Experimental Study to Identify the Emotion from Recipes Reviews
Second Project for NLP Course, Winter 2022

Amir Ali
Warsaw University of Technology
amir.ali.stud@pw.edu.pl

Stanisław Matuszewski
Warsaw University of Technology
stanislaus.matuszewski@gmail.com

Jacek Czupyt
Warsaw University of Technology
jacek.czupyt@gmail.com

Supervisor: Anna Wróblewska
Warsaw University of Technology
anna.wroblewska1@pw.edu.pl

Abstract

In this study, we aimed to identify emotions from food recipes using two different datasets; our own food review data and the Amazon food recipes data. We experimented with different feature extraction techniques such as bag-of-words (BOW), TF-IDF, word2vec, and One_Hot pad sequence. We then trained different models on these features, including Bert, fastText, HAN, RNN, BIRNN, AttBiRNN, CNN, and LSTM. Our results showed that the BERT, BIRNN models performed the best, achieving high accuracy scores on both datasets.

1 Introduction

Sentiment analysis, also known as opinion mining, is the use of natural language processing, text analysis, and computational linguistics to identify and extract subjective information from source materials. This process is commonly used to determine the attitude, opinions, and emotions of a speaker or writer with respect to some topic or the overall contextual polarity of a document. Sentiment analysis is widely applied in brand monitoring, and social media monitoring in order to gain an overview of a public opinion on a particular product or topic. In marketing, it can be used to track brand reputation, identify brand ambassadors and detect potential problems. In customer service, it can help to improve customer satisfaction by identifying and addressing customer complaints or feedback. In social media monitoring, it can be used to track public opinion and sentiment towards a particular topic or brand.

In this work, we propose a method for sentiment analysis using machine learning techniques on a two datasets of food reviews. We compare the performance of several different deep learning techniques such as RNN, CNN, fastText , AttBiRNN, HAN, LSTM, Bert etc.

2 Literature review

Scientists are actively researching sentiment analysis which has become the biggest area of research in the last few years. Sultana, Kumar (Sultana et al., 2019) described that sentimental analysis has three important aspects, positive, negative, and neutral. In the last few years, the world wide web becomes a key factor in customers reviews; by social media and e-commerce websites, such as Facebook, Twitter user can share their reviews, and these reviews can be good or bad, and these reviews help in making choices about applying new plan and decisions about products.

Chen, Xue (Chen at al. 2019) introduced a new technique to remove the traits of sentiment analysis for the reviews of products. The most common TF-IDF vectors can obtain by using the same form of synonyms by viewing the products' reviews; we can categorize the sequences of feature vectors along with clustering algorithms. By applying this technique, we can refine span algorithms for pseudo-consecutive phrases with FPCD having word order details. By using the last steps, the text feature is gathered. As a result of applying the different mechanisms of performance can be enhanced.

In Abbas, Memon (Abbas et al., 2019), the authors introduced a new heuristic method and naïve bias for specified issues. An MNB is an NB classifier used for text categorization and implemented for sentimental analysis. The results of high data references verify the efficiency of the used algorithms.

In Neethu and Rajasree (Neethu et al., 2013), the authors examined Twitter posts by using ML techniques for different products like mobile, pad, laptop, etc. these strategies applied to Twitter sentiment analysis. Using sentimental analysis, it is easy to explore the main consequences of sentiment analysis. Some issues can create, and to resolve these issues, feature extraction can do after preprocessing in two steps. In the first step, features are firstly removed from tweets and then done features extraction, and then added to the feature vector. Feature classification is done by applying classifiers like NB, SVM, and maximum entropy.

Hua Feng and Ruixi Leng (Feng et al., 2016) perform sentiment analysis using 4 different deep learning recurrent models on the Amazon Fine Food Reviews dataset also used in this paper. In their study, they use two ways to deal with imbalance among comments. In the first, they exclude the over-represented class '5' from the study by narrowing down the task of classifying to the remaining four classes with similar counts. In the second approach, they sample from each class the size of the least represented class, which simultaneously has the effect of significantly reducing the size of the total sample. For the first approach, the authors obtain an accuracy score of 68.75% for the second approach it is 51.74

3 Data Collection

3.1 Amazon Food Review Dataset

The primary dataset we plan to use is the Amazon Fine Food Reviews dataset. It contains over 560k reviews of fine foods from amazon, gathered over 10 years, up to October 2012. The comments come from over 260k different users and concerns 74k+ products. Among all users, 260 have given more than 50 opinions. It is publicly available on the Kaggle platform. Each row consists of 10 features in turn: Id, ProductId, UserId, ProfileName, HelpfulnessNominator, HelpfulnessDenominator, Score, Time, Summary and Text. From the perspective of this research, the most important focus lies on the column containing the content of the review – Text - and the column containing numerical total Score, that lies on a five-point scale.

3.2 Food Recipes Dataset

The second dataset is one we created and used during the previous NLP project. It consists comments and reviews left on the list of 100 most popular recipes presented by the well-known cooking recipe website tasteofhome.com. We used the Selenium library to scrape the list of ingredients from each of the HTML pages. As for the comments, we found and used the hidden backend API used by the site, which avoided the trouble of loading all the comments on the HTML page, and gave us some additional data that would be otherwise difficult to scrape.

In total, we obtained 100 lists of ingredients and 18182 comments, gathered in two CSV files. The ingredients dataset simply contains name of the recipe, and scraped text with the list of ingredients. The comment dataset contains a bit more information, which could be useful for some machine learning problems, as well as for scraping additional related data from the site. For the purposes of this project, we have balanced the data.

4 Exploratory Data Analysis

The histogram in figure 1 shows the distribution of comments by the recipe ranking. As expected, the recipe's popularity is positively correlated with the number of comments and the data appears to be following some form of power law distribution.

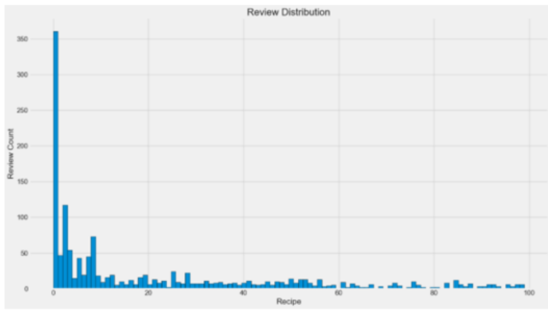

Figure 1: Review Distribution

The figures 2 and 3 show the comment's character count and word count distributions respectively. Most comments are rather short, averaging around 41 words or 220 characters. There are no zero-length comments, as those are impossible to post one the site, but there are a few single character comments. The longest comment has 293 words and 1602 characters.

The figure 4 shows the distribution of the review scores given by the comments, on a range from 1 to 5 stars. Comments with no review attached were removed. In the original dataset, the vast majority of the reviews gave the maximum score. This is to be expected, not only because online reviews of anything are in general mostly positive, but also because this represents the list of the most popular recipes. This could result in serious difficulties when training our model, so we have

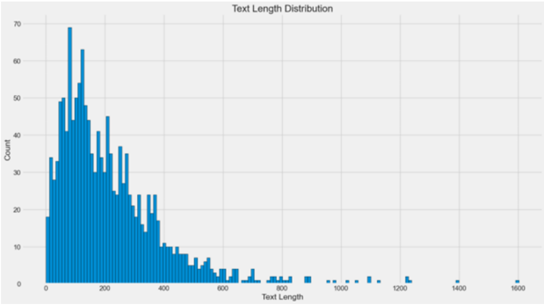

Figure 2: Review length (character count) Distribution

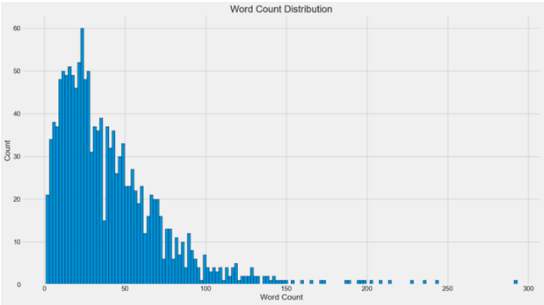

Figure 3: Review length (word count) Distribution

balanced the data to a reasonable degree, by removing samples from the most populated classes.

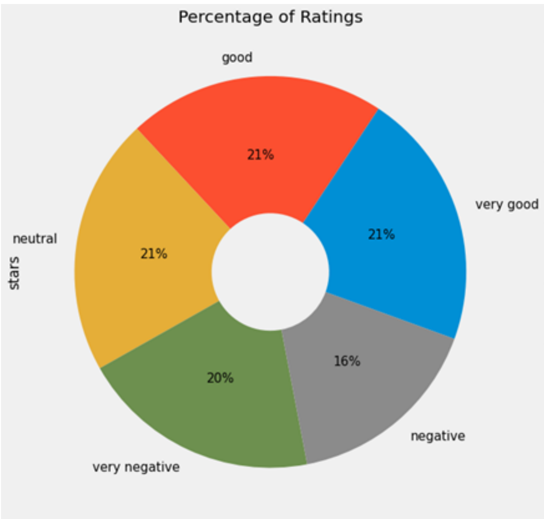

Figure 4: Rating Distribution

On order to better understand the context and sentiment of the text, we used the TextBlob library to find the estimated sentiment polarity of the comments. The results are present in figure 5. The polarity score ranges between -1 and 1, where positive values represent a positive sentiment and negative values represent a negative sentiment. The graph shows that more than 88% of the comments have a positive sentiment. However, the majority of them to be less extreme then could have been expected from the ratings distribution, bringing the mean value to only 0.36.

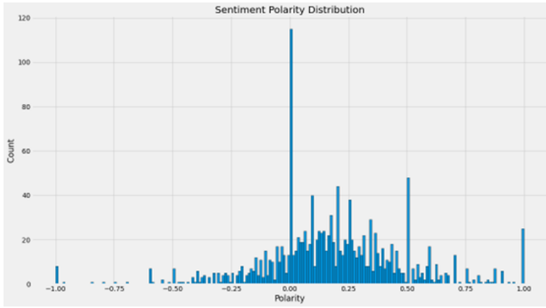

Figure 5: Sentiment Polarity Distribution

5 Data Preprocessing

5.1 Text Cleaning

5.1.1 Lower Case

In this step, we convert all the text into lowercase letters. Lowercasing text is a common preprocessing step in natural language processing tasks, including sentiment analysis. The purpose here is to make the text more consistent, making it easier to process and analyze.

5.1.2 Remove Punctuation

Removing punctuation from text is another common pre-processing step in natural language processing tasks, including sentiment analysis. There are several ways to remove punctuation from text in Python. The method that we used is string.punctuation property, which contains a string of all ASCII punctuation characters.

5.1.3 Remove Special Characters

Removing punctuation from text is another common pre-processing step where we remove a particular letter to make the text clean. There are several ways to remove special characters from text in Python. One standard method we use is the re-library to match and remove special characters with regular expressions.

5.1.4 Remove Digits

In this step, we remove the digits because numbers are not necessary when working, especially on sentiment analysis. To do we use re-library to match and remove digits with regular expressions.

5.2 Preprocessing Operations

5.2.1 Tokenization

Tokenization is the process of breaking down a text into individual words, phrases, or symbols. There are several ways to tokenize text in Python. One standard method we use is the split() method to break the text into a list of words.

5.2.2 Remove StopWords

As we deal with textual data of recipe review. If we see our target as extracting the ingredients from the text then words like A, The, IS, ARE, etc don't necessary. So, it's very important to remove such words which have no meaning.

5.2.3 Lemmatization

Lemmatization is a process where we can derive words into root words. For instance, both "tomatoes" and "Tomato" are output as "tomato". The purpose is to categorize into 1 word.

5.3 Feature Extraction

5.3.1 Bag of Word Model

Bag of Words (BoW) is a commonly used method for feature extraction in natural language processing tasks, including sentiment analysis. The basic idea behind the BoW model is to represent text as a bag (or unordered set) of its words, disregarding grammar and word order but keeping track of the frequency of each word.

In Python, we use the CountVectorizer class from the sklearn library, which helps to create a BoW representation of text.

5.3.2 TF-IDF Model

TF-IDF (term frequency-inverse document frequency) is a statistical method for feature extraction in natural language processing tasks, including sentiment analysis. It is similar to the bag-of-words (BoW) model but takes into account the importance of each word in the text.

TF-IDF is a combination of two values:

Term frequency (TF): it measures how frequently a word appears in a document.

Inverse Document Frequency (IDF): it measures how important a word is across all documents in the corpus.

The TF-IDF value for a word in a document is calculated as the product of its TF and IDF values. Words with a high TF-IDF value are more critical and informative than words with a low TF-IDF value. In Python, we use the TfidfVectorizer class from the sklearn library to create a TF-IDF representation of text.

5.3.3 Word2Vec Model

Word2Vec is a method for feature extraction in natural language processing tasks, including sentiment analysis. It is a neural network-based approach that learns distributed representations, also known as word embeddings, for words from large amounts of unstructured text data.

The basic idea behind Word2Vec is to use a neural network to learn a high-dimensional vector representation for each word in the vocabulary. These vectors capture the meaning and context of words in a numerical form, which can be used as input features for various NLP tasks such as sentiment analysis, text classification, and machine translation.

In Python, we use the gensim library to provide an implementation of the Word2Vec model. Word2Vec is helpful in feature extraction because it provides a dense vector representation of each word by capturing the context, meaning, and relationship between words.

5.3.4 One-Hot Encoding

One-hot encoding is applied after stemming. The one_hot function from the tensorflow.keras.preprocessing.text module used to perform this encoding. It takes in the text data and a vocabulary size, which is the maximum number of unique words to consider in the encoding. In this case, the vocabulary size is set to 5000. Then we applied the pad_sequences function from the tensorflow.keras.preprocessing.sequence module to ensure that all the encoded sequences have the same length. This will be useful when we are training a deep learning model because as input sequences with different lengths can be challenging to process. The pad_sequences function takes in a list of sequences and the desired sequence length and pads or truncates the sequences to that length. In this case, the desired sequence length is set to 30.

6 Proposed System Architecture

To correctly Identity the emotion from the recipe review, we experiment with different machine learning models we explained below. And then, we compared the performance of these models and also two different datasets on the task of identify the emotion from recipes review. This process involves training each model on the training set, evaluating its performance on the test set, and comparing the results.

6.1 fastText

In the process of identify the emotion from recipes review using fastText, a look-up table is first used to convert bags of ngrams (sequences of n consecutive words) into word representations. These word representations are then averaged to create a text representation, which is a hidden variable that captures the meaning of the text. The text representation is fed to a linear classifier, which uses it to predict the probability of the text belonging to one of the predefined classes (in our case, the classes is five different review retings). The classifier uses the softmax function to compute the probability distribution over the classes, and the class with the highest probability is chosen as the prediction.

6.2 Recurrent Neural Network

In the process of identify the emotion from recipes review using Text Recurrent Neural Network (TextRNN), the input text is first processed and transformed into a numerical representation, such as a sequence of word embeddings. This representation is then fed into the TextRNN model, which consists of one or more layers of recurrent cells. The recurrent cells process the input sequence one element at a time, using the hidden state from the previous element to inform the prediction for the current element. This allows the TextRNN to capture long-term dependencies and patterns in the data. After the input sequence has been processed, the final hidden state is used to make the prediction, which is typically done using a fully connected layer with a softmax output. The softmax function is used to compute the probability distribution over the predefined classes.

6.3 Bidirectional Recurrent Neural Network

In the process of identify the emotion from recipes using Text bidirectional RNN, the first step is to represent the sentence as a sequence of word embeddings. These word embeddings capture the meaning of the words in the sentence and are learned from the data during training. The word embeddings are then fed into a bidirectional RNN, which processes the sequence in both forward and backward directions. This allows the model to capture the context of the words in the sentence and use it to make the prediction. The output of the bidirectional RNN is then fed into a fully connected layer with dropout, and the softmax function is used to compute the probability distribution over the predefined classes.

6.4 Attention Bidirectional Recurrent Neural Network

In the process of identify the emotion from recipes using using Text Attention bidirectional RNN, the model first encodes the input text into a sequence of hidden states using a bi-directional RNN. The hidden states capture the contextual information of the words in the text. Next, an attention mechanism is applied to the hidden states to weight the importance of each word in the input text. The weighted hidden states are then passed through a fully connected layer to make the prediction.

6.5 Hierarchical Attention Network

In the process of identify the emotion from recipes using a Hierarchical Attention Network(HAN), the model first encodes the ingredients into a fixed-length representation using word embeddings and a bi-directional GRU (Gated Recurrent Unit) layer. The GRU layer is a type of recurrent neural network that is used to process sequential data, such as text. The encoded reviews are then passed through an attention layer, which weights the importance of each word in the foodreview based on its relevance to the task of predicting the sentiment. Finally, the weighted sum of the encoded ingredients is passed through a fully connected layer with a SoftMax output to predict the probability distribution over the predefined classes. The class with the highest probability is chosen as the prediction.

6.6 Convolutional Neural Network

In the process of identify the emotion from recipes using Text Convolutional Neural Network (TextCNN), the first step is to represent the sentence with static and non-static channels. The static channel captures the meaning of the words in the sentence, while the non-static channel captures the context in which the words are used. Next, the representation is convolved with multiple filter widths and feature maps to extract features from the text. Max-over-time pooling is then used to reduce the dimensionality of the representation. Finally, a fully connected layer with dropout is used to make the prediction, and the softmax function is used to compute the probability distribution over the predefined classes.

6.7 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) that is well suited for processing sequential data with long-term dependencies, such as text. LSTMs are able to maintain information about previous steps in the sequence over a prolonged period of time by using gates to allow information to pass through the hidden state selectively.

In natural language processing, LSTMs are often used for tasks such as text classification. They can be trained to encode a given input text into a fixed-length vector representation, which can then be used as input to a classifier or other model.

6.8 Bert Model

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained transformer-based model developed by Google for natural language processing tasks. BERT is trained on a massive amount of text data and can be fine-tuned for a variety of tasks, such as text classificatio in our case sentiment analysis.

One of the key innovations of BERT is its use of bidirectional attention, which allows the model to take into account the context of a word in both the left and right directions of the input sequence. This allows BERT to better understand the meaning of a word in the context of the entire sentence, which is important for many natural language processing tasks.

7 Results and discussion

The results from the models that we trained on the Amazon Food Review dataset show that the BERT model has the highest accuracy at 72%, followed by the Recurrent Neural Network (RNN) and the Bidirectional RNN (BIRNN) at 68% and 69% respectively. The Attention-based Bidirectional RNN (AttBiRNN) and the Long Short-Term Memory (LSTM) models both have an accuracy of 70%. The fastText model has the lowest accuracy at 51%, while the CNN model an accuracy of 60%.

It is worth noting that the BERT model, which is a transformer-based model, has significantly higher accuracy compared to other models, this may be because it is pre-trained on a large corpus of text data and fine-tuned on the specific task, which allows it to understand the context and semantics of the text better. Additionally, the bidirectional RNN models (BIRNN and AttBiRNN) may have performed better than the unidirectional RNN model due to their ability to take into account both past and future context.

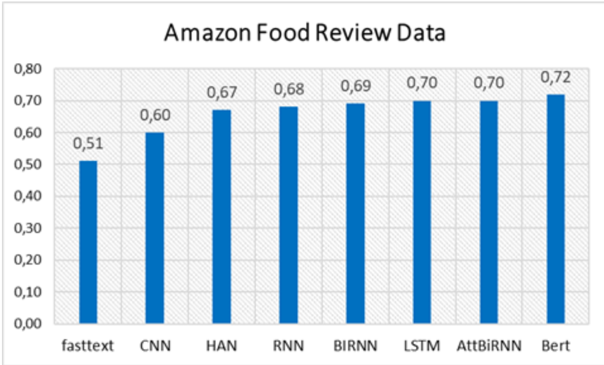

Figure 6: Results of Amazon Food Review Data

The results from the models that trained on the food recipes dataset for sentiment analysis show that the Bidirectional RNN (BIRNN) model has the highest accuracy at 70%, followed by the Hierarchical Attention Network (HAN) at 63%. The BERT model has an accuracy of 69%, while the Attention-based Bidirectional RNN (AttBiRNN) has an accuracy of 68%. The Recurrent Neural Network (RNN) and the Long Short-Term Memory (LSTM) models both have an accuracy of 61% and 42% respectively. The CNN model has an accuracy of 43%. The fastText model has the lowest accuracy at 20%.

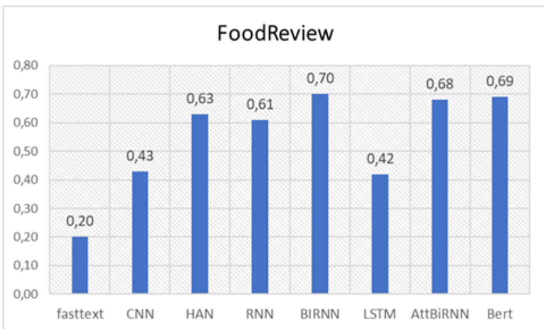

Figure 7: Results of Food Review Data

It is worth noting that the bidirectional RNN models (BIRNN and AttBiRNN) performed better than the unidirectional RNN and LSTM models, this may be because bidirectional RNNs take into account both past and future context, which is important for sentiment analysis as the meaning of a sentence can change depending on the context. Additionally, the HAN model also performed well, this may be because it allows the model to focus on different parts of the input text, such as individual words and phrases, which can be useful for sentiment analysis.

It's also worth noting that the fastText model performed significantly worse than the other models, which may be because it is not as well-suited for this specific task or limited dataset compared to amazon food dataset. The CNN model also performed poorly, this may be because it is not well-suited for text classification tasks or that the dataset is not large enough for the model to learn effectively.

8 Conclusions and future works

In conclusion, the results from the models trained on the own food review data and amzon food review dataset for sentiment analysis indicate that the bidirectional RNN models, HAN, and BERT model performed the best. The BIRNN model achieved the highest accuracy of 70%, followed by HAN with 63% and BERT with 69%, 72%. While the AttBiRNN, RNN, and LSTM models all have accuracy scores in the range of 61-68%. The fastText model and CNN models had the lowest accuracy scores.

It's worth noting that bidirectional RNN models and HAN may performed well due to their ability to take into account both past and future context, which is important for sentiment analysis as the meaning of a sentence can change depending on the context. Additionally, BERT model also performed really well as it is pre-trained on a large corpus of text data and fine-tuned on the specific task, which allows it to understand the context and semantics of the text better. However, it's important to consider other metrics and also analyze the performance of the models on different subsets of the data to get a more comprehensive understanding of their strengths and weaknesses.

One area for future work would be to perform a more thorough hyperparameter tuning of the models to see if their performance can be improved. Additionally, combining the predictions of multiple models, such as the bidirectional RNN and BERT, may result in a more robust and accurate model.

Author	Contribution
Amir Ali	Literature Review, Data Preprocessing, Exploratory Data Analysis, Feature Engineering, System Architecture, Report, Presentation
Stanisław Matuszewski	Data Collection, Literature Review, Report, Presentation, GitHub, Overleaf
Jacek Czupyt	Exploratory Data Analysis, Report, Presentation, System Architecture

Table 1: Division of work

Contribution

References

Yin, W., Kann, K., Yu, M. & Schütze, H. Comparative Study of CNN and RNN for Natural Language Processing. *CoRR*. **abs/1702.01923** (2017), http://arxiv.org/abs/1702.01923

Solovyeva, E. & Abdullah, A. Binary and Multiclass Text Classification by Means of Separable Convolutional Neural Network. *Inventions*. **6** (2021), https://www.mdpi.com/2411-5134/6/4/70

Mahoney, M. Fast Text Compression with Neural Networks. *The Florida AI Research Society*. (2000)

Du, C. & Huang, L. Text Classification Research with Attention-based Recurrent Neural Networks. *International Journal Of Computers Communications Control*. **13** pp. 50 (2018,2)

Pappas, N. & Popescu-Belis, A. Multilingual Hierarchical Attention Networks for Document Classification. *CoRR*. **abs/1707.00896** (2017), http://arxiv.org/abs/1707.00896

Bai, X. Text classification based on LSTM and attention. (2018,9)

Sun, C., Qiu, X., Xu, Y. & Huang, X. How to Fine-Tune BERT for Text Classification?. *CoRR*. **abs/1905.05583** (2019), http://arxiv.org/abs/1905.05583

Sultana, N., Kumar, P., Patra, M., Chandra, S. & Alam, S. SENTIMENT ANALYSIS FOR PRODUCT REVIEW. *International Journal Of Soft Computing*. **9** pp. 7 (2019,4)

Chen, X., Xue, Y., Zhao, H., Lu, X., Hu, X. & Ma, Z. A Novel Feature Extraction Methodology for Sentiment Analysis of Product Reviews. *Neural Comput. Appl.*. **31**, 6625-6642 (2019,10), https://doi.org/10.1007/s00521-018-3477-2

Abbas, M., Ali, K., Memon, S., Jamali, A., Memon, S. & Ahmed, A. Multinomial Naive Bayes Classification Model for Sentiment Analysis. (2019,3)

Neethu, M. & Rajasree, R. Sentiment analysis in twitter using machine learning techniques. *2013 Fourth International Conference On Computing, Communications And Networking Technologies (ICCCNT)*. pp. 1-5 (2013)

Feng, H. & Lin, R. Sentiment Classification of Food Reviews. *CoRR*. **abs/1609.01933** (2016), http://arxiv.org/abs/1609.01933