Names: Aya Hajjeh and Bilan Aden
Kaggle usernames: BilanAdenn & ayahajjeh
Team Name: Aya & Bilan
Multiclass Classification of Amazon Product Ratings Project

Data:

- Describe the dataset you worked with, including explanations of interesting feature variables and the target variable related to this task. Highlight if and how you have changed the feature engineering process in this task from the binary classification task.
    - Description of the dataset
        - The dataset contains various information about amazon reviews across a variety of products and product categories. It includes various features such as overall, verified, review time, reviewer id, asin, reviewer name, review text, summary, unix review time, vote, image, style, category, and id of the sample in the data.
    - Some of the interesting features were
        - Verified
            - This denotes if a review has been verified by Amazon as being legitimate. Verified reviews should have a higher weight than non-verified reviews when processing the data.
        - Review text
            - The textual content of the body of the review. This astronomically influences the overall rating of the product. Positive reviews indicate a higher-rated product, and vice versa.
        - Summary
            - A high-level summary of the review. This also astronomically influences the overall rating of the product. Positive review summaries indicate a higher-rated product, and vice versa.
        - Category
            - High-level product category (e.g. Toys, Automotive). It can be relevant to distinguish between different categories of products since reviewers might leave reviews of makeup products in a different tone than they might describe automotive products.
    - The target variable
        - The target variable is the overall rating of the products in the data set. That is, whether a product has a label 0, 1, 2, 3, 4, or, 5
    - Any changes in the feature engineering process in this task from the binary classification task?
        - The only difference is that in the binary classification model we had to convert the overall column into numbers of 0s and 1s only according to our definition of low-star and high-star ratings. In this classification problem, we didn't touch the overall column at all.

Methods:

- Describe the steps you took to build your model.
    - Steps for classifier #1
        - First we imported the necessary libraries to manipulate the data the way we want it. Then the train and test data were loaded into their perspective DataFrame. The focus here was the reviewText and summary columns and these became the features. Secondly, we split the data using train_test_split to split the training data into actual training data and a validation data set. Third, we calibrated the Perceptron Model. Then we fitted the model and made sure to vectorize the training data. Next, we set up the evaluation metrics to calculate such as confusion matrix, accuracy, f1 score and ROC AUC score based on the real and pred values.
    - Steps for classifier #2 and #3
        - First I dropped all the columns that I won't be working with. In the training data set, that would be all the columns except for the overall, text review, and summary columns. For the testing data set, that would be all the columns except for the review text and the summary columns. After dropping the columns I won't be working with, I combined both of the review text and summary columns into one column by joining the strings in each row into one single column instead of two.
        - After having the two text columns merged into one single text column, I tokenized and lemmatized the text to make the data more consistent. I tried out some models where I deleted stop words, but for those models, deleting the stop words only resulted in minor changes in the scores of the corresponding columns.
        - After that, I used the TfidfVectorizer function to vectorize the text data so that it's ready to be used by the different classification models that I used, that is, Logistic Regression, Perceptron, and Random Forest Classifier.
        - After having the text data and the target label ready to be used by the different classifiers, first I chose what classifier I wanted to experiment with. I experimented with three different classifiers: Logistic Regression, Perceptron, and Random Forest Classifier. Then, for each classifier, I used the GridSearchCV function to find the best hyperparameters for each classifier through 5-fold cross validation.
- Describe the hyperparameters you used in your best model. Describe how you did your hyperparameter search. What are the things that you did differently than the binary classification task?
    - For binary classification tasks, hyperparameter tuning might focus on metrics like precision, recall, and F1 score. In multi-class classification, additional techniques like one-vs-rest or one-vs-one strategies can be employed for models that inherently support binary classification, like Perceptron or SVM. In a hyperparameter search for multi-class tasks, the approach might involve performing cross-validation with multiple parameter combinations, optimizing for a specific metric like macro/micro F1-score or weighted average accuracy across all classes.

- In our implementation of this project we used the GridSearchCV function to find optimal parameters through 5-fold cross validation and a scoring method based on f1 macro score and accuracy.
- The hyperparameters we used in our best model were {'C': 3.0, 'max_iter': 2000} in the logistic regression model.

Results:

- Show the confusion matrix, ROC score, macro F1 score, and accuracy for the best set of hyperparameters using 5-fold cross-validation for each of your three models.

Confusion matrices: Performance table:

- Perceptron:

[[844 214 78 48 49]

[310 459 222 92 65]

[125 249 505 221 95]

[66 118 176 507 262]

[50 53 81 160 789]]

- Logistic Regression:

[[870 255 60 28 20]

[275 580 203 63 27]

[92 229 623 211 40]

[36 65 150 667 211]

[37 29 37 184 846]]

- Random Forest Classifier:

[[834 283 46 26 44]

[296 594 139 70 49]

[119 297 545 173 61]

[74 170 146 577 162]

[77 93 57 169 737]]

Model	ROC_AUC_Score	Macro F1	Accuracy
Model 1 (Perceptron)	0.9554761098731867	0.5257938906193264	0.5316889345666324
Model 2 (Logistic Regression)	0.8889343769875788	0.6128614873290748	0.6142514559780747
Model 3 (Random Forest Classifier)	0.8889343769875788	0.5635655270686144	0.5630352860568688

- List the score you achieved using Kaggle
    - 0.62339
- What is a suitable metric for this task?
    - It seems like accuracy and F1 macro score are suitable metrics for this task. Accuray provides a simple and intuitive measure of overall model performance. In terms of F1 macro score, it is a great evaluation metric for multi-class classification problems because it handles class imbalances on top of other reasons.
- Does your best model provide the best performance across all metrics? If so, why?
    - No, in this case, our best model, that is logistic regression, has the worst roc auc score, and our worst model, that is perceptron, has the highest roc auc score.
- Which class(es) are more challenging to predict? Can you reason why they are harder to predict? (note I consulted chatgpt 3.5 for this question)
    - Labels in the middle such as 2 and 3 might be harder to predict as they may represent more neutral sentiments. Distinguishing between moderately positive and moderately negative reviews could require more subtle analysis, and the features that differentiate between these classes might be less pronounced in the reviews of these products.

Future Work

- Describe potential future work: what would you do differently next time?
    - For this project, we focused primarily on text features of the data set. As helpful and relevant text features are, we might in the future try to include other features that can improve our classification that are not necessarily textual features such as the verified, category, and review time features especially because customer opinion changes over

time and things like product quality can only be determined over time especially when it's a product people want to keep for a long time.

- It would also be helpful to add a range of prices for products when the review was given and that could lead to a better classification.