NC STATE UNIVERSITY

Sentiment Analysis of amazon Reviews

Project ID: P20

Members:

Kaushik Arvind Jadhav (kajadhav)

Shreya Maheshwari (smahesh4)

Devashish Vachhani (dvachha)

Palash Pravin Rathod (prathod)

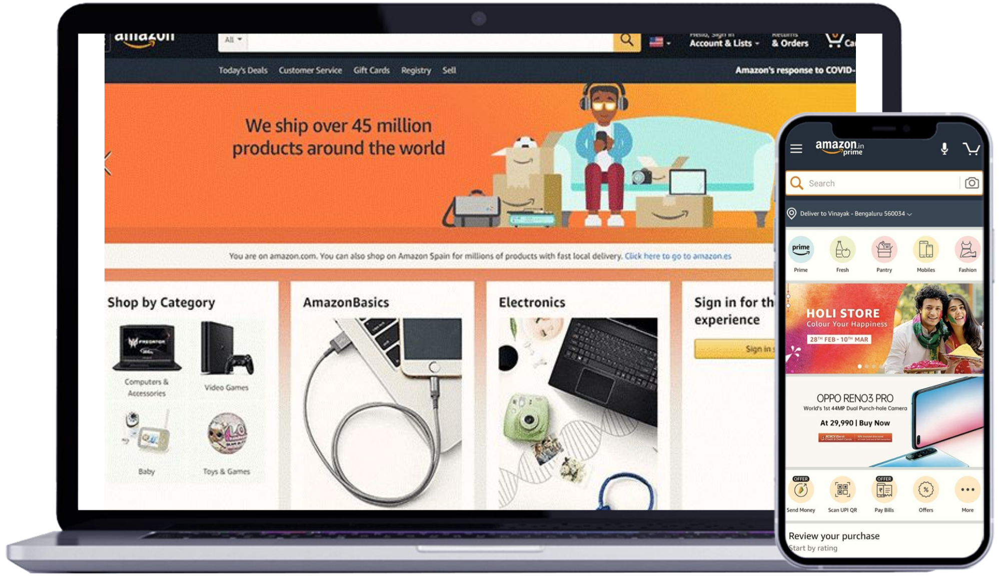
Course: CSC 522 - Automated Learning and Data Analysis

Instructor: Dr. Min Chi

Contact TA: Md Mirajul Islam

Contents

Introduction



There are billions of products on e-commerce sites making it difficult for businesses to identify good and bad products so that they can promote the good products & remove the bad ones to increase customer satisfaction and thereby, the revenue.

Sentiment Analysis is an application of NLP wherein human sentiment in text is analyzed.

In this project, we train 4 machine Learning and 1 deep learning model to generate positive or negative sentiments in Amazon product reviews and compare their effectiveness in categorizing the products.

Data Collection

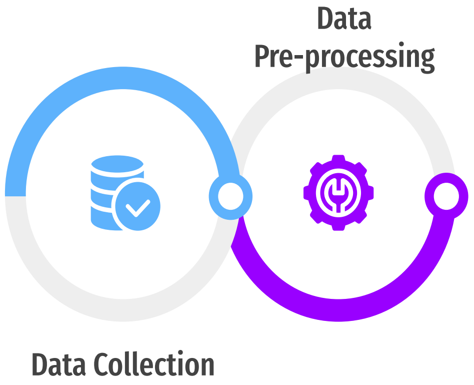
For this project, we have used the **Amazon Kindle Reviews** dataset downloaded from Kaggle. The dataset has a total of *982619* product reviews given from from May 1996 - July 2014. Each reviewer has at least 5 reviews and each product has at least 5 reviews in this dataset.

Data Collection

Data Pre-Processing

Due to computational power limitations, we only take a subset (*10%* of the rows) of the dataset for our analysis. The original dataset is unclean and so we apply the following pre-processing techniques to it:

1. **Handling Missing Values:** The dataset contains *22* missing values in reviewText and *3816* missing values in reviewerName column. Out of the total *98261* rows, as the count of missing values is relatively smaller, we can simply drop the rows with missing values.
2. **Removing unwanted columns:** We observe that columns like 'Unnamed: 0' and 'reviewerName' in the dataset are unnecessary. So, we simply drop them.
3. **Tokenization:** We summarize the most important information in reviews by applying techniques like Tokenization, Stemming & Lemmatization. Tokenization is the breaking down complex data like paragraphs into their smallest unit called a token, which can be punctuation marks, words, or numbers.
4. **Stemming:** Stemming is a normalization technique where list of tokenized words are converted into shortened root words to remove redundancy. We do this using *PorterStemmer()* method of nltk library.
5. **Lemmatization:** Stemming gives intermediate representation of the token and may or may not give meaningful words. So, we apply Lemmatization which gives the form of related word in dictionary order using *WordNetLemmatizer()*.



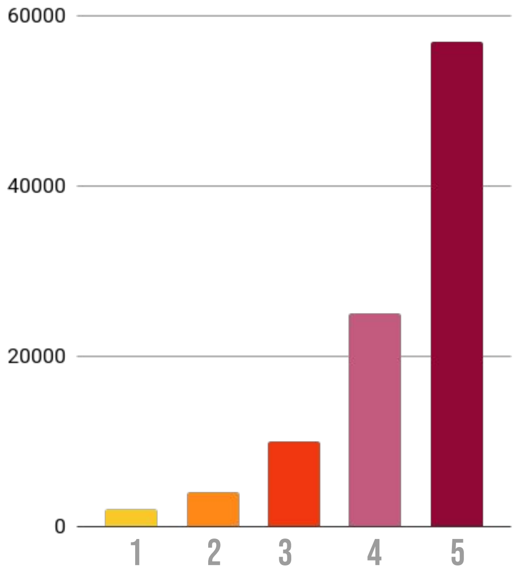
Sentiment Modeling

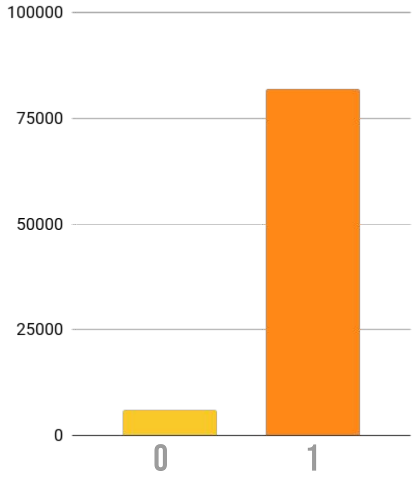
The graph alongside shows the distribution of ratings in the ‘overall’ column of our dataset. We observe that there are 5 ratings 1 being the worst, 3 being neutral and 5 being the best rating. To train our models on the semantic meaning of ratings, we add a column of Sentiment_Score to our dataset such that:

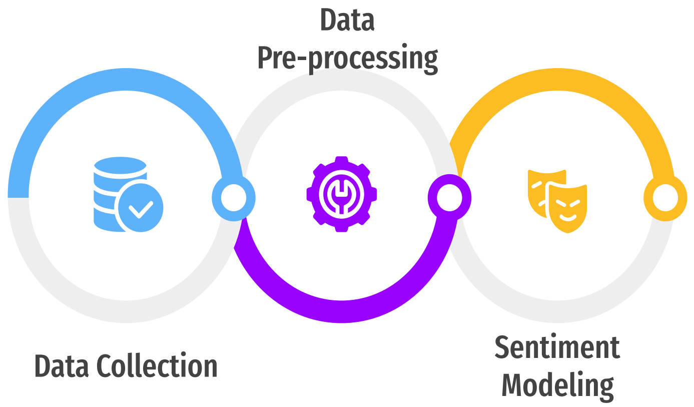
Rating 1,2 = **0**, that is negative sentiment

Rating 4,5 = **1**, that is positive sentiment

Rating 3 rows will be removed

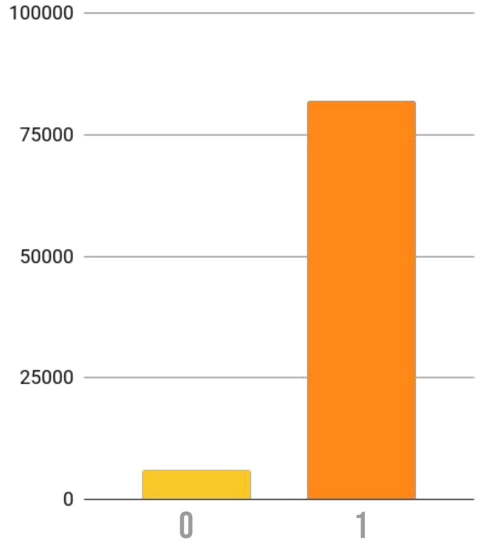


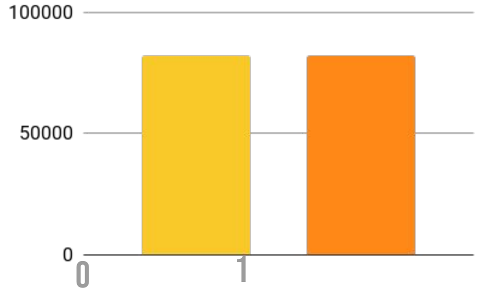


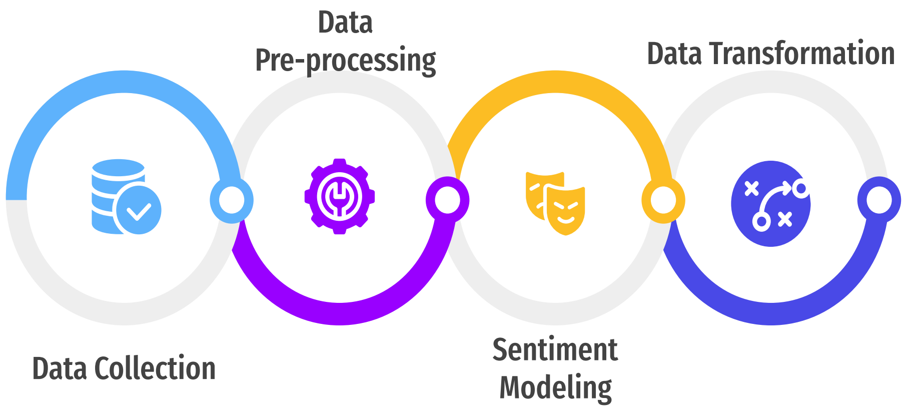


Data Transformation

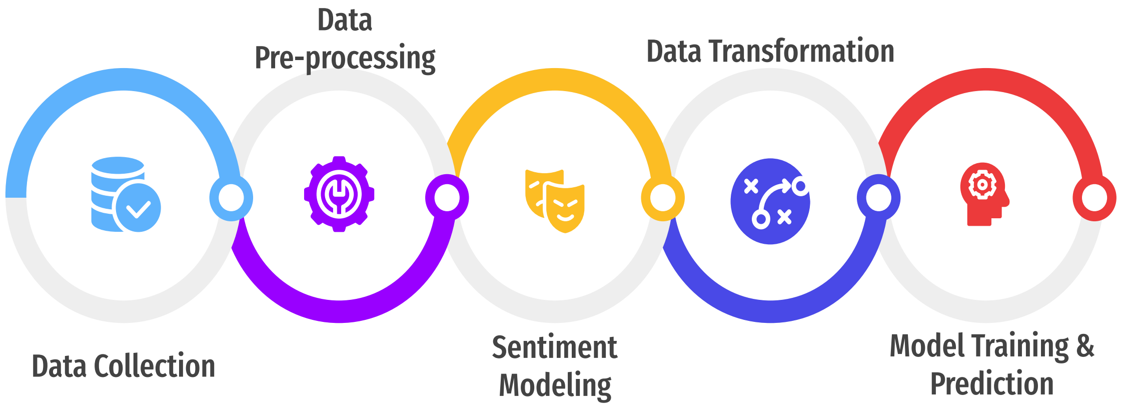
1. **Handling Imbalanced Data:** From the graph alongside, we see that the data is highly imbalanced as it contains very few rows with 0 Sentiment Score. To balance this, we use *RandomOverSampler*.
2. **Vectorization:** We cannot directly feed our cleaned reviews to any machine learning model because machine learning models take numbers as input while our reviews are in string format. So, we use *CountVectorizer* to convert the cleaned reviews into vectors of frequency count of each word that occurs in the entire text.







Model Training and Prediction



Multinomial Naive Bayes

- The basic ideology behind a Naive Bayes model is the assumption that all the features are conditionally independent of one another given some class.
- With a standard *sklearn* implementation of a Naive Bayes Classifier (alpha: *1.0*), we obtained a testing accuracy of 94%, precision of 96%, recall of 92% and an F-1 score of 94%

Decision Tree

- A decision tree classifier uses a tree-like structure to model decisions. Each internal node denotes an attribute, each branch represents an outcome, and each leaf node holds a class label.
- With a standard *sklearn* implementation of a Decision Tree Classifier (criterion: *gini*, min_samples_split: *2*), we obtained a testing accuracy of 96%, precision of 100%, recall of 93% and an F-1 score of 96%

Logistic Regression

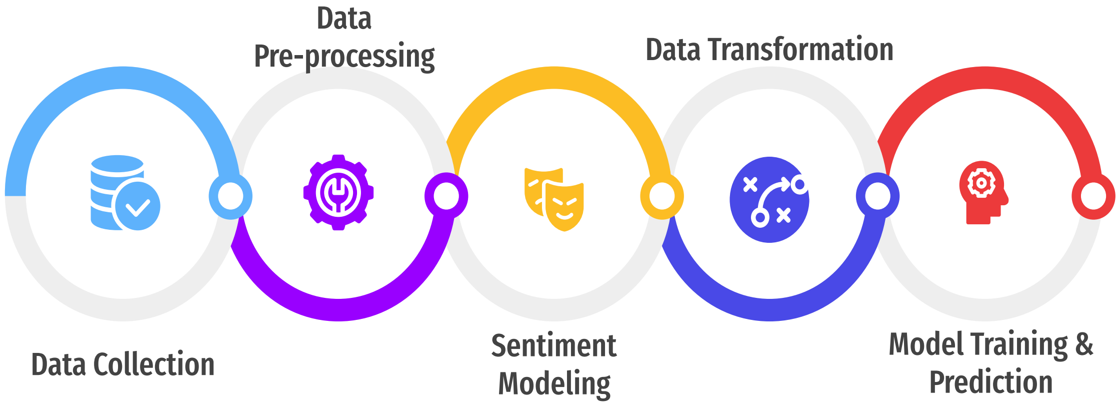
- Logistic regression is one of the most widely used classification algorithms and performs very well on classes which are linearly separable.
- With a standard *sklearn* implementation of a Logistic Regression Classifier (penalty: *l2*, C: *1.0,* solver*: liblinear*), we obtained a testing accuracy of 97%, precision of 99%, recall of 96% and an F-1 score of 97%

K-Nearest Neighbour

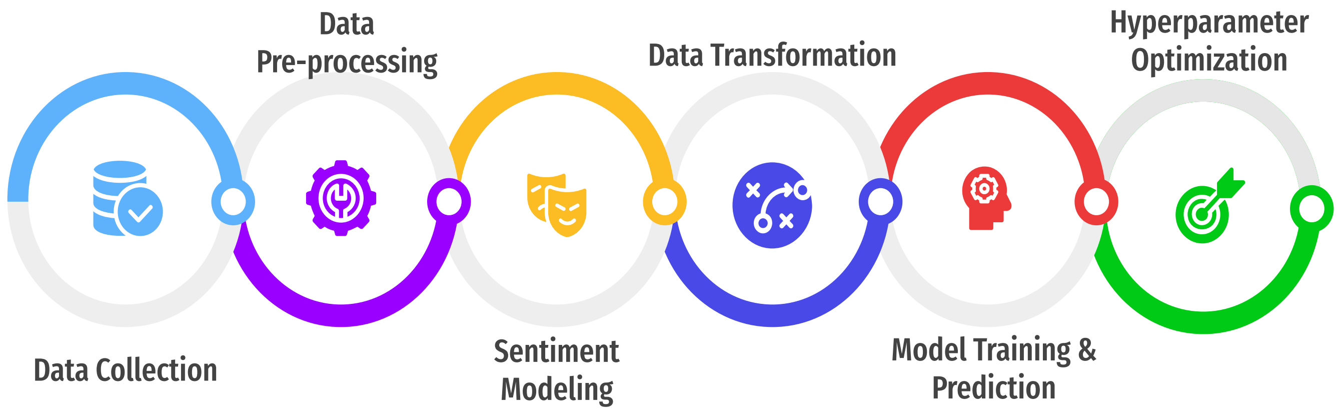
- The K-Nearest Neighbour (KNN) algorithm is a supervised machine learning algorithm used for estimating the likelihood that the point will become a member of one group or another depending on the labels of the data points near it.
- With a standard *sklearn* implementation of kNN Classifier (neighbors: 1), we obtained a testing accuracy of 95%, precision of 100%, recall of 91% and an F-1 score of 95%.

Long Short Term Memory (LSTM)

- We first tokenize the product reviews using *Tokenizer* & then use *pad_sequences* keras function to vectorize them into 2D tensors suitable for LSTM training.
- While tokenizing, we use the top 20000 words in our corpus and so the first layer of the LSTM will be an Embedding layer which takes **20000** rows as input and produces embeddings of shape **128**.
- Next we add LSTM layer with **128** neurons to take the previous embeddings as input and finally add our final output layer with 2 neurons (because of 2 classes).
- Activation function used is *Softmax* and loss function is *binary_crossentropy*.
- The model got a final accuracy of **98.1%**.



Hyperparameter Optimization



To improve the accuracy of the standard models, we use *GridSearchCV* to tune the hyperparameters

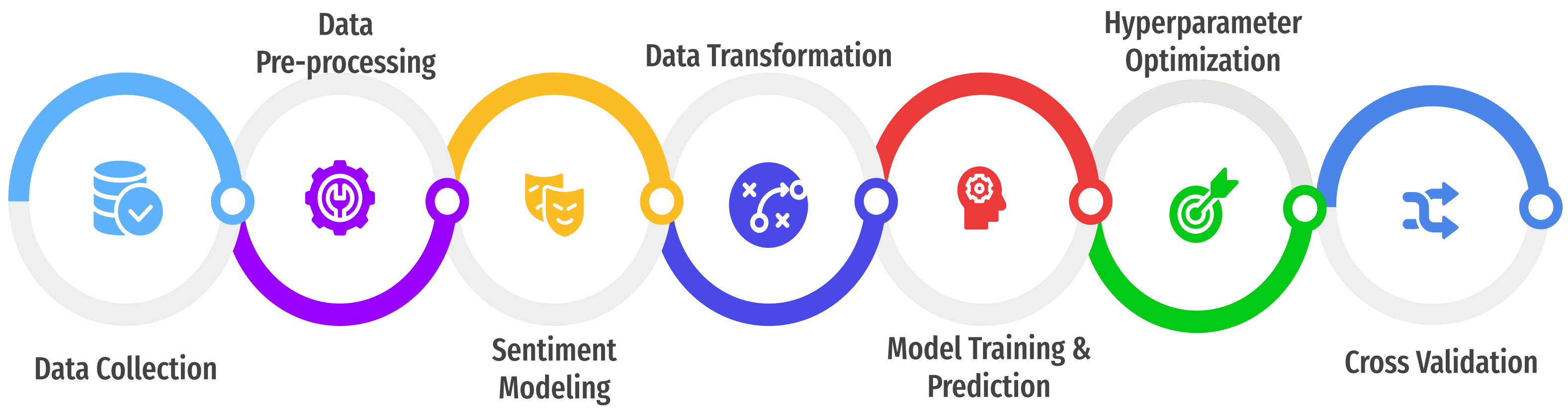
Model	Optimal Parameters	Accuracy	Precision	Recall	F-1 Score
Multinomial Naive Bayes	clf__alpha: 0.0001	95%	97%	94%	95%
Decision Tree	criterion: entropy, min_samples_split: 2	97%	100%	94%	97%
Logistic Regression	C:1.0, penalty: l2	97%	99%	95%	97%
K-Nearest Neighbour	neighbors: 1	95%	100%	91%	95%

Stratified K-fold Cross Validation

Cross Validation: We perform cross validation to make sure that our model is generalized and works well for new, unseen data.

K-fold Cross Validation: We choose k=5 for our cross validation and run 5 splits. In each split, 4 folds of the data will be used for training and the remaining 1 fold for testing.

Stratified K-fold Cross Validation: K-fold Cross Validation selects data by random sampling. So there might be a case where all points in the testing set of a fold belong to a single class (either all 0s or all 1s). To avoid this, we used Stratified K-fold Cross Validation because it selects data by stratified sampling which ensures that data of both the classes is uniformly distributed in each set.



Conclusion

We used both Machine Learning & Deep Learning models to analyze their effectiveness in predicting the positive or negative sentiment of product reviews. Of all the Machine Learning models, Logistic Regression had the highest accuracy of 97.5% followed by Decision tree which was second highest, 96.91%. However, Deep Learning outperformed Machine Learning as we got 98.1% accuracy for LSTM. Initially, the accuracies for all models were different, but we got the aforementioned figures after applying proper Hyperparameter Optimization, Vectorization and Cross Validation.

References

- Wassan, Sobia & Chen, Xi & Shen, Tian & Zaman, Noor. (2021). Amazon Product Sentiment Analysis using Machine Learning Techniques. 30. 695-703. 10.24205/03276716.2020.2065.
- Abbas, Muhammad & Ali, Kamran & Memon, Saleem & Jamali, Abdul & Memon, Saleemullah & Ahmed, Anees. (2019). Multinomial Naive Bayes Classification Model for Sentiment Analysis. 10.13140/RG.2.2.30021.40169.
- Wankhade, Mayur & Chandra, A & Rao, Sekhara & Dara, Suresh & Kaushik, Baij. (2017). A Sentiment Analysis of Food Review using Logistic Regression. 2456-3307.
- Murthy, Dr & Allu, Shanmukha & Andhavarapu, Bhargavi & Bagadi, Mounika. (2020). Text based Sentiment Analysis using LSTM. International Journal of Engineering Research and. V9. 10.17577/IJERTV9IS050290.

Thanks!