Dataset Description:
Contains 34,660 Amazon product reviews for different products. The entry includes fields like review text, rating, sentiment, reviewer name, review date, etc.
Ratings are on a scale of 1 to 5

Preprocessing Steps:

- Dropped unwanted columns and kept only relevant ones; review text, and rating, and cleaned review text.
- Tokenised the review text using spaCy
- Counted the most common words in the reviews to help identify words within the stop-word list that are relevant for sentiment analysis.
- Applied sentiment analysis to the cleaned review text using TextBlob
- Mapped the ratings to sentiment labels (positive, neutral, negative)
- Saved the preprocessed data to a new CSV file

Sentiment Analysis with TextBlob

Evaluation of Results:

- Compared the sentiment labels obtained from TextBlob with the sentiment labels derived from ratings
- Created a confusion matrix to visualize the performance of the sentiment analysis
- The majority of reviews were classified as positive by both TextBlob and rating-based sentiment
- There were some mismatches between TextBlob and rating-based sentiment labels

Model's Strengths:

- Provides a quick and automated way to analyze sentiment of a large number of reviews
- Utilizes both review text and ratings to derive sentiment
- Preprocessing steps help clean and prepare the data for analysis
- Confusion matrix gives a good overview of the model's performance

Model's Limitations:

- Relies on a lexicon-based approach (TextBlob) which may not capture context and nuances in the reviews
- May struggle with sarcasm, idioms, or domain-specific language
- Doesn't handle negation or complex sentence structures very well
- Requires manual mapping of ratings to sentiment labels
- Needs more robust evaluation metrics beyond just a confusion matrix

Sentiment Analysis using vectors:

Evaluation of Results:

- Used spaCy to create document vectors for each review
- Scaled the document vectors using MinMaxScaler
- Trained a Multinomial Naive Bayes classifier on the scaled document vectors

- Evaluated the classifier using a classification report
- The classifier achieved high accuracy (93%) but performed poorly on negative and neutral classes
- Also trained a KNN classifier which showed slightly improved performance on negative and neutral classes

Model's Strengths:

- Uses document vectors created by spaCy which capture semantic information
- Scales the document vectors to bring them into a similar range
- Evaluates the models using a classification report which provides metrics like precision, recall, and F1-score
- Tries multiple classifiers (Multinomial Naive Bayes and KNN) to compare performance

Model's Limitations:

- Still struggles with accurately classifying negative and neutral sentiment, despite using document vectors
- The performance is heavily skewed towards the positive class, likely due to class imbalance in the dataset
- Doesn't utilize cross-validation or a separate validation set for model evaluation

In summary, the sentiment_model notebook builds upon the preprocessing steps from the first notebook and tries to train machine learning models for sentiment classification. While it achieves high overall accuracy, it struggles with the negative and neutral classes, likely due to class imbalance.