

# Sentiment Analysis of Amazon review dataset

Aleksandra Wieczorek

March 2024

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Data Analysis</b>	<b>3</b>
2.1	Exploration of dataset . . . . .	3
<b>3</b>	<b>Data Cleaning Process</b>	<b>4</b>
3.1	Text Data Processing . . . . .	5
3.2	Sentiment Analysis Preparation . . . . .	6
<b>4</b>	<b>Machine Learning</b>	<b>6</b>
4.1	Basic information of Machine Learning models . . . . .	7
4.2	K-Nearest Neighbmrs . . . . .	8
4.3	Logistic Regression . . . . .	9
4.4	Random Forest . . . . .	9
<b>5</b>	<b>Results</b>	<b>11</b>
<b>6</b>	<b>Conclusion</b>	<b>12</b>

# 1 Introduction

In this report, I explore the Amazon reviews dataset, available on Kaggle Ipage[1]. I begin with an initial understanding of the dataset's origin and a detailed examination of its structure. After understanding the dataset, my focus shifts to preparing it for analysis. This involves cleaning and manipulating the dataset to ensure it's ready for the subsequent stages of investigation.

With the data in a usable state, I proceed to explore several machine learning models. I talk about strengths and weaknesses of each model and I show practical application of these models, involves training and testing them on my cleaned dataset.

The culmination of my efforts is a comparative analysis of the performance of each machine learning model and identifying the model that performs best.

## 2 Data Analysis

In this part of report, I delve into the initial steps of managing and preparing dataset for analysis. After obtaining a CSV file from Kaggle, I loaded it into a Pandas dataframe named `amazon_reviews`. This step is crucial as it allows to utilize Python's data manipulation tools to explore and prepare data.

Subsequently, I established a connection to a MySQL database using the `create_engine` method from SQLAlchemy. This process involves specifying my database credentials and the location of my database, enabling Python environment to communicate directly with MySQL.

The next step was transferring the `amazon_reviews` dataframe into MySQL database as a table. This action was performed using the `to_sql` method, which conveniently maps the dataframe into a SQL table in the specified database without losing the structure or the integrity of the data.

Storing the dataset as a table in a MySQL database offers numerous advantages:

- **Data Persistence:** Unlike dataframes that exist temporarily during the execution of a script, data stored in a database is persistent and can be accessed or modified at any time, providing a reliable storage solution.
- **Scalability:** Databases are designed to handle large volumes of data efficiently. As dataset grows, the MySQL database can manage the increased load without significant degradation in performance.
- **Security:** MySQL databases offer security features, including access control and encrypted connections, ensuring that data is protected against unauthorized access.
- **Collaboration:** By storing data in a centralized database, multiple users can work on the data concurrently, facilitating collaboration among team members working on the analysis.

### 2.1 Exploration of dataset

The `amazon_reviews` dataframe provides a comprehensive view of product reviews, showcasing a variety of information across 18 columns and 111,098 records. The dataframe includes both numerical and textual data, with columns such as `reviewID`, `amazon-id`, and `unixReviewTime` containing integer values, and `price` along with `first-release-year` represented as floating-point numbers.

Textual information is stored in columns like `reviewText`, `summary`, `categories`, `label`, `songs`, and `related`, with `helpful` and `reviewTime` also stored as objects indicating more complex or string-based data. Notably, the dataset is almost complete, but there are instances of missing data (NaN values) in four columns: `reviewText`, `summary`, `label`, and `first-release-year`, which could require attention during data cleaning and preprocessing stages.

This diversity in data types (integers, floats, and objects) highlights the dataset's complexity and the need for careful handling, especially when preparing data for machine learning models or in-depth analysis.

```
RangeIndex: 111098 entries, 0 to 111097
Data columns (total 18 columns):
#   Column                Non-Null Count  Dtype
---  ---                ---
0   reviewerID            111098 non-null  int64
1   amazon-id             111098 non-null  int64
2   helpful               111098 non-null  object
3   unixReviewTime        111098 non-null  int64
4   reviewText            111093 non-null  object
5   overall               111098 non-null  int64
6   reviewTime            111098 non-null  object
7   summary               111094 non-null  object
8   price                 111098 non-null  float64
9   categories            111098 non-null  object
10  root-genre            111098 non-null  object
11  title                 111098 non-null  int64
12  artist                111098 non-null  int64
13  label                 111037 non-null  object
14  first-release-year    99826 non-null  float64
15  songs                 111098 non-null  object
16  salesRank             111098 non-null  int64
17  related               111098 non-null  object
dtypes: float64(2), int64(7), object(9)
memory usage: 15.3+ MB
```

### 3 Data Cleaning Process

In the data cleaning phase of project, I took several steps to ensure the integrity and usefulness of dataset, which is crucial for the accuracy of analysis and machine learning models.

Given the substantial size of dataset, with over 111,000 records, I had the flexibility to remove duplicated rows and rows with missing values in certain critical columns without significantly impacting dataset’s comprehensiveness. Specifically, I focused on eliminating rows with NaN values in the `reviewText`, `summary`, and `label` columns. These fields are vital for understanding the context and sentiment of the reviews, making their completeness essential for analysis.

However, I noticed that the `first-release-year` column had 11,272 missing values. Given that this information is not directly relevant to determining the sentiment of a review, I did not remove these rows. This decision helps preserve the robustness of my dataset, ensuring that I do not discard valuable review data unnecessarily.

Additionally, I streamlined dataset by dropping columns that do not contribute significantly to discerning a review’s positive or negative sentiment. These included `label`, `categories`, and `root-genre`. Removing these columns helps focus analysis on the most impactful data, enhancing the efficiency of machine learning models.

Upon examining the `helpful` column, I discovered it contained complex data that could be more insightful if broken down. Therefore, I split this column into two distinct columns: `Helpful_votes` and `All_votes`. This separation allows for a more nuanced analysis of how helpfulness perceptions relate to the sentiment expressed in the reviews, providing valuable insights into consumer behavior and review credibility.

Lastly, I aimed to improve the quality of textual data by removing records with very short reviews, specifically those with less than 10 characters. This step ensures that analysis is based on more substantive and informative reviews, thereby enhancing the reliability of sentiment analysis and the overall quality of dataset.

Through these meticulous data cleaning efforts, I have refined dataset to focus on the most relevant and high-quality information. This preparation lays a solid foundation for the subsequent stages of project, including in-depth data analysis and the development of machine learning models to classify review sentiments

accurately.

```
RangeIndex: 111098 entries, 0 to 111097
Index: 110956 entries, 0 to 111097
Data columns (total 13 columns):
#   Column              Non-Null Count  Dtype
---  -
0   reviewerID           110956 non-null  int64
1   amazon-id            110956 non-null  int64
2   unixReviewTime        110956 non-null  int64
3   reviewText            110956 non-null  object
4   overall              110956 non-null  int64
5   summary              110956 non-null  object
6   price                110956 non-null  float64
7   title                110956 non-null  int64
8   artist               110956 non-null  int64
9   first-release-year    99693 non-null   float64
10  salesRank            110956 non-null  int64
11  helpful_votes         110956 non-null  int64
12  all_votes            110956 non-null  int64
dtypes: float64(2), int64(9), object(2)
memory usage: 11.9+ MB
```

### 3.1 Text Data Processing

At this stage of data preparation, I recognized that among the various columns within my dataset, only two were non-numeric and directly relevant to sentiment analysis: `reviewText` and `summary`. To streamline analysis and consolidate the valuable textual data, I created a new column named `combined_text`, which merges the content of both `reviewText` and `summary`. Subsequently, I removed the original `reviewText` and `summary` columns to simplify dataset further, focusing only on this text data for sentiment analysis.

However, upon utilizing the `langdetect` library to analyze the language of the entries in the `combined_text` column, I discovered that not all records are in English. Recognizing the importance of language consistency for accurate sentiment analysis and machine learning applications, I decided to remove all non-English records from dataset. This is important for several reasons:

- **Consistency:** Machine learning models trained on English text are optimized to understand and analyze the nuances of the English language. Introducing texts in other languages could lead to misunderstandings and inaccuracies in sentiment classification.
- **Feature Extraction:** Many of the techniques for processing and extracting features from text, such as tokenization and stemming, are language-specific. Non-English texts would not be accurately processed using tools designed for English, potentially skewing the analysis.
- **Comparability:** Sentiment analysis depends on comparing the sentiment of texts. Consistency in language ensures that all texts are judged by the same standards and with the same tools, allowing for meaningful comparisons.

To further enhance the clarity and consistency of data, I executed several text preprocessing operations on the `combined_text` column:

1. **Removing Punctuation:** I removed all punctuation marks, as they do not contribute to understanding the sentiment of the text and can complicate the analysis.

2. **Converting to Lowercase:** I converted all text to lowercase to ensure that the analysis does not differentiate between words based solely on capitalization, which is irrelevant to sentiment.
3. **Stemming:** I applied stemming to reduce words to their root form, helping in the normalization of the text data and improving the efficiency of the analysis by treating different forms of the same word as identical.

These preprocessing steps makes text data more suitable for machine learning models to analyze sentiment accurately. By standardizing the format and reducing the complexity of the text, I increase the likelihood of deriving meaningful insights from sentiment analysis, ensuring models can effectively learn from and interpret the data.

### 3.2 Sentiment Analysis Preparation

In the final phase of data preparation, I closely examined the overall column, which rates reviews on a scale from 1 to 5 stars. my analysis revealed a significant imbalance: a majority of the reviews were rated with 5 stars, while the occurrences of ratings 4, 3, 2, or 1 were considerably less frequent. Such a distribution poses a challenge for sentiment analysis, potentially leading to biased machine learning models that are overfitted to predict positive outcomes more accurately than negative ones.

To mitigate this issue and strive for a balanced dataset, I defined my criteria for categorizing reviews as positive or negative, where 5 and 4 are considered positive, while 2 and 1 as well as reviews with a 3-star rating as negative to approximate a more balanced distribution of positive and negative sentiments. In a different scenario, with a more evenly distributed set of ratings, a 3-star review might have been considered neutral instead.

Following this logic, I introduced a new column named `sentiment`, coded with binary values: 1 for positive reviews and 0 for negative reviews. This simplification into a binary classification system aids in the clarity of my analysis and the effectiveness of my machine learning models by providing a clear difference between positive and negative sentiments.

Having refined dataset through various stages of cleaning and manipulation, including language filtering, text preprocessing, and sentiment categorization, I completed my data preparation. The final step was to store the cleaned and processed dataset in MySQL database under the table name `amazon_reviews.cleaned.eng`. This action makes easy access for future data analyses or machine learning.

## 4 Machine Learning

Before proceeding to train machine learning models, I encountered the necessity to process the `combined_text` column, which contains textual data. Machine learning algorithms typically require numerical input. To change `combined_text` , I employed the `TfidfVectorizer` from the sklearn library.

### TfidfVectorizer

`TfidfVectorizer` stands for Term Frequency-Inverse Document Frequency Vectorizer. It transforms text into a numerical representation by calculating two statistics: term frequency (TF) and inverse document frequency (IDF).

**Term Frequency (TF):** measures how frequently a term occurs in a document. Since every document is different in length, the term frequency is often divided by the document length as a way of normalization.

**Inverse Document Frequency (IDF):** measures how important a term is within the entire corpus of documents. The terms that occur in many different documents are considered as less significant than terms that occur in a smaller portion of the documents.

By combining these two statistics, `TfidfVectorizer` assigns each word in the document a score that reflects both its frequency in a document and its importance across the entire dataset.

To improve the effectiveness of my vectorization, I decided to extend the list of stop words. Stop words are commonly used words (such as "the", "a", "an", "in") that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. These words usually have no significant meaning and are removed to increase the processing speed and improve the relevance of the results.

[illegible]

## Implementing the Extended TfidfVectorizer

Through these steps, I prepared textual data for machine learning, ensuring that models would be analyzing the most relevant features to predict sentiments accurately.

## 4.1 Basic information of Machine Learning models

In the context of evaluating a machine learning model, several metrics are used to measure its performance.

- 7

- **Recall Score:** Recall measures the ability of the model to find all the relevant cases within a dataset. For example, in a binary classification, it's the number of true positives divided by the number of true positives plus the number of false negatives.
- **F1 Score:** The F1 score is the harmonic mean of precision and recall, providing a balance between them. It's particularly useful when the class distribution is uneven.
- **Cohen's Kappa Score:** This score measures the agreement between two raters who each classify items into mutually exclusive categories, corrected for chance agreement.
- **ROC Curve:** The Receiver Operating Characteristic curve plots the true positive rate against the false positive rate at various threshold settings.
- **ROC AUC Score** The Area Under the ROC Curve measures the ability of the model to discriminate between positive and negative classes. An AUC of 1.0 represents a perfect model, while an AUC of 0.5 represents a model that performs no better than random guessing.

Finally, `classification_report` provides a summary of the precision, recall, F1 score, and support for each class. It gives a detailed view of the performance of the classification model, making it easier to understand where the model is doing ill and where it needs improvement.

## 4.2 K-Nearest Neighbors

The k-Nearest Neighbors (kNN) model is an algorithm used in machine learning for both classification and regression tasks, although it is more commonly associated with classification. KNN makes predictions about the classification of a sample based on the categories of its nearest neighbors in the feature space.

### How kNN Works:

- **Distance Measurement:** kNN calculates the distance between the data point in question and all other points in the dataset.
- **Identify Nearest Neighbors:** It identifies the 'k' closest points (neighbors) to the data point that is classified.
- **Majority Vote or Averaging:** For classification, the algorithm assigns the class based on the most common class among the nearest neighbors.

### Advantages of kNN:

- **Simplicity:** kNN is straightforward to understand and implement. It does not require complex mathematical assumptions or model training.
- **Versatility:** It can be used for both classification and regression problems.
- **Adaptability:** It can handle multi-class cases and is relatively effective in scenarios where the decision boundary is very irregular.

### Disadvantages of kNN:

- **Scalability:** kNN can be computationally expensive as the size of the dataset grows, because it requires calculating the distance of each query instance to all training samples.
- **High Memory Requirement:** the algorithm stores all of the training data, memory usage can be significant for large datasets.
- **Sensitivity to Irrelevant Features:** kNN is sensitive to irrelevant or redundant features because all features contribute equally to the distance computation.



- **Sensitivity to Imbalanced Data:** If one class dominates, it can influence the outcome more due to its sheer number, leading to biased predictions.
- **Choosing the Right 'k':** Selecting the optimal value of 'k' is crucial for the performance of the algorithm. A value too low can make the algorithm sensitive to noise, whereas a value too high makes it computationally expensive and possibly less accurate.

In summary, the kNN algorithm is easy to implement and can be highly effective for certain problems, but its performance and practicality can be limited by factors like data size, dimensionality, and feature relevance.

### 4.3 Logistic Regression

Logistic Regression is a statistical method for predicting binary outcomes from data. It estimates probabilities using a logistic function, which is an S-shaped curve that can take any real-valued number and map it between 0 and 1, but not exactly at those limits.

#### How Logistic Regression Works:

- **Probability Estimation:** It models the probability that a given input point belongs to a certain class.
- **Binary Classification:** While it can be extended to multiclass classification, logistic regression is binary and aims to categorize data into one of two groups.

#### Advantages of Logistic Regression:

- **Simplicity and Interpretability:** Logistic regression is straightforward to understand and interpret.
- **Efficiency:** It is computationally less intensive than more complex algorithms, making it relatively fast for training and prediction.
- **Probabilistic Results:** It provides probabilities for outcomes, which can be a crucial advantage when decisions need to be made on the basis of certainty.
- **Good Performance with Small Datasets:** It can perform well when the dataset is not too large and the relationship between the variables is approximately linear.

#### Disadvantages of Logistic Regression:

- **Assumption of Linearity:** It assumes a linear relationship between the independent variables, which is not always the case.
- **Performance on Non-linear Problems:** It may not perform well with complex relationships between variables or where there are non-linear interactions.
- **Not Suitable for Large Number of Categorical Features:** Logistic regression can become unstable when dealing with categorical variables that have a large number of levels.
- **Sensitivity to Imbalanced Data:** Like many other classification algorithms, logistic regression can perform poorly if there is a significant imbalance in the observations across the binary outcomes.

In summary, logistic regression is a model for binary classification problems with an emphasis on probability and odds. However, its performance can be limited by non-linear data relationships and assumptions about the data distribution.

### 4.4 Random Forest

Random Forest is a learning method used for both classification and regression tasks, though it is more commonly associated with classification. It operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes of the individual trees. Random forests correct for decision trees' habit of overfitting to their training set.

## How Random Forest Works:

- **Ensemble of Decision Trees:** Random Forest combines multiple decision trees to form a more robust prediction model. Each tree is trained on a random subset of the data and makes its own predictions.
- **Majority Voting :** For classification tasks, the prediction by the forest is the one that receives the majority votes from the trees.

## Advantages of Random Forest:

- **High Accuracy:** Random Forest can achieve high accuracy in many tasks, including those with complex datasets that have nonlinear relationships.
- **Robustness to Overfitting:** Due to the ensemble approach, it is less prone to overfitting compared to individual decision trees.

## Disadvantages of Random Forest:


- **Model Complexity:** Random Forest models can be quite complex and require more computational resources and time to train compared to simpler models like logistic regression.
- **Memory Consumption:** The large number of trees can lead to significant memory consumption.
- **Predictive Latency:** Making predictions can be slow if the forest is very large.

In summary, Random Forest is machine learning method that offers high accuracy and robustness to overfitting through its ensemble approach.

## 5 Results


I trained three models using the `X_train` and `y_train` subsets, below the outcomes from the classification reports are presented :

### 1. k-Nearest Neighbors




	precision	recall	f1-score	support
0	0.62	0.01	0.02	2996
1	0.86	1.00	0.93	18948
accuracy			0.86	21944
macro avg	0.74	0.50	0.47	21944
weighted avg	0.83	0.86	0.80	21944

### 2. Logistic Regression



	precision	recall	f1-score	support
0	0.80	0.48	0.60	2996
1	0.92	0.98	0.95	18948
accuracy			0.91	21944
macro avg	0.86	0.73	0.78	21944
weighted avg	0.91	0.91	0.90	21944

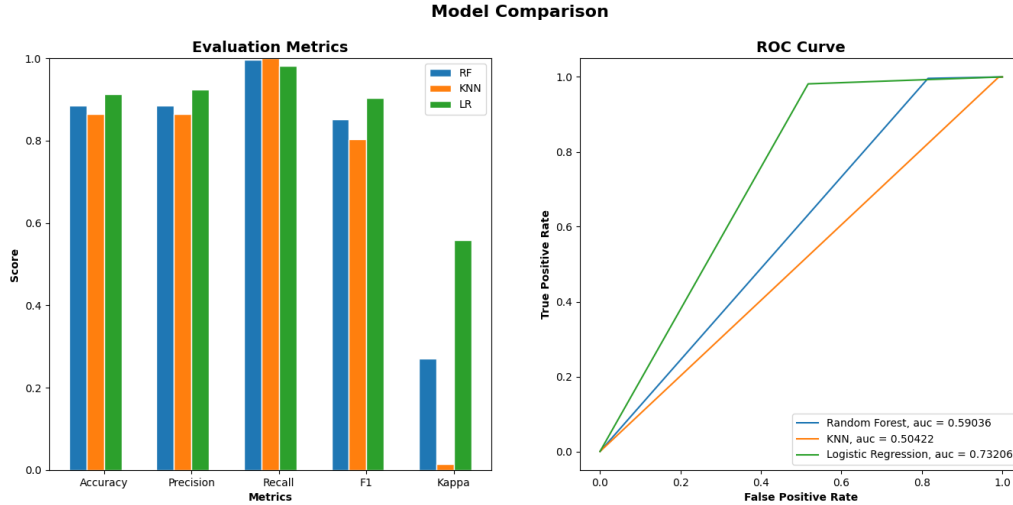
### 3. Random Forest



	precision	recall	f1-score	support
0	0.88	0.18	0.31	2996
1	0.89	1.00	0.94	18948
accuracy			0.89	21944
macro avg	0.88	0.59	0.62	21944
weighted avg	0.89	0.89	0.85	21944

## Comparison of performance

To facilitate a more straightforward comparison of the performance metrics across the three models, I visualized the results in a graph. This visual representation allows for an immediate and clear understanding of how each model scores on various evaluation metrics.



Based on the provided graph, I can see that:

- Logistic Regression has the highest AUC (Area Under the ROC Curve) score, indicating it has the best model performance in terms of distinguishing between the classes.
- In terms of Accuracy, Precision, and Recall, all three models perform well, but Logistic Regression shows a slight better scores.
- The F1 Score, which balances Precision and Recall, is also highest for Logistic Regression, suggesting that it has a balanced performance regarding false positives and false negatives.
- The lower Kappa score for the kNN model could be attributed to its sensitivity to noisy data and imbalance in the dataset, which may affect its ability to make accurate predictions.

For the amazon\_review dataset analyzed, Logistic Regression demonstrated the highest scores across several key performance metrics. These results suggest that Logistic Regression is good at handling the nuances of this dataset, providing mostly correct classification outcomes.

Therefore, in the context of this study, I conclude that Logistic Regression is the appropriate model to chose for sentiment analysis tasks.

## 6 Conclusion

In conclusion, this report has provided process of preparing, analyzing, and modeling a dataset of Amazon reviews for sentiment analysis. I began with a detailed exploration and cleaning of the data, ensuring that it was of high quality and suitable for machine learning applications. During this phase, I addressed issues such as missing values, irrelevant features, and non-English records, which refined dataset to a more manageable subset for this task.

I then proceeded to transform the textual data using TfidfVectorizer. I also extended my stop words list with the aid of a WordCloud visualization. This alloedd us to focus analysis on the most meaningful words within the reviews.

Following the preparation of data, I split my dataset into training and testing sets and trained three different machine learning models: k-Nearest Neighbors (kNN), Logistic Regression, and Random Forest. Each model was evaluated based on a range of metrics, including accuracy, precision, recall, F1 score, and the area under the ROC curve (AUC).

The results from the classification reports and ROC curve analysis indicated that Logistic Regression outperformed the other models in my specific scenario. It showed the highest AUC score, implying a superior

ability to differentiate between positive and negative sentiments. The Logistic Regression model also presented the most balanced results across all metrics, making it the most reasonable choice for this sentiment analysis task.

I can assert that the choice of model in sentiment analysis is crucial and highly dependent on the specific characteristics of the dataset at hand. While Logistic Regression was the standout model in this instance, it is essential to consider the context and demands of each unique dataset when selecting a model.

## References

- [1] Amazon reviews. <https://www.kaggle.com/datasets/angusemmett/amazon-reviews>.