Sentiment Analysis of Amazon Sales Review Using Multinomial Naive Bayes Method

Danica Alana Sjurjahady

About Me

Hello everyone! My name is Danica Alana Sjurjahady.
I'm a fresh graduate majoring in Agro-Industrial Technology from University of Darussalam Gontor, who have a strong interest in data science and data analysis.
Have a good understanding of basic concepts of statistics and machine learning. Skilled in using MySQL, Python, Google Looker Studio, Google Colab, and Microsoft Power BI. Ready to learn and grow in the role of Data Analyst.

Introduction

This dataset is having the data of 1K+ Amazon Product's Ratings and Reviews as per their details listed on the official website of Amazon.
You can access the dataset through this hyperlink:
Amazon Sales Dataset
This project is a sentiment analysis project using a machine learning model. It analyzes Amazon product reviews to determine whether the sentiment expressed is positive, negative, or neutral. The project uses a dataset of Amazon product reviews (amazon.csv) and applies a Multinomial Naive Bayes model for classification.

Objectives

- Developing a model that can accurately classify the sentiment of Amazon product reviews.
- Cleaning and preparing the data for modeling, including handling missing values, removing duplicates, and converting ratings to sentiment labels.
- Using TF-IDF (Term Frequency-Inverse Document Frequency) to convert text reviews into numerical representations for the model.
- Assessing the accuracy and performance of the trained model using metrics such as accuracy and classification report.

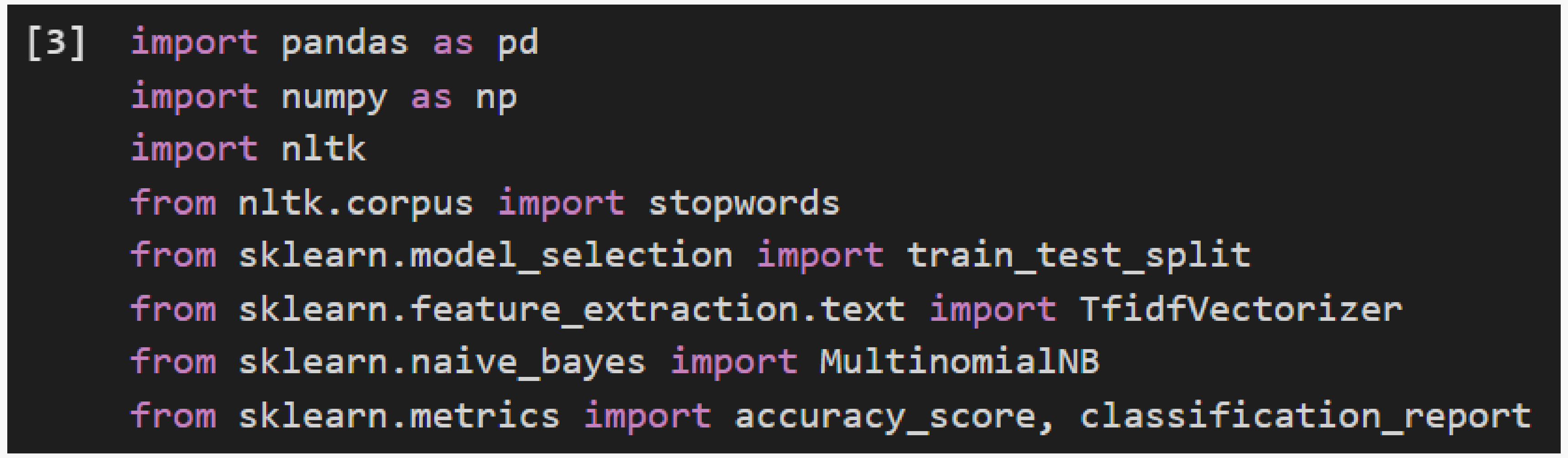
Tools Used

NLTK

pandas

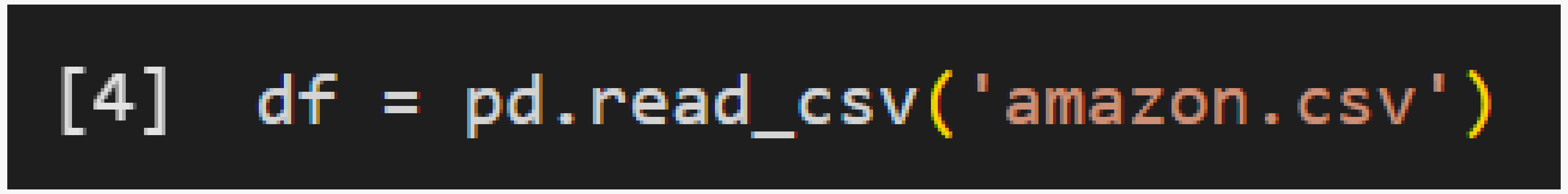
scikit
learn

Library Preparation and Data Loading



```
[3] import pandas as pd
    import numpy as np
    import nltk
    from nltk.corpus import stopwords
    from sklearn.model_selection import train_test_split
    from sklearn.feature_extraction.text import TfidfVectorizer
    from sklearn.naive_bayes import MultinomialNB
    from sklearn.metrics import accuracy_score, classification_report
```

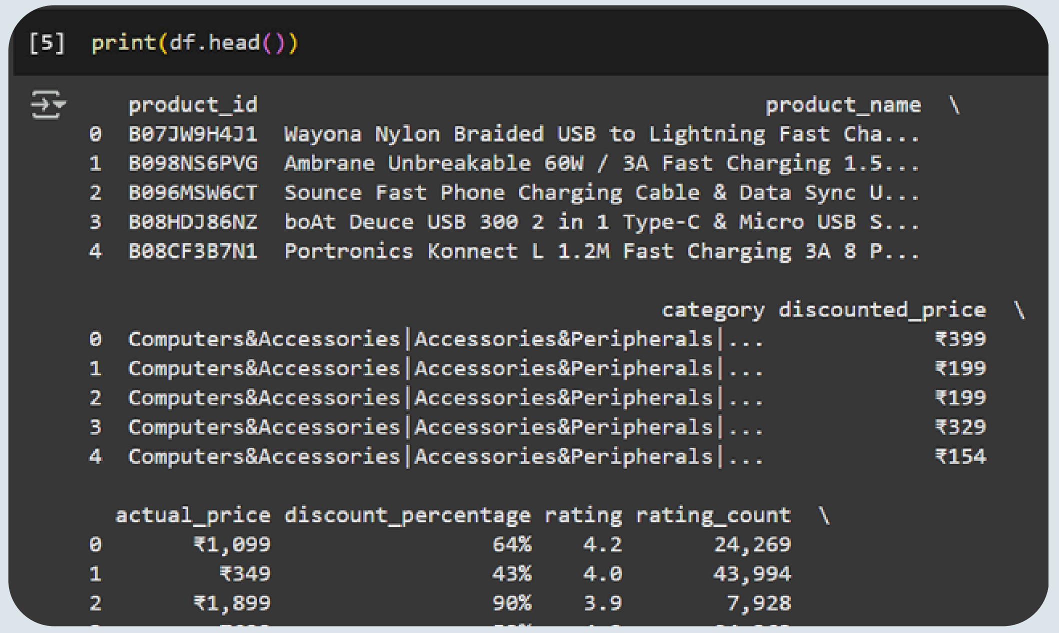
Before starting to process data, it is necessary to prepare the necessary libraries, such as pandas (used for data manipulation and analysis), NumPy (used for numerical computing in Python), NLTK (used for natural language processing in Python), and scikit-learn (used for machine learning and statistical modeling).



```
[4] df = pd.read_csv('amazon.csv')
```

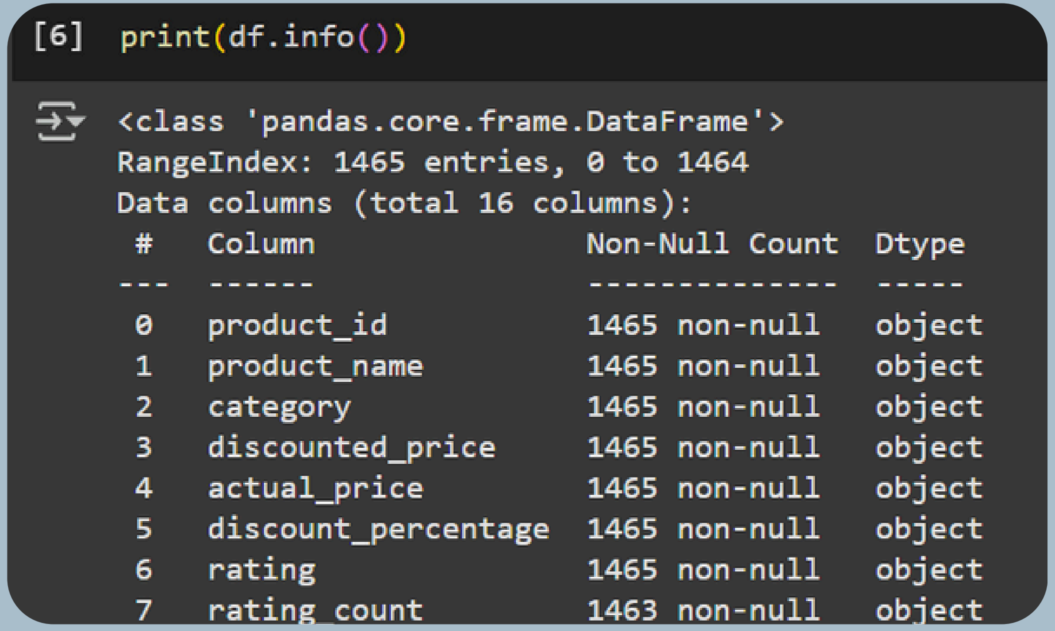
This line of code reads the data from the 'amazon.csv' file and stores it in a pandas DataFrame called df, allowing to work with the data within the Python code.

Exploratory Data Analysis (EDA)



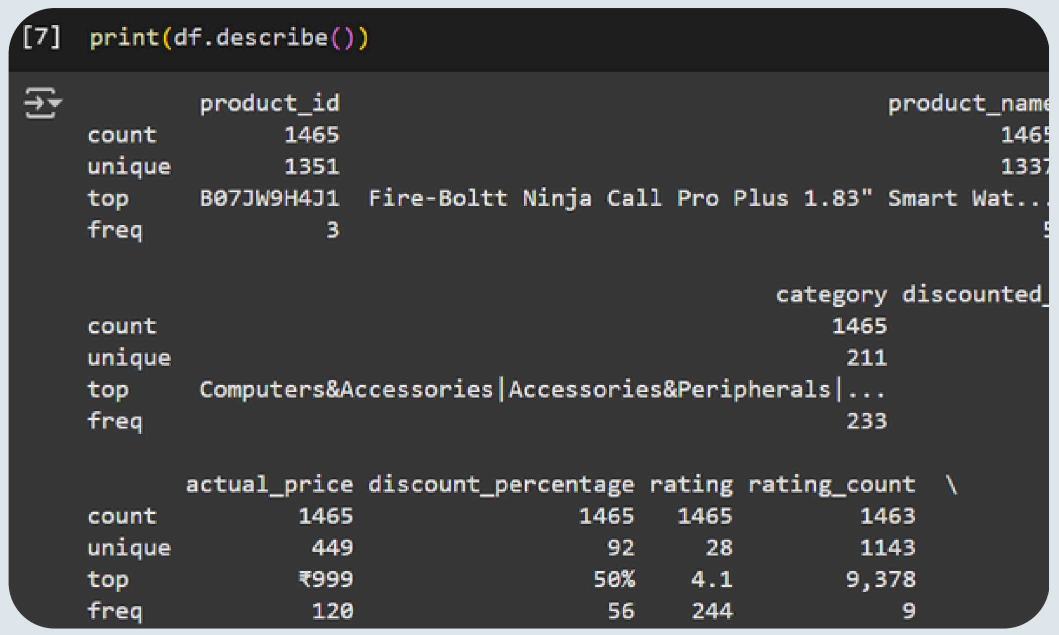
```
[5]  print(df.head())

      product_id                                       product_name  \
   0  B07JW9H4J1  Wayona Nylon Braided USB to Lightning Fast Cha...
   1  B098NS6PVG  Ambrane Unbreakable 60W / 3A Fast Charging 1.5...
   2  B096MSW6CT  Sounce Fast Phone Charging Cable & Data Sync U...
   3  B08HDJ86NZ  boAt Deuce USB 300 2 in 1 Type-C & Micro USB S...
   4  B08CF3B7N1  Portronics Konnect L 1.2M Fast Charging 3A 8 P...

                                              category discounted_price  \
   0  Computers&Accessories|Accessories&Peripherals|...             ₹399
   1  Computers&Accessories|Accessories&Peripherals|...             ₹199
   2  Computers&Accessories|Accessories&Peripherals|...             ₹199
   3  Computers&Accessories|Accessories&Peripherals|...             ₹329
   4  Computers&Accessories|Accessories&Peripherals|...             ₹154

     actual_price discount_percentage rating rating_count  \
   0       ₹1,099                 64%    4.2       24,269
   1         ₹349                 43%    4.0       43,994
   2       ₹1,899                 90%    3.9        7,928
```

This line of code helps the user getting a quick preview of the data you're working with, by taking the data table df, selecting its first 5 rows, and then showing those rows on the screen.



```
[6]  print(df.info())

     <class 'pandas.core.frame.DataFrame'>
     RangeIndex: 1465 entries, 0 to 1464
     Data columns (total 16 columns):
      #   Column               Non-Null Count  Dtype
     ---  ------               --------------  -----
      0   product_id           1465 non-null   object
      1   product_name         1465 non-null   object
      2   category             1465 non-null   object
      3   discounted_price     1465 non-null   object
      4   actual_price         1465 non-null   object
      5   discount_percentage  1465 non-null   object
      6   rating               1465 non-null   object
      7   rating_count         1463 non-null   object
```

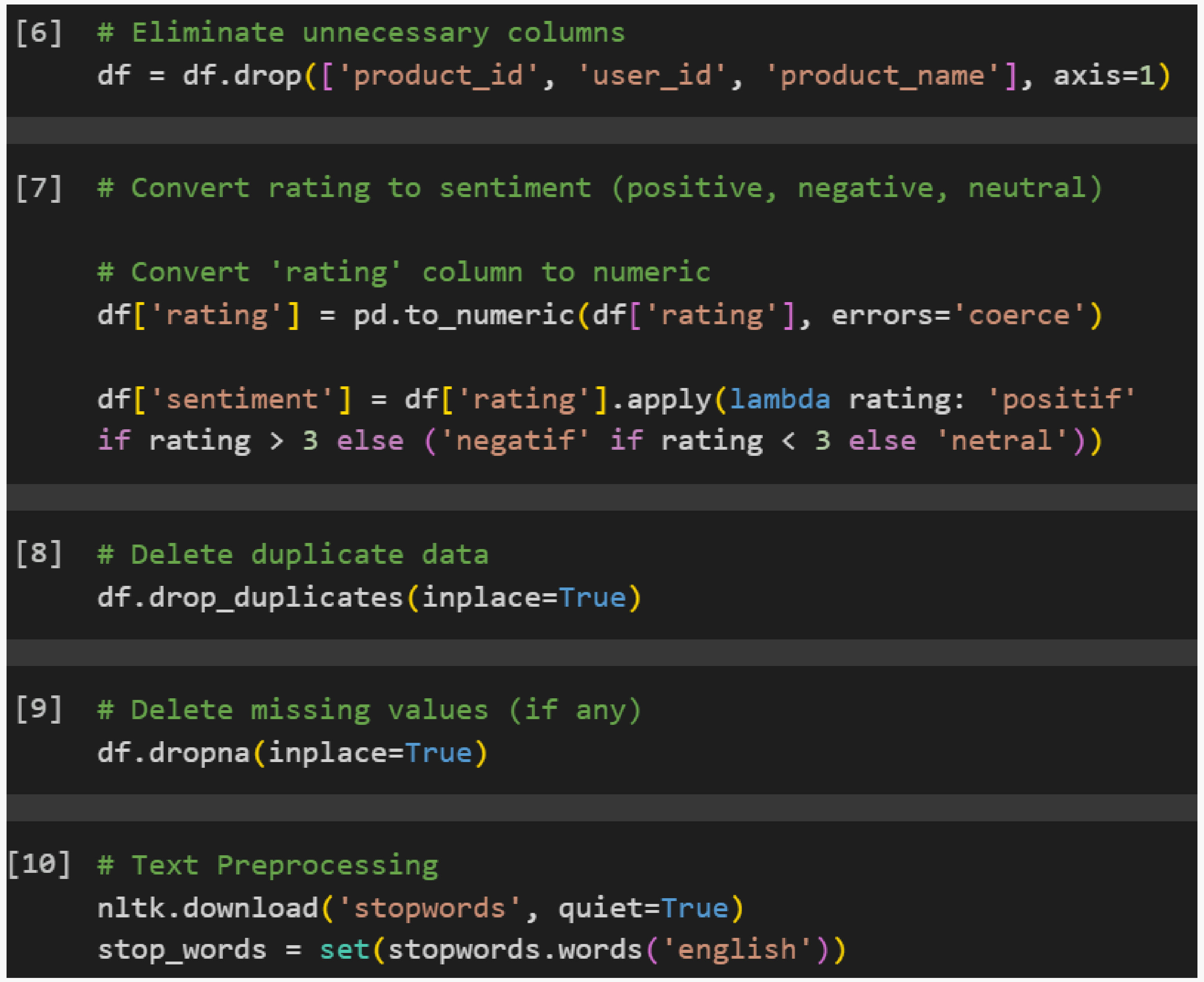
This line of code will display several things, such as total number of rows and columns in DataFrame, etc. This crucial step provides a quick overview, allowing to understand the type of data and spot potential issues like missing values.



```
[7]  print(df.describe())

           product_id                                      product_nam
   count         1465                                              146
   unique        1351                                              133
   top     B07JW9H4J1  Fire-Boltt Ninja Call Pro Plus 1.83" Smart Wat..

   freq             3


                                                    category discounted_
   count                                                1465
   unique                                                211
   top     Computers&Accessories|Accessories&Peripherals|...
   freq                                                  233

          actual_price discount_percentage rating rating_count  \
   count          1465                1465   1465         1463
   unique          449                  92     28         1143
   top            ₹999                 50%    4.1        9,378
   freq            120                  56    244            9
```

This line of code asks for a summary of the numerical data. It will tell the average rating, the highest price, the lowest price, and other useful statistical information about the data.

Data Preprocessing



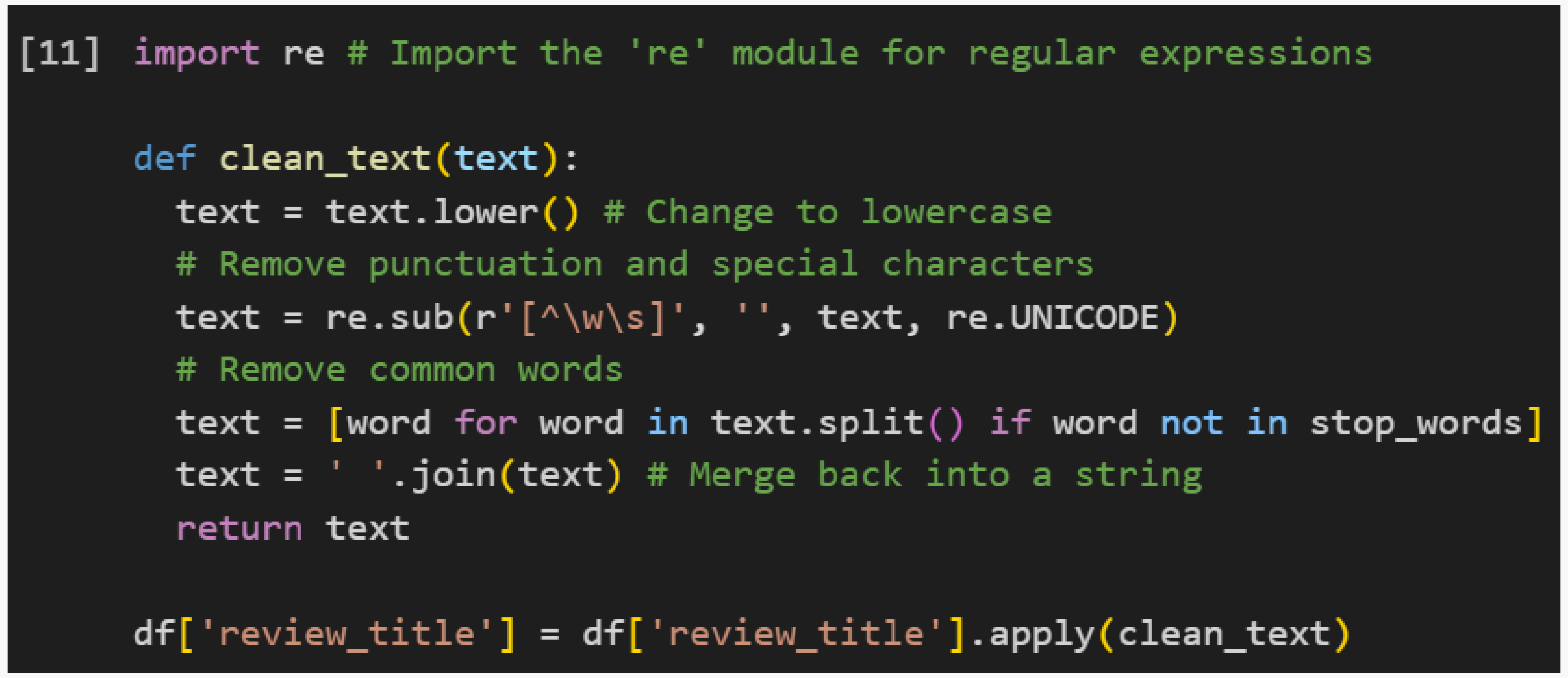
```
[6]  # Eliminate unnecessary columns
     df = df.drop(['product_id', 'user_id', 'product_name'], axis=1)
```

```
[7]  # Convert rating to sentiment (positive, negative, neutral)

     # Convert 'rating' column to numeric
     df['rating'] = pd.to_numeric(df['rating'], errors='coerce')

     df['sentiment'] = df['rating'].apply(lambda rating: 'positif'
     if rating > 3 else ('negatif' if rating < 3 else 'netral'))
```

```
[8]  # Delete duplicate data
     df.drop_duplicates(inplace=True)
```

```
[9]  # Delete missing values (if any)
     df.dropna(inplace=True)
```

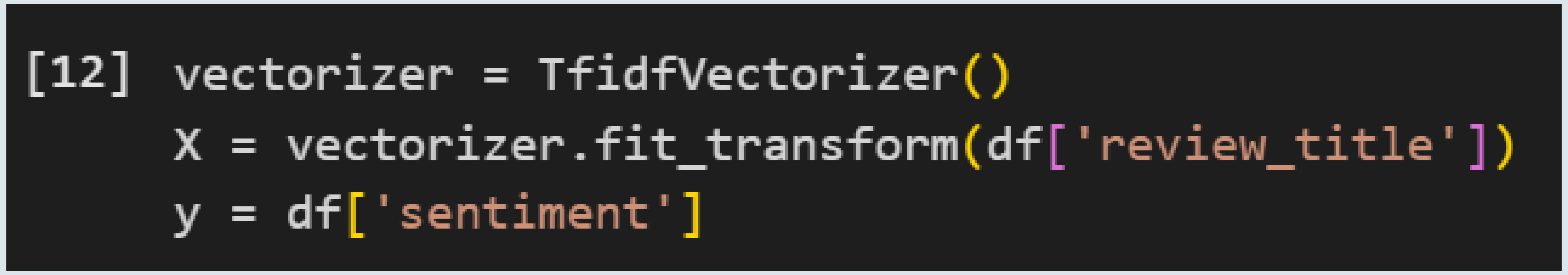
```
[10] # Text Preprocessing
     nltk.download('stopwords', quiet=True)
     stop_words = set(stopwords.words('english'))
```



```
[11] import re # Import the 're' module for regular expressions

     def clean_text(text):
       text = text.lower() # Change to lowercase
       # Remove punctuation and special characters
       text = re.sub(r'[^\w\s]', '', text, re.UNICODE)
       # Remove common words
       text = [word for word in text.split() if word not in stop_words]
       text = ' '.join(text) # Merge back into a string
       return text

     df['review_title'] = df['review_title'].apply(clean_text)
```

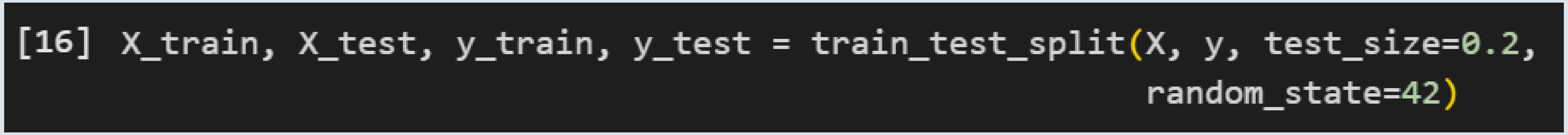
These lines of code aim to clean and prepare the book review text data for sentiment analysis. In short, these codes prepare book review data by cleaning irrelevant data, converting ratings to sentiments, removing duplicate and empty data, and cleaning the review text so that it is ready for further analysis.

Feature Extraction & Data Splitting



```
[12] vectorizer = TfidfVectorizer()
     X = vectorizer.fit_transform(df['review_title'])
     y = df['sentiment']
```

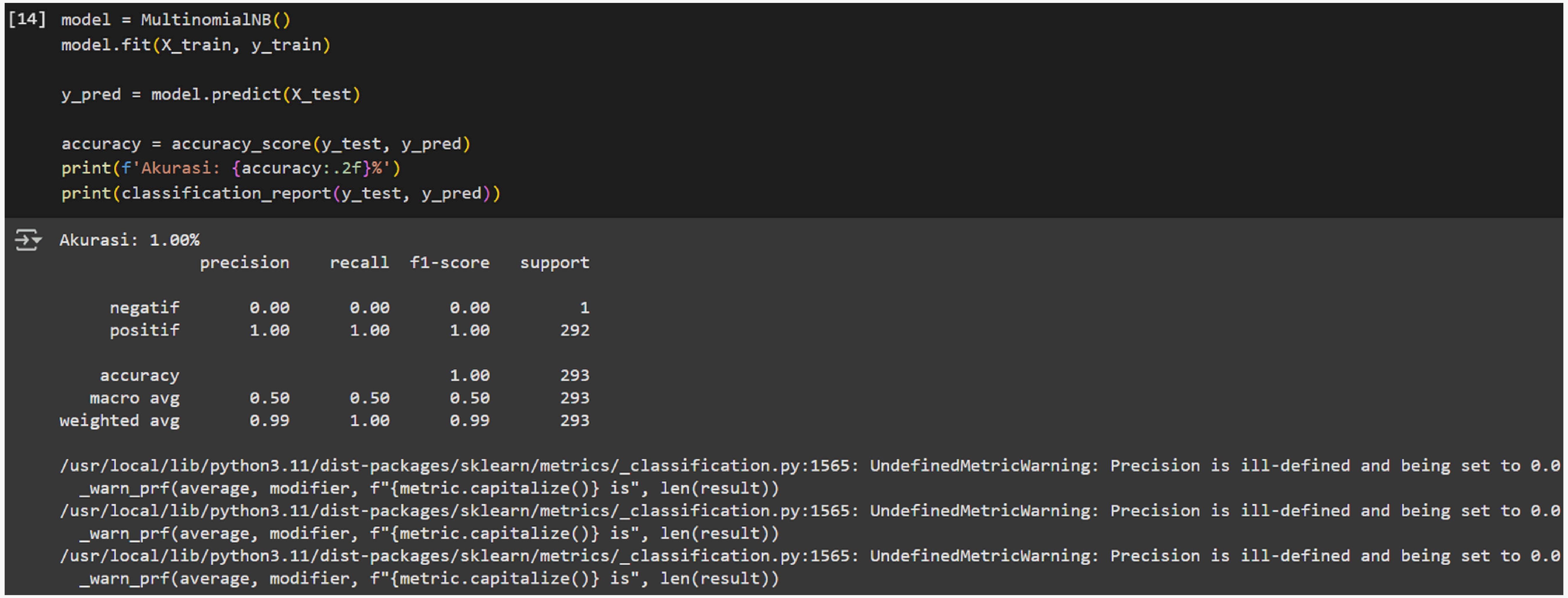
These lines of code take the text data (review titles), transform them into a numerical representation using TF-IDF, and create the input (X) and output (y) variables needed for training a machine learning model to predict sentiment.



```
[16] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                         random_state=42)
```

This line of code prepares data for machine learning by splitting it into training and testing sets, ensuring a consistent and reproducible split for model evaluation.

Modelling & Evaluating

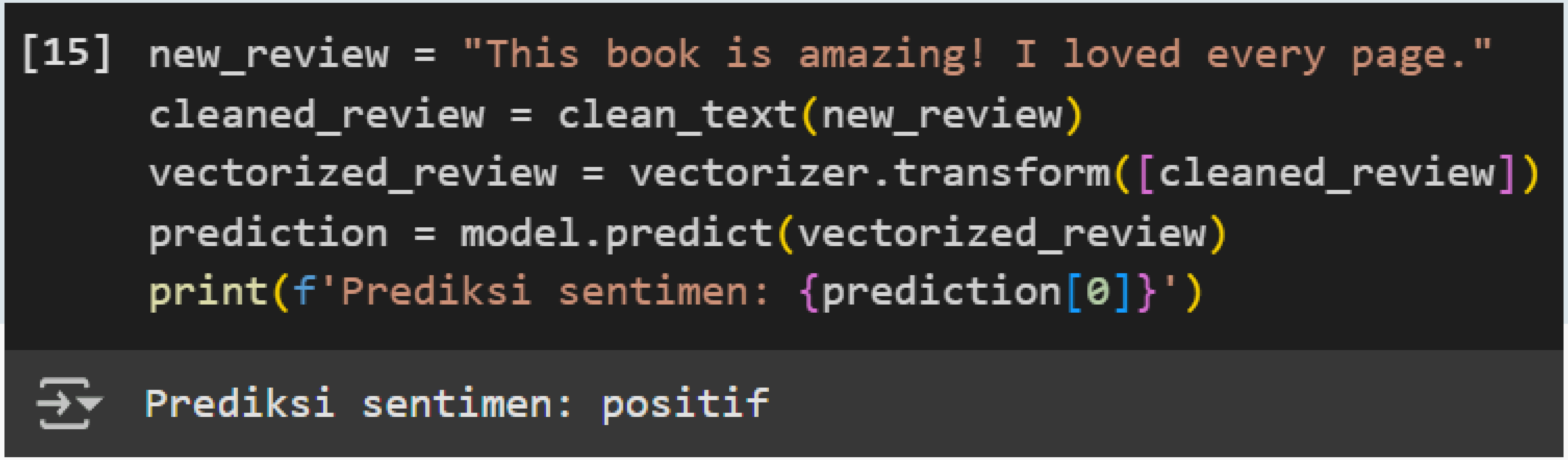


```
[14] model = MultinomialNB()
     model.fit(X_train, y_train)

     y_pred = model.predict(X_test)

     accuracy = accuracy_score(y_test, y_pred)
     print(f'Akurasi: {accuracy:.2f}%')
     print(classification_report(y_test, y_pred))
```

```
Akurasi: 1.00%
              precision    recall  f1-score   support

     negatif       0.00      0.00      0.00         1
     positif       1.00      1.00      1.00       292

    accuracy                           1.00       293
   macro avg       0.50      0.50      0.50       293
weighted avg       0.99      1.00      0.99       293

/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

These lines of code train a Naive Bayes model for sentiment classification, then test its ability to predict sentiment on new data and measure how well the model performs using accuracy and other metrics.

Model Usage Example



```
[15] new_review = "This book is amazing! I loved every page."
     cleaned_review = clean_text(new_review)
     vectorized_review = vectorizer.transform([cleaned_review])
     prediction = model.predict(vectorized_review)
     print(f'Prediksi sentimen: {prediction[0]}')
```

```
Prediksi sentimen: positif
```

These lines of code take a new review, clean it up, convert it into a numerical format that the model can understand, use the model to predict the sentiment of the review, and then display the results.

Conclusion

The project successfully built a sentiment analysis model for Amazon product reviews using a Multinomial Naive Bayes classifier. The model achieved a certain level of accuracy in classifying reviews as positive, negative, or neutral.
The project highlights the importance of data preprocessing and feature extraction in achieving accurate sentiment analysis results, with 100% accuracy. Future improvements could involve exploring other models or using more advanced techniques for feature engineering.

Thank you!

If you have any questions, suggestions or feedbacks, please do not hesitate to reach me through these contacts:

sjurjahady29@gmail.com

github.com/DanicaAlana

linkedin.com/in/danicaas