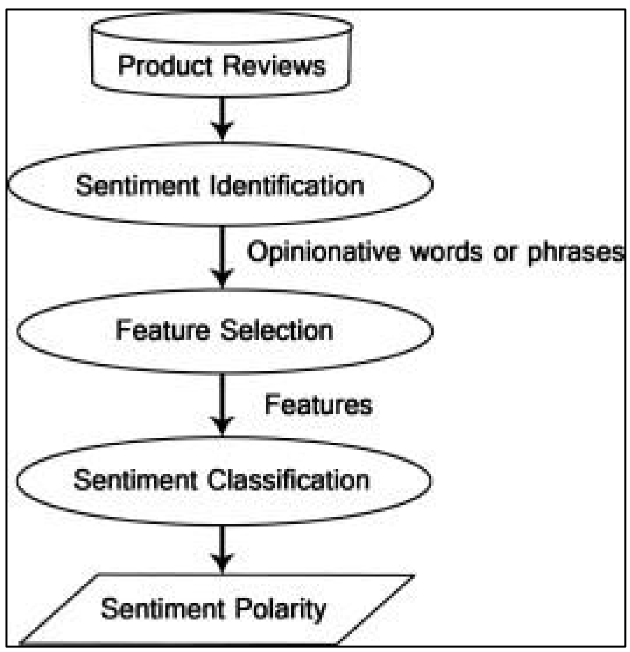

Figure 4. Sentiment analysis process on product reviews [15]

Given the main focus of this research on product feature extraction and rating prediction, the literature review is divided into two main sections. The first section focuses on state-of-the-art rating prediction models and the second part reviews the existing approaches for product feature extraction from online reviews.

2.4. Product Rating Prediction

The literature on online reviews analysis is composed of several areas such as review helpfulness prediction, feature extraction, reviews sentiment analysis, opinion mining, customer retention, review rating prediction and so on which usually follow the same paradigm [18]. Wang et al. [19] proposed a review rating prediction method based on user context and product context by integrating user information and product information into review texts. Their method comprises a global review, a user-specific review, and a product-specific rating prediction model. The first model is a global review rating prediction, and it can be learned from training datasets of all users and all products, and it is shared by them. The second model is a user-specific review rating prediction, which learns the user's personalized sentiment information from the training data of an individual user. The third model is a product-specific review rating prediction, which learns the parameters of the model using individual product training datasets. To learn the parameter of these models, the least-squares error loss principle and the stochastic gradient descent algorithm were used. The mean absolute error (MAE) of their proposed model (review rating prediction based on user context and product context) is reduced by 14%-17% compared to baseline models.

Ahmed et al. [1] in a recent work proposed a review rating prediction framework using deep learning. The framework consists of two phases based on DL bidirectional gated recurrent unit (Bi-GRU) model architectures. Bi-GRU neural networks link two hidden layers with opposing output directions. The output layer of this type of generative deep learning can receive input from both the past (backward) and future (forward) states at the same time [20]. The first phase of their research is used for polarity prediction, and the second phase is used to predict review ratings from the review text. The experimental results demonstrated that their proposed framework can improve the rating prediction in terms of precision, recall, and F1-score by 2%-6% and reduce the root mean square error (RMSE) by 13%-27%, comparing with baseline approaches on balanced datasets. Cao et al. [18] proposed a new review semantics-based model to enhance the performance of the review-based recommender. Their model includes the review semantics extractor, the review semantics generator, and the rating regressor. The review semantics extractor extracts the semantic features of a particular review text using a CNN. CNN is a multiple-layer neural network approach for learning hierarchical data characteristics. CNN has advanced considerably in the design and computation of NLP in recent years [21]. After extracting the semantic features, the semantics generator uses a memory network, such as the structure and attention mechanism to simulate the decision-making process. The generated semantics is then compared with the semantics extracted by the review semantics extractor. Finally, the generated semantic features are fed into the rating regressor to predict the overall rating. Their review semantics-based model (RSBM) could reduce the MSE by 1%-6% in all datasets from Amazon and Yelp, compared to baseline models.

Rating prediction is also one of the most critical tasks for recommender systems (RS) [17]. Hasan Zadeh et al. [22] developed a review-based rating prediction system of information by applying user reviews and rating scores. Their proposed model handles the uncertainty problem of the rating histories, by fuzzifying the given ratings. Instead of using traditional models such as Bag of Words (BOW) and Term Frequency-Inverse Document Frequency (TFIDF), the authors utilized a word embedding representation model for textual reviews which makes use of the helpfulness voting scores to prune data. To construct a semantic vector for each of the reviews, they used the Doc2Vec embedding model [23] which refers to the process of generating the aggregated vector created for the whole review. To create the recommender system, they labeled every predicted high score (>3) rating by '+', while the label of '-' was assigned to the others. The final recommendation list consists of a set of products with positive labels. Finally, they revealed that the proposed recommender system outperforms its counterparts such as the rating-based K-Nearest Neighbor (KNN), based on the sentiment polarity of reviews, and user-personalized review rating prediction [24]. Their proposed model could significantly reduce MAE to 0.082, comparing baseline models.

2.5. Product Feature Extraction

The existing rating prediction approaches are mainly based on review contents and employ only a single model to interpret the reviews' sentiments, ignoring products' features that are being reviewed. Nevertheless, product features have a significant influence on review rating prediction.

Shrestha et al. [25] employed paragraph vectors to learn the syntactic and semantic relationship of a review text. They grouped and sorted review embedding to form a product sequence which is fed to a gated recurrent unit (GRU) to learn product embedding. The concatenation of review embedding generated from paragraph vectors and product embedding generated from GRU is used to train a support vector machine (SVM) for sentiment classification. The authors demonstrated that with only review embedding their proposed model performed at an accuracy of 81.29% and the inclusion of product embedding increased the accuracy to 81.82%. This shows that product information is a powerful feature that can be employed in sentiment analysis. They also used this classifier through a web service to predict the rating of a review and compare it against a given rating. This web service takes review text and review rating and provides a warning to the reviewer if there is any inconsistency between the given rating and review.

In [26], the authors proposed new methods for extracting product features from online consumer reviews based on natural language processing (NLP) and machine learning techniques. Their proposed models identify new features and filter irrelevant features through a classification system based on subjective features and objective features. Subjective features are those features that appear in subjective statements where reviewers reflect their sentiments explicitly. Whereas, objective features, such as brand names or special models, appear in objective statements that don't evolve reviewers' positive or negative opinions. To extract subjective features, they employed double propagation [27], a recognized grammar-based technique, which uses a dependency parser to detect the opinion word and features, and comparison patterns. To extract objective features, they identified different patterns (e.g., Noun Phrase (NP) + Verb + NP) in the text and implemented the appropriate algorithm regarding each pattern. Then they applied the frequency filtering method (e.g., TFIDF), textual, and semantically similarity to prune the extracted features in the previous stage. Compared to Double propagation, their proposed model achieved a higher recall score of 86% due to the fact that they considered both subjective and objective features.

Wang et al. [12], extracted product features from the online product description and customer reviews, employing a Kansai text mining approach. The extraction process was done by NLP tools such as regular expression based on detection of punctuations, and Tree Tagger which is a probabilistic Part-of-Speech (PoS) tagger. They also used syntactic rules to extract nouns and noun phrases from the text as the candidate features, since important product features are always expressed as nouns and noun phrases. In order to group the candidate features, they use heuristic rules and a semantic database (i.e., WordNet) to analyze the parent-child and is-neighbor relationships among the candidate features. Although the performance of their proposed model was different in positive and negative reviews, their model could outperform baselines in precision but led to lower results in F1 score and accuracy.

Another recent widely used subtask of information extraction is Named Entity Recognition (NER). NER is one of the most important components of NLP systems for question answering [28], information retrieval [29], and machine translation [30], among other algorithms. NER systems have been studied and developed extensively for decades, but accurate systems based on Deep Neural Networks have just recently been established [31]. The majority of NER research has been structured by taking an unlabeled block of text and creating an annotated block of text that identifies the names of entities. Examples of named entities are: "Person", "Location", "Organization", or "Dates" but one can define customized named entities and be able to label any new categories mentioned within the unstructured text. Shelar et al. [32] compared different techniques and algorithms for creating custom NER models. The authors used the IOB[4] tagging format and LSTM to create the NER model and evaluated their customized algorithms in terms of accuracy, F-score, prediction time, model size, and ease of training. LSTM is a type of recurrent neural network (RNN). In RNN, the output from the previous step is used as input in the next phase [33]. LSTM is created to address the problem of RNN long-term dependency, in which the

[4] Short for Inside, Outside, Beginning

RNN cannot predict the word stored in long-term memory but can offer more accurate predictions based on recent information. RNN does not function efficiently as the gap length increases while LSTM can keep information for a long time.

In general, NER models are user-friendly and straightforward algorithms that now have many pre-trained libraries for some specific entities in Python and Java [34].

2.6. The Existing Gaps in The Literature

Despite a host of research on online consumer reviews, there is still a great demand for research to improve the techniques for predicting customers' behavior towards their purchases. Most of the existing related literature, predicts the review rating score by identifying the sentiment polarity of the reviews (sentiment analysis) [1][16][25][35]. In other words, current studies fail to incorporate product features as an input to the overall rating prediction algorithm. More specifically, sentiment analysis is not suitable for this research given that our main objective is to assist businesses to predict the overall rating of a new product based on its features before launching it to the market. This is expected to provide a good indication of the product's future sales potential before it is manufactured. At this point, no reviews have been posted about that product; therefore, the existing sentiment analysis approaches are not applicable.

Furthermore, in previous studies, product features are usually extracted from product details and descriptions published on the web pages by the manufacturers. Although these features are still important ones, this study focuses on other product aspects that consumers have highlighted the most in their comments and that are most important to them. In addition, to extract product features from online product reviews, the traditional approaches are time-consuming for cleaning and preprocessing the data [2][12][19][26]. Furthermore, they are usually applied to identify noun words as candidates for feature names. However, implementing DL-NLP algorithms, such as Named entity recognition (NER) [36] which simplifies the task of extracting custom features seems to be a more promising path that will be pursued in this study.

3. Thesis contributions

This work is focusing on predicting the popularity of the product based on its overall rating. The overall rating score is the average of every single rating coming along with the product review. Unlike previous works, instead of predicting the rating based on sentiment analysis and polarity of the review, the overall rating score of the product is going to be predicted based on general and custom product features. Nevertheless, customer reviews still play a significant role and are used indirectly in the prediction model. The product features which are the main attributes of shaping customer satisfaction will be extracted from the text of the review. These features in combination with other features extracted from product general information will act as the inputs to the predictive overall score rating model. In this manner, the effect of product features in determining its popularity will be directly measured and analyzed.

This idea originates from the fact that reviews are mainly generated from users' personal experiences of consuming the product and the key elements of shaping the satisfaction or dissatisfaction of a consumer are basically product features. Hence, these features are going to be directly utilized to predict the product's overall rating. For example, Figure 5 shows different types of laptops with different features and their own rating scores, provided on Amazon.com. The record shows more than 10,000 results for this specific product. Despite the exterior differences, these products share some mutual features that will be employed in the training set to predict the product's final rating score.

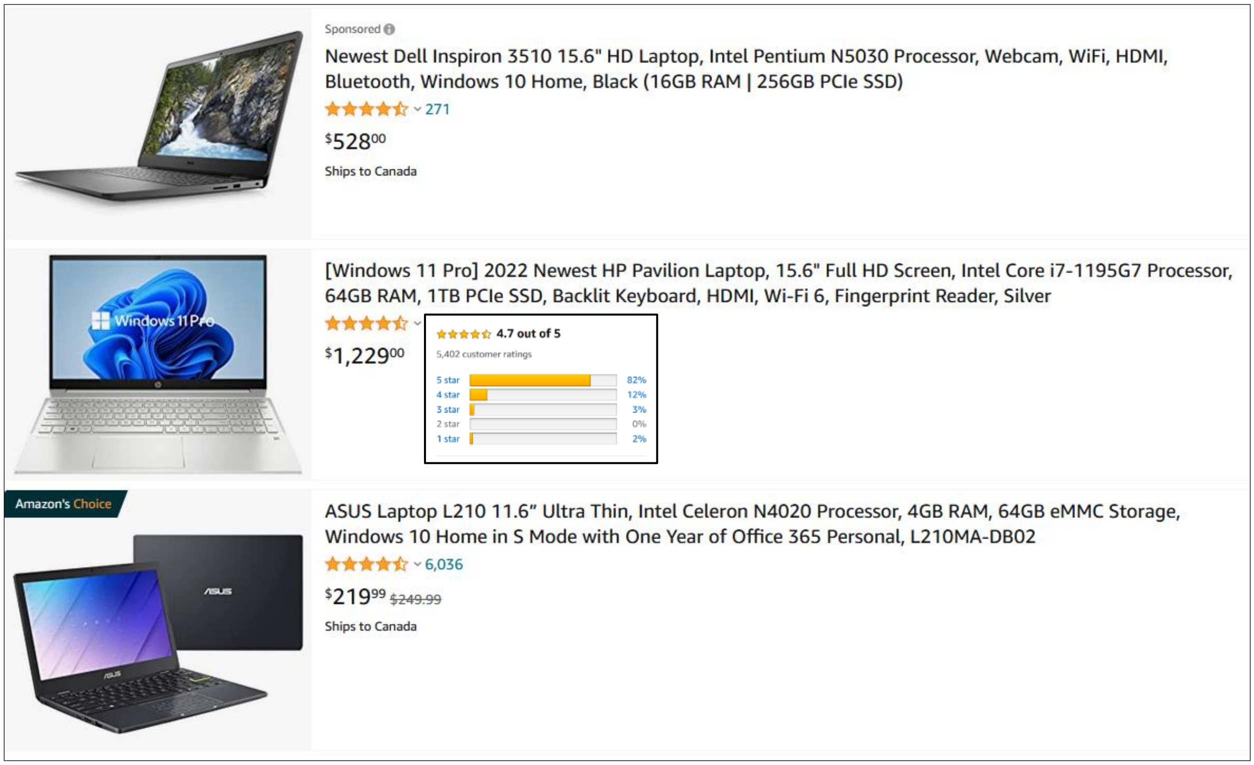

Figure 5. Final rating scores of different products in the same category (Laptop) [37]

The main contribution of this study can be summarized as follows:

1. We first develop a general product rating prediction model, Figure 6, using general features extracted from online product information. The rating can be learned from training datasets of all different types of a specific category of products that share mutual features. Deep learning algorithms (Deep Neural Network algorithms) are explored to predict the final product rating score.
2. The second contribution relies on extracting custom product features from the reviews as summarized in Figure 7. In this model additional features that actually matter to customers, are extracted from users' reviews content by applying NLP techniques (NER Algorithm).
3. The third contribution focuses on the combination of the first and second approaches, Figure 8, to predict the final product rating score. Furthermore, feature selection methods are employed to select the most relevant features in order to reduce the dimensionality of feature vector spaces and provide the highest level of accuracy.

4. Due to the lack of a benchmark dataset, a collection of products of the same product family (laptops) is gathered from the Amazon website to validate the model developed in phase 1. Likewise, a second dataset of customer reviews in the same product category is prepared to fulfill the objectives of the phase 2 model. A view of both datasets is provided in the Appendix section.
5. Finally, the combined method is compared with earlier works with respect to performance metrics such as accuracy, sensitivity, specificity, and F1 score to check the ability of the proposed model in outperforming traditional approaches.

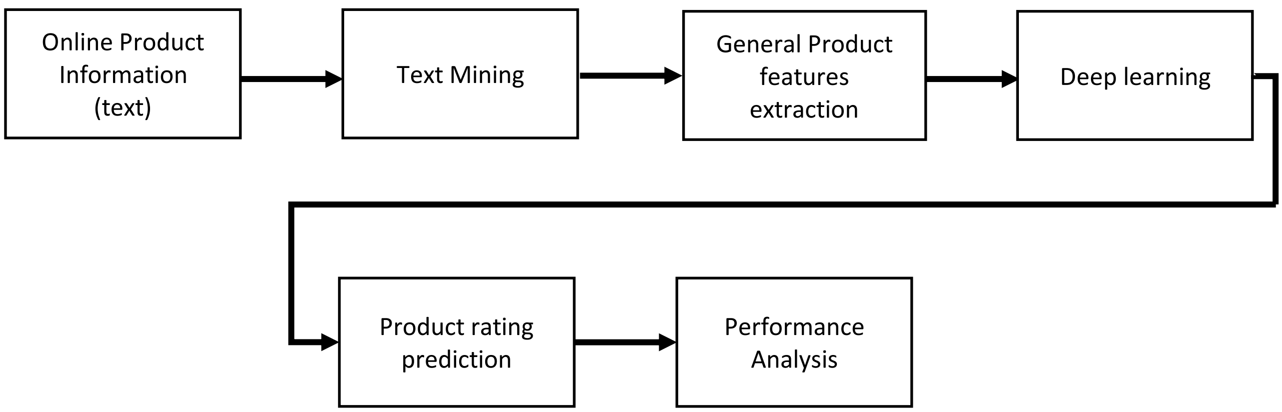

Figure 6. Proposed model phase 1: General product rating prediction

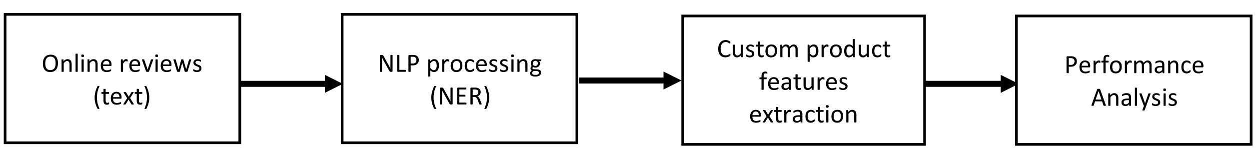

Figure 7. Proposed model phase 2: Product rating prediction based on extracted features from reviews

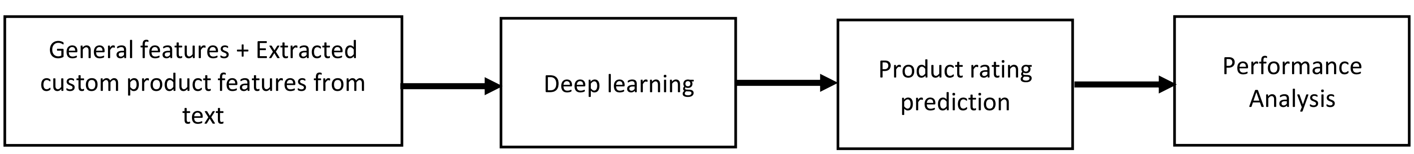

Figure 8. Proposed model phase 3: The combination of the first and second approaches to predict the final product rating score

4. Methodology

This section describes the proposed methodology in detail. The first subsection gives an overview of the system architecture. Afterward, the data gathering process, product feature extraction from online information (Phase 1), then custom feature extraction from reviews (Phase 2), and lastly combination of the most influential features to predict the final rating scores are addressed (Phase 3).

4.1. System Architecture

The general structure of each phase of the proposed methodology is shown in Figure 6 toFigure 8. In Phase 1, a simple product rating prediction is conducted using Deep Learning. To this end, a list of product information is scraped from Amazon webpages, and the extracted common product features are fed to the training model as the input. Then in Phase 2, in order to identify features that customers care about the most, the NER algorithm within NLP applications has been applied to the reviews and this time custom product features are extracted (Figure 7). Finally, in Phase 3, the combination of the general and custom features is fed into the DL algorithm to predict the average rating score of the product. The efficiency of each model is measured through performance metrics to quantify the ability of the models in predicting and verifying whether the final model can outperform traditional approaches.

4.2. Product Feature Extraction

The first step in the process of extracting product features is collecting product information from the Amazon website. In order to retrieve this information, a web scraping tool named Data Miner is utilized. Data Miner is a Google Chrome and Edge Browser Extension that allows the user to crawl and scrape data from websites and save it as a CSV or Excel spreadsheet [38]. With this application, one can create a custom crawler to extract desired information from multiple web pages.

4.2.1. Extracting Product Features from the Online Information

To accomplish the goal regarding the phase 1 model, we first need to extract general laptop features such as brand names, size of the screen, or hard disk capacity from the online product information provided by the manufacturers or distributors on Amazon web pages as shown in Figure 5. After scraping, a list comprising of product technical description, price of the product, rating score, and the number of customers who had rated the products are extracted. By preprocessing and applying Python text mining libraries such as RegEx [39], the detailed technical product features are extracted from the description column. A regular expression (RegEx) is a string of characters that indicates a search pattern in the text. For example, to identify whether the description set contains the information regarding hard disk drive (HDD) or solid-state drive (SSD) capacity, we can filter out these two strings "HDD" and "SDD" in the description by the pattern r’ (?:HDD|SSD)’. Then more complicated patterns can be conducted to determine whether these two strings are accompanying a number as the amount of capacity. With the help of regular expression and some simple coding in Python, the desired features can be extracted. These features act as the input to the model developed in Phase 1.

4.2.2. Named Entity Recognition

In order to extract custom product features from online product reviews, traditional approaches usually use Bag of Words (BOW), uni/bigram features, Part of Speech (PoS) tags [40] to identify noun words as candidates of feature names or utilize TFIDF vector space to train the ML models. However, in the past few years, DL has gained considerable success in many NLP tasks, including Named entity recognition (NER) [36]. NER aims to identify pre-defined name entities such as person, location, organization, etc. in a text [31]. The difference between this approach and the previous one is that here the custom product features that users care about the most are unknown, and going to be identified by review analysis, while in the previous approach, the general features are pre-defined.

In this study, a significant development in the NLP area, namely, the novel approach of Transformers, and more specifically, Simple Transformers is applied. Transformers are built to operate with sequence data, taking an input sequence and generating an output sequence, one element at a time. A transformer, for example, may be used to translate an English sentence to a French one. The Simple Transformers, on the other hand, is a library that is intended to make using Transformer models easier without compromising functionality [41]. The model we used from the Simple Transformers library is called "Named Entity Recognition", where the model type is set to "bert" and the model's name is set to "bert-base-cased". Bert stands for Bidirectional Encoder Representations and is the state-of-the-art pretraining approach and a transformer-based ML technique that directly takes in words as input, by converting them into numbers [42].

The initial step in developing an NER model is training the model using the training data, followed by evaluating the model using the testing data, and lastly making predictions on unlabelled data [41]. The NER model's default label list is obtained from the CoNLL dataset [43], such as person, location, organization, etc. However, it is possible to define custom labels and pass them in when creating the model.

In order to train the NER model, a new dataset on online product reviews is annotated with product features. This dataset was prepared in accordance with the CoNLL2003 dataset and BILOU standards. BILOU is an annotation technique that indicates boundary tokens explicitly. Learning classifiers that recognize the Outside of text segments, the Beginning, Inside, and Last tokens of multi-token chunks, as well as Unit-length chunks, is suggested by the BILOU scheme. For instance, "connecting 4K monitor" is labeled as Connecting: "O", 4K: "B-Feature", and Monitor: "L-Feature". According to Ratinov et al.[44], the simplest IOB scheme (Begin, In, Out) is more difficult to learn than the BILUO scheme which clearly indicates boundary tokens.

4.3. Artificial Neural Networks

Neural networks, also known as artificial neural networks (ANNs), are a subcategory of machine learning and are at the heart of deep learning algorithms. ANNs are inspired by the brain's first models of sensory processing. We can train the network to handle a wide range of problems by using algorithms that imitate the activities of actual neurons [45]. An ANN is formed by a group of processing units, also known as neurons, that are linked together. Each neuron is a transfer function and can be a nonlinear unit with several inputs and a single output. The architecture of a neural network is defined by the network's connections and the neurons' transfer functions [46]. In this study, three baseline neural network models have been used, which will be explained in-depth in the following subsections.

4.3.1. Deep Neural Network

The model of a deep neural network (DNN) is conceptually based on ANNs [47] that operate in a stage-wise manner with multiple layers between the input and output layers. The number of layers in a network determines its depth. Each layer receives input from previous layers, learns a high-level representation of the input, and passes the representation (i.e., output) to a subsequent layer [9]. In this framework, the product features extracted in the previous step along with the product rating score act as the input that is fed to the network to train the model. The underlying architecture of DNN is presented in Figure 9, containing the input layer (features of different products in the same category), hidden layers, and the output layer which is the rating score class. Each of the layers consists of a variable number of fully connected neurons. In the proposed model, the number of neurons in the input layer is equal to the number of product features (m) and there is one neuron in the output layer as it is a binary classification model. There are a certain number of inputs and weights for each neuron. Each neuron in the hidden layer performs a weighted linear summation on the values from the preceding layer, followed by a non-linear activation function. The values from the last hidden layer are received by the output layer and transformed into output values. Equation (1) shows a set of features where m is the number of dimensions for input. Equation (2) describes the weighted linear summation with w_i as the weight parameter and b the bias, and Equation (3) displays the output of the neuron, where Ø (theta) is the activation function.

$$X = x_1, x_2, \dots, x_m \tag{1}$$

$$Z = \sum_{i=1}^{m}(w_i\, x_i) + b \tag{2}$$

$$O = \emptyset(Z) \tag{3}$$

$$\emptyset(Z) = max(Z, 0) \tag{4}$$

The activation function transforms the weighted summation from the neuron into the activation of the neuron in the next layer. In this study, the activation function used in hidden layers is the rectified linear activation unit, or ReLU [48]. Considering Equation (4), it has simple computation and its linear behavior increases the chances of optimizing the DNN [49].

The DNN's last layer takes input from the last hidden layer, transforms it, and outputs a binary (0 or 1). It is made up of a single neuron that calculates the weighted sum of its input values and utilizes the sigmoid activation function to generate the final output. The sigmoid activation function works perfectly in a binary classification problem [49]. The sigmoid activation function that estimates the probability of y = 1, is represented as Equation (5).

$$\hat{y} = \sigma(Z) = \frac{1}{1+e^{-Z}} \tag{5}$$

Where Z is the final hidden layer output determined as in (3) and (4), $\hat{y}$ is the neuron output, and σ is the sigmoid activation function. The initial information is provided by the inputs X, propagates to the hidden neurons at each layer, and eventually creates the output, which is a number in the range of 0 to 1. To compute the average error across all cases, the cross-entropy loss function [50] is employed which is shown as Equation (6).

$$\ell(y, \hat{y}) = -\sum_{i=1}^{n} y_i \, log\hat{y_i} \tag{6}$$

Where y is the actual value, $\hat{y}$ is the output of the DNN, and $\ell(y, \hat{y})$ is the cross-entropy loss function. The DNN finds a set of weights that minimizes the difference between y and $\hat{y}$, after each forward propagation. The procedure moves forward in the network of neurons; hence it is called a feed-forward neural network. To achieve the least difference, the DNN backpropagates the error information across the layers in order to tune the weights and compute a new $\hat{y}$. This procedure which is called backpropagation computes the gradient of the loss function with respect to the weights, figuring the gradient of each layer at a time, iterating backward from the output layer. On the other hand, optimizers are used for training the neural network, applying the gradients computed with backpropagation. The adaptive moment estimation (Adam) optimizer [51] is employed in this study which is an adaptive learning rate optimizer and a search technique for adjusting the weights of each neuron in the hidden layers [49].

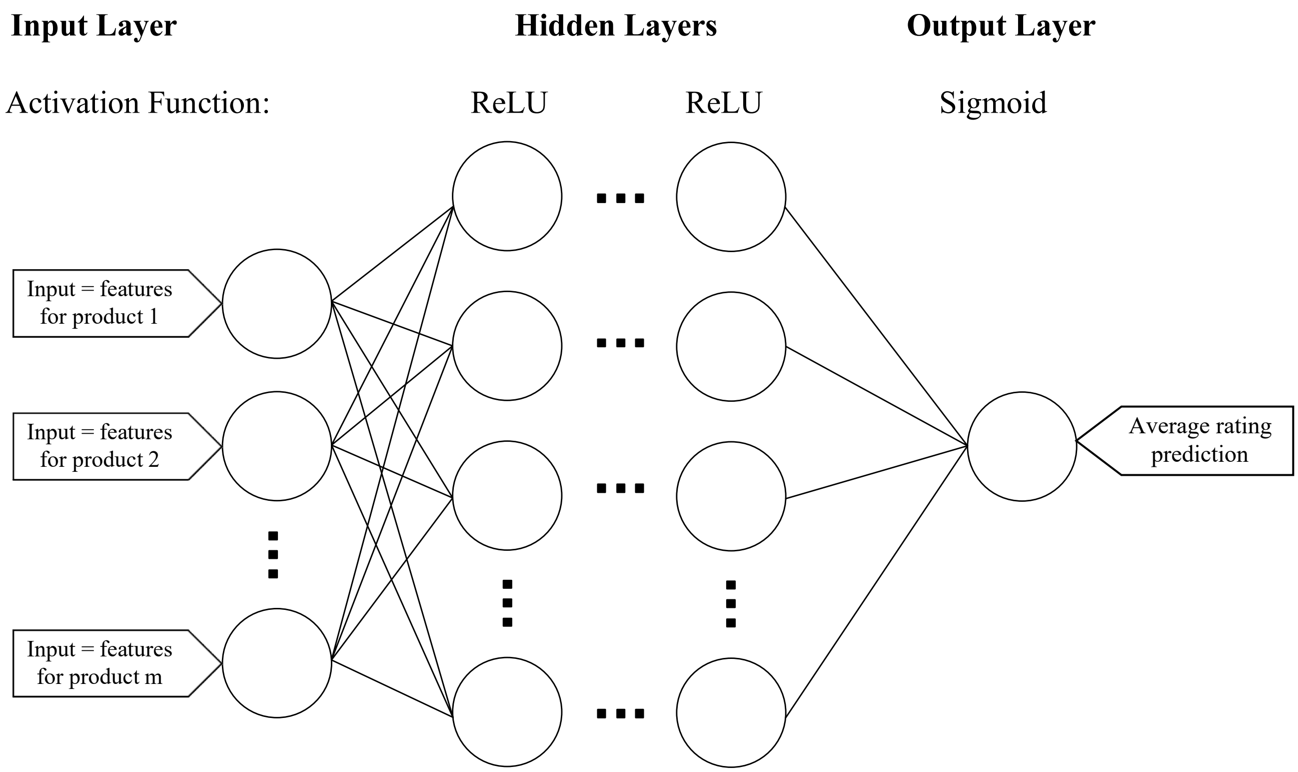

Figure 9. Structure of a deep feedforward neural network

4.3.2. Probabilistic Neural Network (PNN)

A probabilistic neural network (PNN) is a type of feedforward neural network presented by Dr. Specht in 1989 [52] that is used to solve tasks like classification and pattern recognition [53]. When applied to a classification task, these networks use the concept of probability theory to reduce misclassifications. [31]. PNN has a competitive advantage over other neural network models due to the use of Bayesian decision theory and radial basis function (RBF) in its design, as well as the consideration of the cross effect of different pattern types. PNN is capable of converging to the Bayesian classifier without dropping into local minima as the number of massive data increases [54].

The PNN approach approximates the parent probability distribution function (PDF) of each class by estimating a probability density function p(x) from a sample $p(x_n)$ that does not need any previous information or assumptions about the statistical properties. The PDF of each class is then used to estimate the class probability, and the Bayes' rule is used to assign the class with the highest posterior probability to the new data. Specht in [52] defines a basic framework for a probabilistic neural network which is depicted in Figure 10. The network is consisting of four basic layers including the input layer, pattern layer, summation layer, and decision layer.

- Input Layer: The input vector is represented by a pre-processing data set of the training sample, which is distributed to the next layer. The number of its neurons should be the same as the dimension of all the samples.
- Pattern Layer: The Euclidean distance between the feature vector of training sample X and the radial center x_{ij} is employed in the pattern layer to achieve matching between the input feature vector and various types of training sets. Equation (7) shows the Euclidean distance between the feature vector of training sample X and the radial center x_{ij} where: $X = [x_1, x_2, x_3, \dots, x_n]^T, n = 1, 2, \dots, l$, and l is for all types of training. x_{ij} is the j-th center of the i-th training sample, d is the dimension of the eigenvector, and σ is a smoothing factor.
- Summation Layer: This layer calculates the average of the pattern units' output for each class. Each class has its own neuron which is linked to all neurons in the pattern layer of that class. Equation (8) is the averaging L patterns of class i where υ_i is the output for class i neurons and L is the number of class i neurons.
- Output Layer: This layer takes the maximum value from the summation layer and assigns it to the appropriate class label. Equation (9) selects the class that gives maximum output in the summation layer and $\mathrm{Type}(\upsilon_i)$ is the output type of the output layer.

$$\Phi = \frac{1}{(2\pi)^{\frac{d}{2}}\sigma^d} e^{-\frac{\left(X-x_{ij}\right)^T (X-x_{ij})}{2\sigma^2}} \tag{7}$$

$$\upsilon_i = \frac{\sum_{j=1}^{L} \Phi_{ij}}{L} \tag{8}$$

$$Type(\upsilon_i) = argmax(\upsilon_i) \tag{9}$$

σ is determined by the sample density. For each dimension or feature, the easiest way is to utilize the standard deviation of training samples.

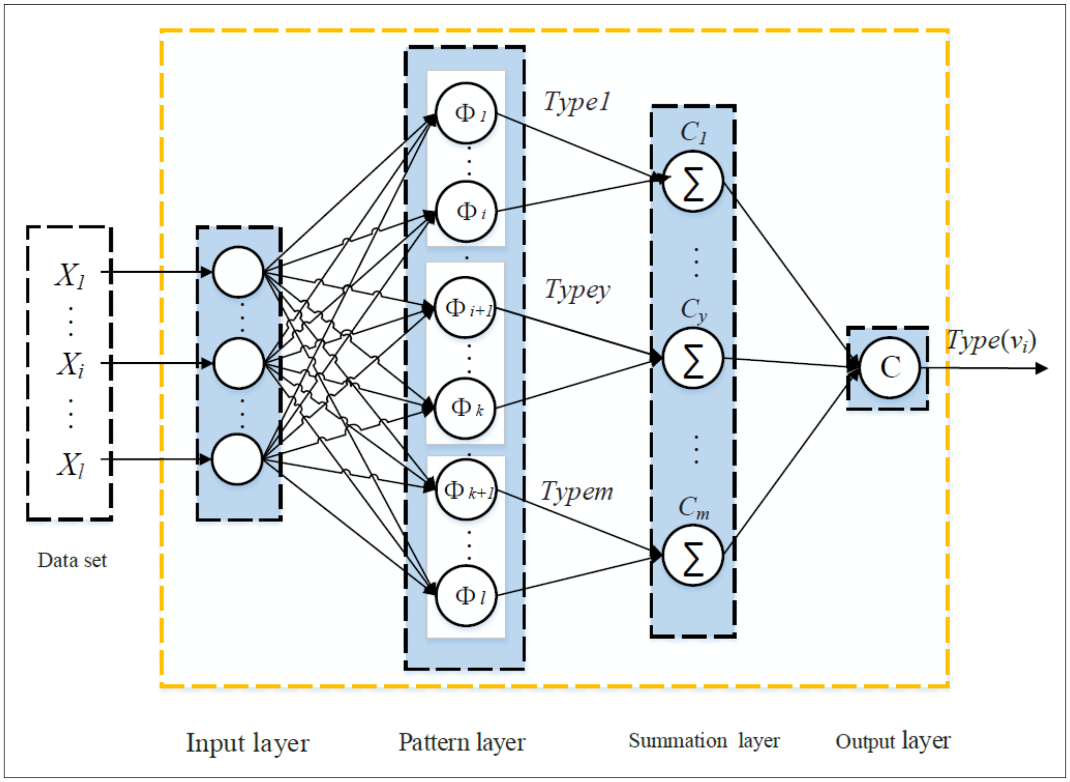

Figure 10. The detailed structure of probabilistic neural networks [54]

In classification applications, PNNs provide a robust alternative to traditional back-propagation neural networks by eliminating the requirement for enormous forward and backward calculations [53]. Therefore, there is no considerable training calculation time associated with back-propagation networks in PNNs.

4.3.3. Radial Basis Function Neural Network (RBFNN)

A radial basis function neural network (RBFNN) is a form of a feed-forward neural network, as well. A radial basis function is any function that is specified as a function of distance from a certain central point (a radius). RBFNN consists of three layers: the input layer, the hidden layer, and the output layer [55]. The hidden layer is the most important component of the RBFNN, as it achieves nonlinear hyper-separation and function expansion. The activation function in the hidden layer which is the radial basis function is usually the Gaussian, Equation (10), where $\|x - c_i\|^2$ is the Euclidean distance between the feature vector of training sample X and the i-th hidden center (c_i), and σ is i-th neuron's bandwidth or the variance and Φ_i is the i-th neuron's output from the hidden layer.

$$\Phi_i = \exp\left(-\frac{\|x-c_i\|^2}{2\sigma_i{}^2}\right) \tag{10}$$

$$y = \sum_{i=1}^{N} w_i \Phi_i \tag{11}$$

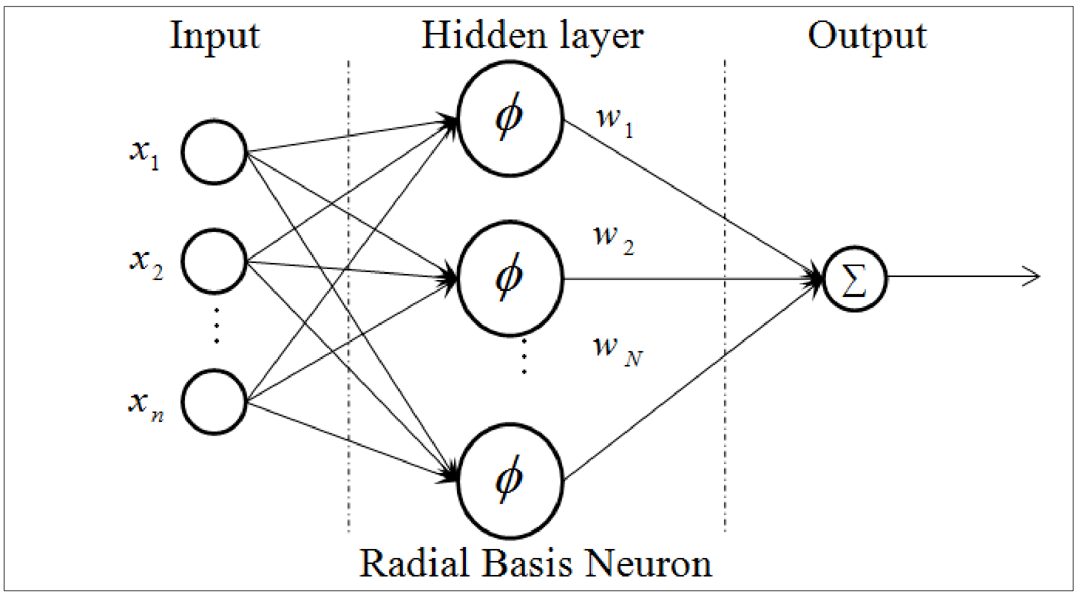

Figure 11. Radial Basis Function Neural Network Structure [56]

The accuracy of RBFN is mostly determined by the initial centers chosen from the dataset before network training begins. One of the widely used procedures in centers selection is clustering algorithms such as K-means as it is the fastest, least complicated method that produces acceptable accuracy for center selection when compared to other existing clustering algorithms [57]. K-means algorithm is an algorithm to classify the dataset based on features into K number of groups by minimizing the sum squares of distances between data and the associated cluster center (c_i in Equation (10)). The output of the RBFNN can be expressed as Equation (11), where w_i is the weight of neuron i in the linear output neuron. The overview of the RBFNN structure is shown in Figure 11 adopted from [56].

The RBFNN and PNN have similar structures; the only difference is that the PNN has one more hidden layer than the RBFNN. The PNN also has an advantage over other RBFNN in terms of training speed [58]. Table 1 provided a detailed comparison of the three benchmark methods used in this study [59]. The final results are elaborated on in the upcoming sections.

Criterion	DNN	PNN	RBFNN
Architecture	An input layer, more than one hidden layer, an output layer	An input layer, a pattern layer (hidden layer), a summation layer, an output layer	An input layer, one hidden layer, an output layer
Activation Function	A non-linear function (ReLU, sigmoid, …)	The activation function is based on the probability density function	A radial basis function, usually the Gaussian, calculates the Euclidean distance between the feature vector and hidden centers
Number of Hidden Neurons	No defined principle to determine the number of neurons	Equal to the number of samples	No defined principle to determine the number of neurons
Output Layer	Uses an activation function before linearly combining it	Choose the maximum of the computed probabilities	Linearly sums up the output of the previous neuron
Training Process	Backpropagation	Bayesian decision rule	Clustering

Table 1. Similarities and differences of three neural networks algorithms (DNN, PNN, RBFNN)

4.4. Performance measures

The baseline methods were evaluated in each phase of the proposed methodology. In all stages, the performance was measured using the metrics of Accuracy, Recall or Sensitivity, Specificity, Precision, and F1-score. These performance measurements are basically evaluated from True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN).

Accuracy is the ratio of the number of correct predictions to the total number of input samples. Sensitivity, also known as Recall or true positive rate is the probability that a randomly selected known positive will be correctly classified as positive. Specificity, also known as the true negative rate is the probability that a randomly-selected known negative will be incorrectly classified as positive. Precision is the ratio of the number of true positives to the total number of positive predictions and it measures the quality of a positive prediction made by the model. Lastly, F1-score is the harmonic mean of precision and recall. These metrics are shown in Equations (12) to (16).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{12}$$

$$Specificity = \frac{TN}{TN + FP} \tag{13}$$

$$Precision = \frac{TP}{TP + FP} \tag{14}$$

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{15}$$

5. Data Collection and Preprocessing

In this research, three separate datasets are prepared to demonstrate the effectiveness of each product rating prediction model. Due to the lack of a benchmark dataset, a collection of products from the same product family (laptops) is gathered from the Amazon website to validate the model developed in phase 1. After scraping webpages to extract product information using Data Miner customized recipe scraper, collected data was extracted to a CSV file. A sample of extracted information is depicted in Figure 12. Then the data was cleaned by removing duplicate lines, followed by extracting general features by using RegEx and simple coding in Python. After the preprocessing, the dataset containing 2028 laptops with unique features was selected. The list of extracted general features is provided in Table 2. The overall rating is the average rating that one product received in total. Since for this dataset all the overall ratings were between 3 to 5, two classes of rating have been defined; class 1: (3-4), and class 2: (4-5) which made the model a binary classification. The last row of Table 2 is the overall rating which is labeled 0 if it is in class 1 and labeled 1 if it is in class 2.

	Description	Rating Score	Price	#Ratings
0	ASUS VivoBook 17.3" FHD IPS LED Backlight Prem...	4.4 out of 5 stars	487	8|More Buying Choices
1	Acer Aspire 5 Slim Laptop, 15.6" Full HD IPS D...	4.7 out of 5 stars	593	1012
2	HP 14z Home and Business Laptop Google Classro...	4.6 out of 5 stars	399	67|Price may vary by color
3	2020 Acer Aspire 3 15.6" Full HD 1080P Laptop ...	4.3 out of 5 stars	513	46|More Buying Choices
4	2021 Newest HP 15.6 FHD IPS Flagship Laptop, 1...	4.8 out of 5 stars	869	12
5	CHUWI HeroBook Pro+ Laptop,13.3" 3200x1800 Res...	4.6 out of 5 stars	279	4
6	2020 HP 15.6" Touchscreen Laptop Computer/ 10t...	4.5 out of 5 stars	599	152|More Buying Choices
7	2021 Newest Dell Inspiron 15 3000 Laptop, 15.6...	4.1 out of 5 stars	449	8
8	Dell Inspiron 17 3793 2020 Premium 17.3" FHD L...	4.4 out of 5 stars	800	410|More Buying Choices
9	Dell Inspiron 15 5502, 15.6 inch FHD Non-Touch...	4.5 out of 5 stars	695	39|More Buying Choices
10	Acer Predator Helios 300 Gaming Laptop, intel ...	4.7 out of 5 stars	1,709	2855|More Buying Choices

Figure 12. The extracted information about laptops from multiple web pages on Amazon

For the sake of extracting custom product features from online customer reviews and training the NER model, another dataset needed to be created. Once more, since no benchmark dataset was available regarding laptop features, a collection of reviews for various types of laptops, are extracted from the Amazon website to train and validate the model in phase 2.

#	Feature	Description	Data Type
1	brand	Laptops brand names e.g., Apple, Asus, …	String
2	price	The price of each product e.g., $750	Integer
3	size	The screen size of the laptop e.g., 11”	Integer
4	cpu	The CPU model e.g., Intel core i7	String
5	hard	The Hard Disc capacity e.g., 128 GB	Integer
6	ram	RAM (Random Access Memory) capacity e.g., 8 GB	Integer
7	number_of_rating	Number of customers who rated the product.	Integer
8	rating_score	The average rating of the laptop received in total.	Float

Table 2. Phase 1 model dataset description

The following is an instance of a review posted online regarding the Surface Pro 7 laptop: *“Pros:• USB-C, Finally! As a past Surface Pro user, this was never a feature. With a USB-C, like any computer, you can do so much more with this device now. From headphones to connecting external 4k monitors, to charging this and other devices (USB-C chargers are 10x cheaper to get), to even connecting an Ethernet adaptor. About time Microsoft. Cons and Oks:• The chunky bezel. • The battery life isn’t great. I’m getting around 8 hours of power. That’s having Wi-Fi always on, web surfing, daily YouTube playing, and a couple of hours with Netflix (plus a couple of smaller apps). Overall: An outdated design that packs a lot of power for those who want more than just a generic tablet on the go. Hope I was a help to you. Love, Honest Reviewer”*

Following the CoNLL2003 dataset instruction on how to prepare the data to train the NER model, a new dataset consisting of 609 reviews, 1,190 sentences, and 21,365 tokens was created. Based on CoNLL2003, no pre-processing is needed. Therefore, the reviews should not be cleaned, and no numbers or stop words such as “a”, “the”, “is”, or “are” have to be removed. In this way, the model can learn all the possible states of input and label any unseen word efficiently. The reviews are then tokenized using Natural Language Toolkit (NLTK) [60]. In order to label the custom product features the BILOU annotation technique is applied. BILOU indicates boundary tokens by the Outside of text segments, the Beginning, Inside, and Last tokens of multi-token chunks, and Unit-length chunks. For instance, "connecting 4K monitor" is labeled as Connecting: "O", 4K: "B-Feature", and Monitor: "L-Feature". The final look of the dataset for the first 15 rows is depicted in Figure 13. Afterward, the most frequently labeled features, predicted by the NER classifier on all gathered reviews will be fed to the model developed in phase 3.

Reviews ID	Sentence ID	Tokens	Label
0	0	pros	O
0	0	•	O
0	0	usb-c	U-F
0	0	,	O
0	0	finally	O
0	0	!	O
0	1	as	O
0	1	a	O
0	1	past	O
0	1	surface	O
0	1	pro	O
0	1	user	O
0	1	,	O
0	1	this	O
0	1	was	O

Figure 13. NER training dataset

6. Computational Experiments

This section outlines a number of experiments that were carried out to assess the performance of the proposed models. First, the baseline methods used in each phase are presented. Then, the experiment setups are described for the proposed approach.

6.1. Baseline

Three baseline methods are employed to run phase 1 and phase 3 models including DNN, Probabilistic Neural Network (PNN) [61], and Radial Basis Function Neural Network (RBFNN) [55]. The DNN and RBFNN methods are implemented in TensorFlow Keras (https://www.tensorflow.org/api_docs/python/tf/keras), and the PNN method is executed in NeuPy (http://neupy.com) framework. Keras's sequential model trains the network using the backpropagation algorithm and is optimized according to the optimization algorithm and loss function specified when compiling the model. For the sake of extracting product features from customer online reviews in phase 2, Simple Transformers (https://simpletransformers.ai), which is the simplest way to implement BERT, is applied.

6.2. Experiment Setups

The input fed to the phase 1 model consists of 2028 examples and 7 features described in Table 2. All methods were run on a GPU in Google Colab (https://colab.research.google.com) platform. The PNN Network is sensitive to cases when one input feature has higher values than the other one. Therefore, the input data is normalized before training, by applying StandardScaler(), which is a function of the Python Scikit-Learn library (https://scikit-learn.org/). Another important parameter for the PNN model is the standard deviation. The standard deviation (std) must be within the range of the input features. Furthermore, RBFNNs are fully connected feedforward neural networks that use a radial basis function as the activation function of the hidden layers. The function, which is usually the Gaussian is defined as a function of distance from a certain central point. For the RBFNN baseline method, node locations corresponding to K-means cluster centers were used to initialize the network.

In phase 2, the model used from the Simple Transformers library is "*Named Entity Recognition*" with the model type "*bert*" and the model's name "*bert-base-cased*" [62]. Data prepared is consisting of 21,365 examples and 3 features (Sentence ID, Tokens, and Labels) which are divided into train and validation sets by a 0.2 ratio.

To ensures that the score of our model does not depend on the way we select our train and test subsets cross-validation technique is implemented. Cross-validation is a resampling technique used to assess machine learning models on a small sample of data. The process contains a single parameter called k that specifies the number of groups into which a given data sample should be divided. As a result, the process is frequently referred to as k-fold cross-validation. When a specific number for k is selected, for example, k=10 resulting in 10-fold cross-validation. Cross-validation uses a small sample size to assess how the model is likely to perform in general when used to generate predictions on data that was not included during the model's training [63]. In this research, 10-fold cross-validation was performed 200 times for each experiment. Moreover, the average accuracy was calculated based on 200 cross-validation replications. Finally, a grid search was used to find the optimum hyper-parameters.

An imbalance in the dataset was the main concern, as the number of products rated from 3 to 4 was not equal to the number of products rated from 4 to 5. To tackle this issue, a combination of oversampling and under sampling techniques, named SMOTE[5]Tomek [64], which is proved to be more effective than just using solely oversampling (SMOTE) or under-sampling (Tomek) [65], was implemented. The details of the hyper-parameters and configurations utilized in the above-mentioned algorithms are presented in Table 3 and Table 4.

Input shape (Phase 1)	2028 × 7
Input shape (Phase 3)	2028 × 13
Dropout Rate	0.3
Learning Rate	0.01
Hidden Layers Activation Function	ReLU [48]
Optimizer	Adam [51]

[5] SMOTE: Synthetic Minority Oversampling Technique

Batch Size	128
epochs	200
PNN std	0.3
RBFNN betas	2.0

Table 3. Phase 1 & Phase 3 hyper-parameters and configurations

Input shape (Phase 2)	21365 × 3
Model type	bert
Model's name	bert-base-cased[62]
num_train_epochs	12
Learning Rate	1e-4
train_batch_size	64
eval_batch_size	32

Table 4. Simple Transformers hyper-parameters and configurations

6.3. Experimental Results

This research aims to thoroughly analyze the performance of the proposed models in the three mentioned stages. The results of each phase are displayed in Table 5 to Table 8, respectively. These results were achieved on the laptop dataset collected from Amazon.

6.3.1. Results for Phase 1 Model

This stage corresponds to a product rating prediction model, using general features extracted from online product information, including Brand Names, Price, Screen Size, CPU, Hard Disk Size, RAM, and Number of Ratings. The three baseline Neural Networks algorithms are used to predict the overall product rating score. Table 5 displays the results for the first phase on the laptop dataset. As it can be observed from the table, the RBFNN had the best performance in terms of accuracy, recall, and F1 score, using the first seven general product features as the input. On the other hand, the DNN model outperformed the PNN and RBFNN in terms of Precision and Specificity which shows the model's ability to truly predict the positive and negative cases.

Model	Input Features	Accuracy	Precision	Recall (Sensitivity)	Specificity	F1 Score
DNN	$x_0 - x_6$ (7 features)	78.92% (+/- 1.68%)	**82.90% (+/- 3.54%)**	72.82% (+/- 4.67%)	**84.98% (+/- 3.22%)**	77.40% (+/- 2.88%)
PNN		79.56% (+/- 1.56%)	81.19% (+/- 2.33%)	76.87% (+/- 2.39%)	82.22% (+/- 1.77%)	79.56% (+/- 1.56%)
RBFNN		**80.31% (+/- 2.74%)**	80.93% (+/- 2.02%)	**79.16% (+/- 4.62%)**	81.25% (+/- 2.97%)	**79.99% (+/- 3.13%)**

DNN: Deep Neural Network, **PNN**: Probability Neural Network, **RBFNN**: Radial Basis Function Neural Network

Table 5. Comparison results for phase 1 model on the laptop dataset with general features

6.3.2. Results for Phase 2 Model

In the second stage, additional features are extracted from online customer reviews to identify the most frequent custom product features that have been mentioned by the consumers. In order to determine these custom features, a Simple Transformers NER model is applied to a new dataset. The details of this newly created dataset were explained in section 0. The model could achieve a good performance in classifying the product features. An instance of a predicted review is provided below. As it can be seen, the model could correctly predict "i7", "16gb ram", "i9", and "battery life" which are referred to as CPU model, RAM size, and battery life, as product features. The results of this part are summarized in Table 6.

Model	Model Type	Eval_loss	Precision	Recall	F1 Score
Simple Transformers NERModel	bert	0.1741	79.34%	80%	79.67%

Table 6. Results for Simple Transformers NERModel to extract additional product features from online customer reviews

Review:

"the i7 version with 16gb ram is ample and it is highly responsive and fast - even in comparison to my heavy duty i9 laptop. Horrible, unacceptable battery life!"

Predicted labels:

```
[[{'the': 'O'},
  {'i7': 'U-F'},
  {'version': 'O'},
  {'with': 'O'},
  {'16gb': 'B-F'},
  {'ram': 'L-F'},
  {'is': 'O'},
```

```
{'ample': 'O'},
{'and': 'O'},
{'it': 'O'},
{'is': 'O'},
{'highly': 'O'},
{'responsive': 'O'},
{'and': 'O'},
{'fast': 'O'},
{'-': 'O'},
{'even': 'O'},
{'in': 'O'},
{'comparison': 'O'},
{'to': 'O'},
{'my': 'O'},
{'heavy': 'O'},
{'duty': 'O'},
{'i9': 'U-F'},
{'laptop.': 'O'},
{'horrible,': 'O'},
{'unacceptable': 'O'},
{'battery': 'B-F'},
{'life!': 'L-F'}]]
```

By analyzing all the gathered reviews, the most frequent labeled features predicted by the NER classifier were obtained. Based on this, 7 extra features are added to the first set of general features that include Battery Life, Screen Resolution, OS, Number of Ports, Weight, USB-C, and Camera Quality. The details of extracted features are provided in Table 7.

#	Feature	Description	Data Type
1	battery_life	The duration that the battery can hold charge in hour	Float
2	resolution	The screen resolution e.g., 1080, 768, …	Interger
3	os	The Operation System e.g., Windows, Mac, …	String
4	number_of_ports	The number of input ports e.g., usb port	Integer
5	weight	The weight of the laptop in lbs	Float
6	usb-c	Whether the laptop has usb-c or not	Boolean
7	camera	The quality of the built-in webcam e.g., VGA, HD, …	String

Table 7. Phase 2 model dataset description

6.3.3. Results for Phase 3 Model

In the third stage, the combination of general features and additional custom product features, a total of 14 features, are fed to the Neural Networks algorithms. Once again, the three baseline models with the same hyper-parameters and configurations are used to predict the overall product rating score. In the phase 3 model, in order to achieve the optimal results and due to the excessive number of features, a feature selection [62] method is implemented. Feature selection is an excellent solution to reduce the dimensionality of feature vector space as it removes irrelevant and redundant data, reduces computation time, improves learning accuracy, and facilitates better comprehension of the learning model or data [62]. To apply this method, sklearn.feature_selection module of python [63] has been utilized. This module works by selecting the best features based on univariate statistical tests [63]. In this model, the chi2 ($\chi 2$) test is performed to select the highest scoring features. As a result, 13 features out of 14 (all features except number_of_ports) yield the best performance in terms of evaluation metrics.

Table 8 displays the results for the third phase model executed on the phase 1 dataset plus custom features. As can be observed from the table, RBFNN outperforms the other neural network models in terms of accuracy, recall, and F1-score, while PNN achieved better results in terms of precision, and specificity which shows the model's ability to truly predict the positive and negative cases.

Model	Input Features	Accuracy	Precision	Recall (Sensitivity)	Specificity	F1 Score
DNN	$x_0 - x_{13}$ (13 features)	83.31% (+/- 1.48%)	86.76% (+/- 2.88%)	78.87% (+/- 3.55%)	87.93% (+/- 2.82%)	82.52% (+/- 1.16%)
PNN		83.66% (+/- 1.84%)	**87.63% (+/- 3.18%)**	78.29% (+/- 3.20%)	**89.08% (+/- 2.20%)**	83.66% (+/- 1.84%)
RBFNN		**84.01% (+/- 1.80%)**	83.75% (+/- 2.86%)	**84.68% (+/- 5.04%)**	83.45% (+/- 3.56%)	**84.06% (+/- 1.90%)**

DNN: Deep Neural Network, **PNN**: Probability Neural Network, **RBFNN**: Radial Basis Function Neural Network

Table 8. Comparison results for phase 3 model on the laptop dataset with combined features (13 features selected)

7. Discussion

By comparing the results in Table 5 and Table 8, it can be concluded that adding the extra features to the proposed product rating prediction model obtained better results in all three neural networks and all the performance metrics. The benchmark models are improved by 3.7%-4.4% in accuracy, 2.8%-6.4% in precision, 1.4%-6% in recall, 2.2%-6.9% in specificity, and 4.1-5.1% in F1 score. Moreover, the corresponding training time for all the models on Google Colab with GPU is also presented in Table 9, where running time may vary significantly due to computation resource fluctuation. In terms of processing time, PNN is the fastest algorithm to predict the product's overall rating scores.

Final Model	Input Features	Training Time
DNN	$x_7 - x_{13}$ (13 features)	00:03:23
PNN		00:00:02
RBFNN		00:03:55

Table 9. Phase 3 models Training Time on Google Colab with 1 GPU (run-time format in hour:minute:second)

Furthermore, a general comparison was performed between this work and some recent research in the area of rating prediction on different datasets. This comparison was made based on the fact that the overall rating score of a product is the average of every single rating coming along with the product review. In [1], the authors used DL algorithms to predict the review rating. The framework consists of two phases comprising DL bidirectional gated recurrent unit (Bi-GRU) model architecture and word embedding as the input features. The first phase is used for polarity prediction (negative, neutral, or positive sentiment), and the second phase is used to predict review ratings from the review text on Amazon books and Yelp restaurants. In both datasets, the phase 3 final model of this study outperformed their scores in terms of precision, recall, and F1-measure.

In [16], the authors performed a review rating prediction as a multiclass classification problem. They used the latent topic of the review, by applying topic modeling techniques, and the sentiment of the review as the input features fed to their DL models. The task of determining which topics best characterizes a given corpus is known as topic modeling and the latent topic is the hidden topic that can be perceived from the document. Furthermore, the sentiment of each review is extracted using a sentiment analyzer developed by the authors. An ML/DL-based prediction model is then trained using the original user ratings, latent topics, and their sentiments. Despite this study, that predicts the overall product rating, their model predicts the ratings for the corresponding reviews. To evaluate the performance of their proposed model, the authors utilized a similar category of products as this study (i.e., cellphones and electronics). Given that they just predicted the rating of each review rather than the overall product rating, they could achieve higher results than this work. However, without using sentiment analysis, which is not employed here, their findings were lower than ours.

Lastly, Liu [66] focuses on rating prediction for restaurants based on their review texts without applying sentiment analysis. The model is a multiclass classification problem, where the input is the textual data (reviews), and the output is the predicted class (1-5 stars). The prediction task is done using several ML and transformer-based algorithms such as BERT, described in section 4.2.2. They could achieve the best results, through the XLNet classifier, a large bidirectional transformer that uses enhanced training methodology, larger datasets, and more computational power. As a result, the XLNet classifier on the Yelp restaurants dataset led to a score of 70% in both accuracy and F1 measure. Table 10 provides the comparison scheme for the above-mentioned studies.

Study	Features	Classifier	Dataset	Accuracy	Precision	Recall	F1
[1]	Word Embedding	Bi-GRU	Amazon (Books)	-	72%	72%	69%
			Yelp (Restaurants)	-	67%	66%	66%
[16]	Latent topic of reviews	RNN	Amazon (Electronics)	82%	74%	80%	81%
			Amazon (Cell Phones and Accessories)	78%	72%	73%	74%
[66]	Textual data from reviews	XLNet	Yelp (Restaurants)	70.44%	-	-	70.87%

Study	Features	Classifier	Dataset	Accuracy	Precision	Recall	F1
Proposed Model	Product Features	DNN	Amazon (Laptops)	83.31%	86.76%	78.87%	82.52%
		PNN		83.66%	**87.63%**	78.29%	83.66%
		RBF NN		**84.01%**	83.75%	**84.68%**	**84.06%**

Bi-GRU: Bidirectional Gated Recurrent Unit, **RNN**: Recurrent Neural Network, **DNN**: Deep Neural Network, **PNN**: Probability Neural Network, **RBFNN**: Radial Basis Function Neural Network

Table 10. Comparing the results of the proposed model with recent related works on other validated datasets

The results justify the research question on whether it is possible to predict the popularity of a product via its rating score using solely the product features. This work demonstrates that product features can be used to predict the overall product rating score while most of the existing related literature, predicts the review rating score by identifying the sentiment polarity of the reviews. Another significant achievement of the proposed model is that not all the general features are the main reason behind shaping customer satisfaction, but also there are some features that actually matter to customers which can be hidden in thousands of reviews they share.

8. Conclusion

This research investigated the influence of general and custom product features on predicting product overall rating scores. The designed system consists of three different phases: (1) predicting overall rating by feeding the general product features, extracted from the online product information available on Amazon webpages to the DL training model. (2) identifying other features that customers care about the most by applying the NER algorithm to the customer online reviews, and (3) feeding the combination of the general and custom features to the DL training model to predict the overall rating score of the product. The datasets used in this research are new sources created by the author. Data has been collected on laptop products from the Amazon website and the annotated dataset of product features to train the NER model has been scraped and prepared using online customer reviews. The experimental results demonstrated an impressive performance of the proposed model in predicting the overall rating score of the product. Also, the proposed model's prediction performance is improved when custom features are added to the input. The RBFNN model could achieve the highest accuracy of 84.01%, 84.68% for recall, and 84.06% for F1 score, while the PNN model obtained the highest score of 87.63% in precision. In the end, the proposed model could outperform similar works in predicting rating scores without applying sentiment analysis.

This research benefits businesses to accurately identify the exact points of strengths and weaknesses of their products or services from the customer's point of view. The investigation of customers' experiences is intrinsic to the business's success as it can give inspiration and insight to companies to understand customers' needs and desires.

9. Future research directions

This study can be extended in several directions elaborated as follow:

1) Due to the lack of a benchmark dataset, this research is based on original data collections, as mentioned in earlier sections. Since having more data in deep learning models results in better and more consistent outcomes, future research can enhance the current datasets by adding more samples.
2) The proposed framework in this study can be further validated in the context of more general categories of products and online reviews such as electronics, movies, books, etc.
3) To resolve the imbalance in data, a combination of oversampling and under sampling techniques has been used in this research. In the future, all the possible sampling schemes such as Adaptive Synthetic Sampling or a combination of SMOTE (oversampling) with other under sampling techniques may be explored and an appropriate one may be used in the proposed framework to improve the prediction performance.
4) When analyzing reviews, it is worth assuming not all of them are meaningful as some of them may be rather fraudulent. To address this issue, spam review detection algorithms can be employed to remove fake and redundant reviews that are less important and cause the model's performance to drop.
5) Covering more e-commerce websites other than Amazon such as eBay, Walmart, BestBuy, Flipkart, etc. could be another path to follow in order to enhance the current datasets.
6) Another research direction would be identifying the product features that altering them would enhance the product rating and determining how these modifications should be made in order to get the optimum results.

Appendix A

This study is done on original datasets developed by the author. For each phase of the research, a unique dataset has been created. A screenshot of each of them is presented in Table A. 1, Table A. 2, and Table A. 3

Table A. 1: Dataset used in phase 1 model

	A	B	C	D	E	F	G	H
1	**Brand**	**Price**	**Size**	**CPU**	**Hard**	**RAM**	**#ratings**	**Rating Score**
2	asus	487	17.3	ryzen3	256	8	8	4.4
3	acer	593	15.6	ryzen5	256	8	1012	4.7
4	hp	399	14	athlon	256	8	67	4.6
5	acer	513	15.6	corei5	256	8	46	4.3
6	hp	869	15.6	corei5	512	16	12	4.8
7	chuwi	279	13.3	celeron	128	8	4	4.6
8	hp	599	15.6	corei5	256	12	152	4.5
9	dell	449	15.6	celeron	128	8	8	4.1
10	dell	800	17.3	corei5	1512	16	410	4.4
11	dell	695	15.6	corei5	512	8	39	4.5
12	acer	1709	17.3	corei7	1000	16	2855	4.7
13	hp	649	15.6	corei3	512	16	4	3.1
14	dell	589	15.6	celeron	1256	12	183	4.5
15	lenovo	269	14	amda6	64	4	1286	3.9
16	hp	359	11.6	celeron	32	4	112	4.3
17	asus	285	14	celeron	128	4	22	4.5
18	dell	499	15.6	celeron	1000	16	30	4.3
19	lenovo	379	14	pentium	128	8	177	4.5
20	asus	549	15.6	ryzen7	512	8	640	4.6

Table A. 2: Dataset used in phase 2 model

	A	B	C	D
1	**reviews ID**	**sentence_id**	**Tokens**	**Tag**
2	0	0	pros	O
3	0	0	•	O
4	0	0	usb-c	U-F
5	0	0	,	O
6	0	0	finally	O
7	0	0	!	O
8	0	1	as	O
9	0	1	a	O
10	0	1	past	O
11	0	1	surface	O
12	0	1	pro	O
13	0	1	user	O
14	0	1	,	O
15	0	1	this	O
16	0	1	was	O
17	0	1	never	O
18	0	1	a	O
19	0	1	feature	O
20	0	1	.	O

Table A. 3: Dataset used in phase 3 model

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	**Brand**	**Price**	**Size**	**CPU**	**Hard**	**RAM**	**#ratings**	**Rating Score**	**Battery Life**	**Resolution**	**OS**	**#ports**	**Weight**	**USB-C**	**Camera**
2	asus	487	17.3	ryzen3	256	8	8	4.4	5	1080	windows	6	5.07	1	HD
3	acer	593	15.6	ryzen5	256	8	1012	4.7	7.5	1080	windows	6	3.97	0	HD
4	hp	399	14	athlon	256	8	67	4.6	4	768	windows	6	3.24	1	HD
5	acer	513	15.6	corei5	256	8	46	4.3	8	1080	windows	6	4.2	0	VGA
6	hp	869	15.6	corei5	512	16	12	4.8	9	1080	windows	6	3.75	1	HD
7	chuwi	279	13.3	celeron	128	8	4	4.6	7	3200	windows	5	2.55	0	VGA
8	hp	599	15.6	corei5	256	12	152	4.5	7	768	windows	6	3.75	1	HD
9	dell	449	15.6	celeron	128	8	8	4.1	8	768	windows	7	3.91	0	HD
10	dell	800	17.3	corei5	1512	16	410	4.4	4	1080	windows	8	6.16	1	HD
11	dell	695	15.6	corei5	512	8	39	4.5	7	1080	windows	6	3.78	1	HD
12	acer	1709	17.3	corei7	1000	16	2855	4.7	5.5	1080	windows	8	6.39	1	HD
13	hp	649	15.6	corei3	512	16	4	3.1	7	768	windows	7	3.86	1	HD
14	dell	589	15.6	celeron	1256	12	183	4.5	10	768	windows	6	7.09	0	HD
15	lenovo	269	14	amda6	64	4	1286	3.9	8	768	windows	5	3.09	0	VGA
16	hp	359	11.6	celeron	32	4	112	4.3	12	768	windows	6	2.37	1	HD
17	asus	285	14	celeron	128	4	22	4.5	12	768	windows	6	2.87	0	VGA
18	dell	499	15.6	celeron	1000	16	30	4.3	10	768	windows	6	3.91	0	HD
19	lenovo	379	14	pentium	128	8	177	4.5	6	768	windows	6	3.31	0	VGA
20	asus	549	15.6	ryzen7	512	8	640	4.6	9	1080	windows	7	3.5	1	HD

References

[1] B. H. Ahmed and A. S. Ghabayen, “Review rating prediction framework using deep learning,” *Journal of Ambient Intelligence and Humanized Computing*, 2020, doi: 10.1007/s12652-020-01807-4.

[2] S. S. Sohail, J. Siddiqui, and R. Ali, “Feature extraction and analysis of online reviews for the recommendation of books using opinion mining technique,” *Perspectives in Science*, vol. 8, 2016, doi: 10.1016/j.pisc.2016.06.079.

[3] S. E. Schaeffer and S. V. Rodriguez Sanchez, “Forecasting client retention — A machine-learning approach,” *Journal of Retailing and Consumer Services*, vol. 52, 2020, doi: 10.1016/j.jretconser.2019.101918.

[4] A. Géron, *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems*. 2017.

[5] J. Salminen, V. Yoganathan, J. Corporan, B. J. Jansen, and S. G. Jung, “Machine learning approach to auto-tagging online content for content marketing efficiency: A comparative analysis between methods and content type,” *Journal of Business Research*, vol. 101, 2019, doi: 10.1016/j.jbusres.2019.04.018.

[6] I. el Naqa and M. J. Murphy, “What Is Machine Learning?,” in *Machine Learning in Radiation Oncology*, 2015. doi: 10.1007/978-3-319-18305-3_1.

[7] R. C. Mayo and J. Leung, “Artificial intelligence and deep learning – Radiology’s next frontier?,” *Clinical Imaging*, vol. 49. 2018. doi: 10.1016/j.clinimag.2017.11.007.

[8] X. Huang, C. Zanni-Merk, and B. Crémilleux, “Enhancing deep learning with semantics: An application to manufacturing time series analysis,” in *Procedia Computer Science*, 2019, vol. 159. doi: 10.1016/j.procs.2019.09.198.

[9] A. Kim, Y. Yang, S. Lessmann, T. Ma, M. C. Sung, and J. E. V. Johnson, "Can deep learning predict risky retail investors? A case study in financial risk behavior forecasting," *European Journal of Operational Research*, vol. 283, no. 1, 2020, doi: 10.1016/j.ejor.2019.11.007.

[10] M. Kraus, S. Feuerriegel, and A. Oztekin, "Deep learning in business analytics and operations research: Models, applications and managerial implications," *European Journal of Operational Research*, vol. 281, no. 3, 2020, doi: 10.1016/j.ejor.2019.09.018.

[11] Y. Wang *et al.*, "A CLSTM-TMN for marketing intention detection," *Engineering Applications of Artificial Intelligence*, vol. 91, 2020, doi: 10.1016/j.engappai.2020.103595.

[12] W. M. Wang, Z. Li, Z. G. Tian, J. W. Wang, and M. N. Cheng, "Extracting and summarizing affective features and responses from online product descriptions and reviews: A Kansei text mining approach," *Engineering Applications of Artificial Intelligence*, vol. 73, pp. 149–162, Aug. 2018, doi: 10.1016/j.engappai.2018.05.005.

[13] K. R. Chowdhary, *Fundamentals of artificial intelligence*. Springer India, 2020. doi: 10.1007/978-81-322-3972-7.

[14] Biemann, *Text Mining From Ontology Learning to Automated Text Processing Applications*. 2006.

[15] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Engineering Journal*, vol. 5, no. 4, 2014, doi: 10.1016/j.asej.2014.04.011.

[16] A. Mahadevan and M. Arock, "Review rating prediction using combined latent topics and associated sentiments: an empirical review," *Service Oriented Computing and Applications*, vol. 14, no. 1, 2020, doi: 10.1007/s11761-019-00278-6.

[17] J. Chambua and Z. Niu, "Review text based rating prediction approaches: preference knowledge learning, representation and utilization," *Artificial Intelligence Review*, vol. 54, no. 2, 2021, doi: 10.1007/s10462-020-09873-y.

[18] R. Cao, X. Zhang, and H. Wang, "A Review Semantics Based Model for Rating Prediction," *IEEE Access*, vol. 8, 2020, doi: 10.1109/ACCESS.2019.2962075.

[19] B. Wang, S. Xiong, Y. Huang, and X. Li, "Review rating prediction based on user context and product context," *Applied Sciences (Switzerland)*, vol. 8, no. 10, 2018, doi: 10.3390/app8101849.

[20] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, 1997, doi: 10.1109/78.650093.

[21] W. Wang and J. Gang, "Application of Convolutional Neural Network in Natural Language Processing," 2019. doi: 10.1109/ICISCAE.2018.8666928.

[22] S. Hasanzadeh, S. M. Fakhrahmad, and M. Taheri, "Review-Based Recommender Systems: A Proposed Rating Prediction Scheme Using Word Embedding Representation of Reviews," *The Computer Journal*, vol. 65, no. 2, 2022, doi: 10.1093/comjnl/bxaa044.

[23] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[24] B. Wang, B. Chen, L. Ma, and G. Zhou, "User-personalized review rating prediction method based on review text content and user-item rating matrix," *Information (Switzerland)*, vol. 10, no. 1, 2018, doi: 10.3390/info10010001.

[25] N. Shrestha and F. Nasoz, "Deep Learning Sentiment Analysis of Amazon.Com Reviews and Ratings," *International Journal on Soft Computing, Artificial Intelligence and Applications*, vol. 8, no. 1, 2019, doi: 10.5121/ijscai.2019.8101.

[26] Y. Kang and L. Zhou, “Extracting product features from online consumer reviews,” in *19th Americas Conference on Information Systems, AMCIS 2013 - Hyperconnected World: Anything, Anywhere, Anytime*, 2013, vol. 3.

[27] G. Qiu, B. Liu, J. Bu, and C. Chen, “Opinion word expansion and target extraction through double propagation,” *Computational Linguistics*, vol. 37, no. 1, 2011, doi: 10.1162/coli_a_00034.

[28] D. Mollá, M. van Zaanen, and D. Smith, “Named Entity Recognition for Question Answering,” 2006.

[29] J. Guo, G. Xu, X. Cheng, and H. Li, “Named entity recognition in query,” 2009. doi: 10.1145/1571941.1571989.

[30] B. Babych and A. Hartley, “Improving machine translation quality with automatic named entity recognition,” 2003. doi: 10.3115/1609822.1609823.

[31] V. Yadav and S. Bethard, “A survey on recent advances in named entity recognition from deep learning models,” 2018.

[32] H. Shelar, G. Kaur, N. Heda, and P. Agrawal, “Named Entity Recognition Approaches and Their Comparison for Custom NER Model,” *Science and Technology Libraries*, vol. 39, no. 3, 2020, doi: 10.1080/0194262X.2020.1759479.

[33] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual prediction with LSTM,” *Neural Computation*, vol. 12, no. 10, 2000, doi: 10.1162/089976600300015015.

[34] N. Kanya and T. Ravi, “Modelings and techniques in named entity recognition-an information extraction task,” in *IET Conference Publications*, 2012, vol. 2012, no. 624 CP. doi: 10.1049/cp.2012.2199.

[35] H. Xia, Y. Yang, X. Pan, Z. Zhang, and W. An, “Sentiment analysis for online reviews using conditional random fields and support vector machines,” *Electronic Commerce Research*, vol. 20, no. 2, pp. 343–360, Jun. 2020, doi: 10.1007/s10660-019-09354-7.

[36] M. Liu, Z. Tu, Z. Wang, and X. Xu, “LTP: A New Active Learning Strategy for Bert-CRF Based Named Entity Recognition,” 2020. Accessed: Mar. 07, 2022. [Online]. Available: https://arxiv.org/abs/2001.02524v1

[37] “Amazon.com : laptop.” https://www.amazon.com/s?k=laptop&crid=34N6RB1C7AV85 (accessed Mar. 07, 2022).

[38] “Scrape data from any website with 1 Click | Data Miner.” https://dataminer.io/ (accessed Mar. 07, 2022).

[39] “re — Regular expression operations — Python 3.10.2 documentation.” https://docs.python.org/3/library/re.html (accessed Mar. 07, 2022).

[40] S. Wang and C. D. Manning, “Baselines and bigrams: Simple, good sentiment and topic classification,” in *50th Annual Meeting of the Association for Computational Linguistics, ACL 2012 - Proceedings of the Conference*, 2012, vol. 2.

[41] “Simple Transformers.” https://simpletransformers.ai/ (accessed Mar. 07, 2022).

[42] “BERT: Why it’s been revolutionizing NLP | by Jerry Wei | Towards Data Science.” https://towardsdatascience.com/bert-why-its-been-revolutionizing-nlp-5d1bcae76a13 (accessed Mar. 07, 2022).

[43] E. F. Tjong Kim Sang and F. de Meulder, “Introduction to the CoNLL-2003 shared task,” 2003. doi: 10.3115/1119176.1119195.

[44] L. Ratinov and D. Roth, “Design challenges and misconceptions in named entity recognition,” 2009. doi: 10.3115/1596374.1596399.

[45] A. Krogh, “What are artificial neural networks?,” *Nature Biotechnology*, vol. 26, no. 2. 2008. doi: 10.1038/nbt1386.

[46] S. Ding, H. Li, C. Su, J. Yu, and F. Jin, "Evolutionary artificial neural networks: A review," *Artificial Intelligence Review*, vol. 39, no. 3. 2013. doi: 10.1007/s10462-011-9270-6.

[47] T. K. Gupta and K. Raza, "Optimizing Deep Feedforward Neural Network Architecture: A Tabu Search Based Approach," *Neural Processing Letters*, vol. 51, no. 3, 2020, doi: 10.1007/s11063-020-10234-7.

[48] K. Eckle and J. Schmidt-Hieber, "A comparison of deep networks with ReLU activation function and linear spline-type methods," *Neural Networks*, vol. 110, 2019, doi: 10.1016/j.neunet.2018.11.005.

[49] S. A. Abdullah and A. Al-Ashoor, "An artificial deep neural network for the binary classification of network traffic," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 1, 2020, doi: 10.14569/ijacsa.2020.0110150.

[50] "Cross-Entropy Loss Function. A loss function used in most… | by Kiprono Elijah Koech | Towards Data Science." https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e (accessed Mar. 22, 2022).

[51] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," 2015.

[52] D. F. Specht, "Probabilistic Neural Networks," 1990.

[53] B. Mohebali, A. Tahmassebi, A. Meyer-Baese, and A. H. Gandomi, "Probabilistic neural networks: A brief overview of theory, implementation, and application," in *Handbook of Probabilistic Models*, 2019. doi: 10.1016/B978-0-12-816514-0.00014-X.

[54] W. Zhang, X. Yang, Y. Deng, and A. Li, "An Inspired Machine-Learning Algorithm with a Hybrid Whale Optimization for Power Transformer PHM", doi: 10.3390/en13123143.

[55] C. C. Lee, P. C. Chung, J. R. Tsai, and C. I. Chang, "Robust radial basis function neural networks," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 29, no. 6, 1999, doi: 10.1109/3477.809023.

[56] M. Martynova, "A novel approach of the approximation by patterns using hybrid RBF NN with flexible parameters," in *Advances in Intelligent Systems and Computing*, 2019, vol. 859. doi: 10.1007/978-3-030-00211-4_21.

[57] L. E. Aik, T. W. Hong, and A. K. Junoh, "Distance weighted K-means algorithm for center selection in training radial basis function networks," *IAES International Journal of Artificial Intelligence*, vol. 8, no. 1, 2019, doi: 10.11591/ijai.v8.i1.pp54-62.

[58] W. B. Zhao, D. S. Huang, and L. Guo, "Comparative study between radial basis probabilistic neural networks and radial basis function neural networks'," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 2690, 2004, doi: 10.1007/978-3-540-45080-1_50.

[59] H. Lassoued, R. Ketata, and S. Yacoub, "ECG decision support system based on feedforward neural networks," *International Journal on Smart Sensing and Intelligent Systems*, vol. 11, no. 1, 2018, doi: 10.21307/IJSSIS-2018-029.

[60] S. Bird and E. Loper, "The natural language toolkit NLTK: The Natural Language Toolkit," *Proceedings of the ACL-02 Workshop on Effective tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, no. March, 2016.

[61] D. F. Specht, "Probabilistic neural networks," *Neural Networks*, vol. 3, no. 1, 1990, doi: 10.1016/0893-6080(90)90049-Q.

[62] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2019, vol. 1.

[63] D. Berrar, "Cross-validation," in *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics*, vol. 1–3, 2018. doi: 10.1016/B978-0-12-809633-8.20349-X.

[64] G. E. A. P. A. Batista, A. L. C. Bazzan, and M. C. Monard, "Balancing Training Data for Automated Annotation of Keywords: a Case Study," *In Proceedings of the Second Brazilian Workshop on Bioinformatics*, no. January, 2003.

[65] "Stop using SMOTE to handle all your Imbalanced Data | by Satyam Kumar | Towards Data Science." https://towardsdatascience.com/stop-using-smote-to-handle-all-your-imbalanced-data-34403399d3be (accessed Mar. 07, 2022).

[66] Z. Liu, "Yelp Review Rating Prediction: Machine Learning and Deep Learning Models," Dec. 2020, [Online]. Available: http://arxiv.org/abs/2012.06690

[67] "CatBoost - open-source gradient boosting library." https://catboost.ai/ (accessed Mar. 07, 2022).

[68] "XGBoost Documentation — xgboost 1.6.0-dev documentation." https://xgboost.readthedocs.io/en/latest/ (accessed Mar. 07, 2022).

[69] "Welcome to LightGBM's documentation! — LightGBM 3.3.2.99 documentation." https://lightgbm.readthedocs.io/en/latest/ (accessed Mar. 07, 2022).