# **CS74 Final Project – Amazon Review Analysis**

**Author**: Max Fortner  
 **Dartmouth ID**: F006TYR

## **Background**

Analyzing Amazon product reviews typically include textual feedback, star ratings, and other features such as verification status, timestamps, and product categories. Using these features, we can train multiple types of models to predict things about the reviews:

1. **Classifying** reviews into positive or negative using different cutoff definitions of “positive” (binary classification).
2. **Classifying** product ratings on a 5-star scale (multiclass classification).
3. **Clustering** reviews to discover underlying structure in the data (e.g., grouping reviews by product category).

These tasks demonstrate core principles from **COSC 74/274** (Machine Learning and Statistical Data Analysis), including data preprocessing, feature extraction, hyperparameter tuning, and evaluation metrics.

## **Project Objectives**

* **Binary Classification** with cutoffs = 1, 2, 3, 4.
  + We create binary labels by comparing the rating to a chosen cutoff.
* **Multiclass Classification** from 1 to 5 stars.
* **Clustering** of the test dataset reviews by product category (using K-Means).

These tasks apply multiple machine learning models, using 5-fold cross-validation, and evaluate performance using standard metrics (confusion matrix, ROC, AUC, macro F1, accuracy, silhouette score, and adjusted rand index).

## 

## **Dataset**

### **Data Description**

We used the **Amazon Review dataset**, which comes as two main CSV files:

1. **Training.csv**: Includes product reviews, star ratings (1–5), verification status, review text, timestamps, helpfulness votes, product category, etc.
2. **Test.csv**: Contains the same features except the star ratings are withheld.

Key fields used in our analysis include:

* **overall**: Star rating (1–5).
* **verified**: Boolean indicating if the review is verified.
* **reviewText**: The textual review.
* **vote**: Number of helpful votes.
* **unixReviewTime**: Unix timestamp for the review.
* **category**: Product category.

### **Data Preprocessing**

* **Missing Values**: We filled missing numeric columns such as vote with 0 and unixReviewTime with the median value (as needed).
* **Text Cleaning**: We primarily used raw text from reviewText. Minimal cleaning was done besides standard TF-IDF tokenization, lowercasing, etc.
* **Feature Engineering**:
  + **TF-IDF**: We converted the reviewText column into TF-IDF features with a chosen vocabulary size (e.g., max\_features=10000 or 3000 for the clustering part).
  + For clustering, we additionally used numeric columns: verified, vote, unixReviewTime and scaled them via **StandardScaler**.

## **Binary Classification**

We performed **four separate binary classification tasks**, each based on a different cutoff:

1. **Cutoff = 1**: Label = 1 if rating > 1, else 0.
2. **Cutoff = 2**: Label = 1 if rating > 2, else 0.
3. **Cutoff = 3**: Label = 1 if rating > 3, else 0.
4. **Cutoff = 4**: Label = 1 if rating > 4, else 0.

### **Label Definition**

For each cutoff, any review with a rating **≤ cutoff** is labeled as 0 (negative), and reviews with a rating **> cutoff** are labeled as 1 (positive).

### **Models Explored**

We tried at least three models for each cutoff:

1. **Logistic Regression** (with variations in C, solver, class\_weight, etc.)
2. **Linear SVM** (LinearSVC from scikit-learn; tuned C, loss, class\_weight)
3. **Multinomial Naive Bayes** (tuned alpha)

### **Hyperparameter Tuning**

* **5-Fold Cross-Validation**: For each model, we tested multiple hyperparameter combinations:
  + **Logistic Regression**: C=[0.1, 1.0, 10.0], class\_weight=[None, 'balanced'], solver=['liblinear','saga']
  + **Linear SVM**: C=[0.1, 1.0, 10.0], class\_weight=[None, 'balanced'], loss=['hinge','squared\_hinge']
  + **Naive Bayes**: alpha=[0.01, 0.1, 1.0, 10.0]
* For each combination, we computed the mean **macro F1** score over 5 folds and selected the best.

### **Best Model Selection**

* We chose the best hyperparameters based on the highest mean F1 from cross-validation.
* **Example**: For Logistic Regression, if C=1.0, class\_weight=None, solver='liblinear' gave the highest F1, we selected that as the final combination.

*(A similar process was repeated for* ***cutoff=1, 2, 3, and 4****. Specific numeric results or tables are included in the Results section)*

## **Multiclass Classification**

### **Problem Definition**

Here, we classify reviews into one of **5 classes** (ratings 1, 2, 3, 4, or 5).

### **Models**

1. **Logistic Regression** (multinomial solver)
2. **Linear SVM** (handling multiple classes via one-vs-rest or one-vs-one internally)
3. **Multinomial Naive Bayes**
4. **Random Forest**

### **Hyperparameter Tuning**

* Again, used **5-Fold Cross-Validation** across parameter grids:
  + **Logistic Regression**: e.g., C=[0.1, 1.0, 10.0], class\_weight=[None, 'balanced'], multi\_class='multinomial'.
  + **Linear SVM**: C=[0.1, 1.0, 10.0], class\_weight=[None, 'balanced'].
  + **Naive Bayes**: alpha=[0.01, 0.1, 1.0, 10.0].
  + **Random Forest**: n\_estimators, max\_depth, class\_weight, etc.

### **Best Model Selection**

* We picked the model and parameter combination yielding the highest **macro F1** from cross-validation.
* The chosen model’s confusion matrix, ROC curves, and final metrics are reported in **Results**.

## **Clustering**

### **Objective**

We applied **K-Means** on the **Test.csv** set (since its ratings are not revealed). We treat **product category** as ground truth labels to measure the **Adjusted Rand Index** (ARI).

### **Feature Extraction**

* **TF-IDF** on reviewText with a maximum of ~3000 features.
* Numeric features: **verified** (converted to int), **vote**, **unixReviewTime**.
* After combining text features and numeric features, we optionally did **TruncatedSVD** (to 100 components) for dimensionality reduction.

### **Number of Clusters**

* We varied **k** from 2 to 10 and calculated the **Silhouette Score** for each.
* The **best k** is the one with the highest silhouette score.

## **Results and Analysis**

### **Binary Classification Results**

**Cutoff 1:**

Best Model: Logistic Regression

Best parameters: {'C': 1.0, 'class\_weight': 'balanced', 'solver': 'saga'}

Best cross-validation F1 (macro): 0.7468

Accuracy: 0.7982

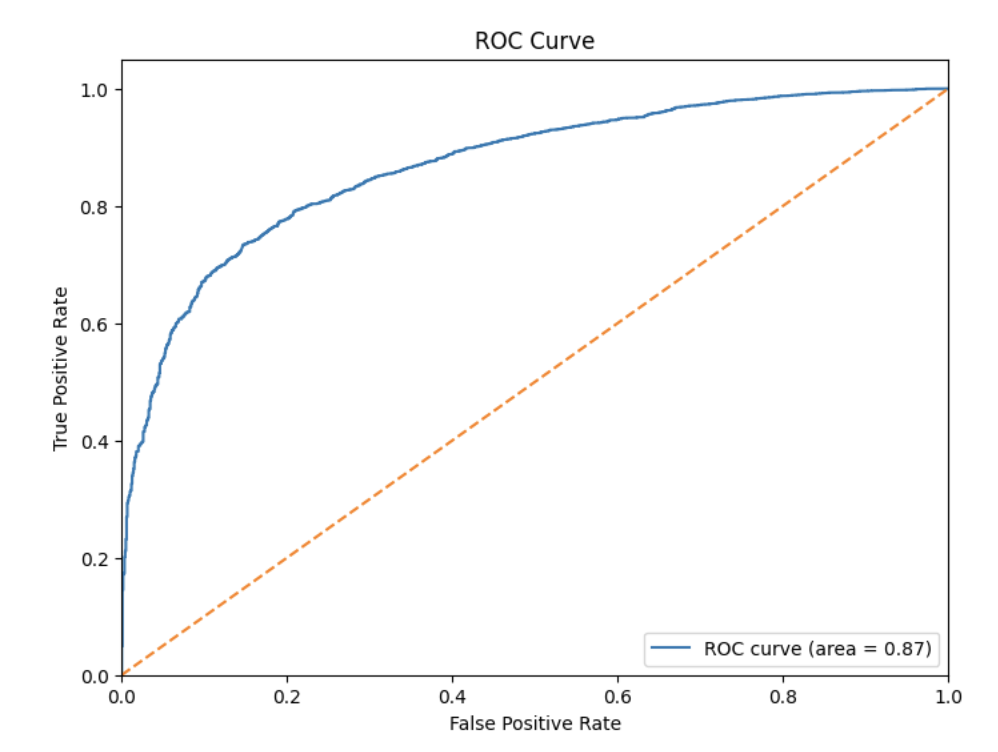
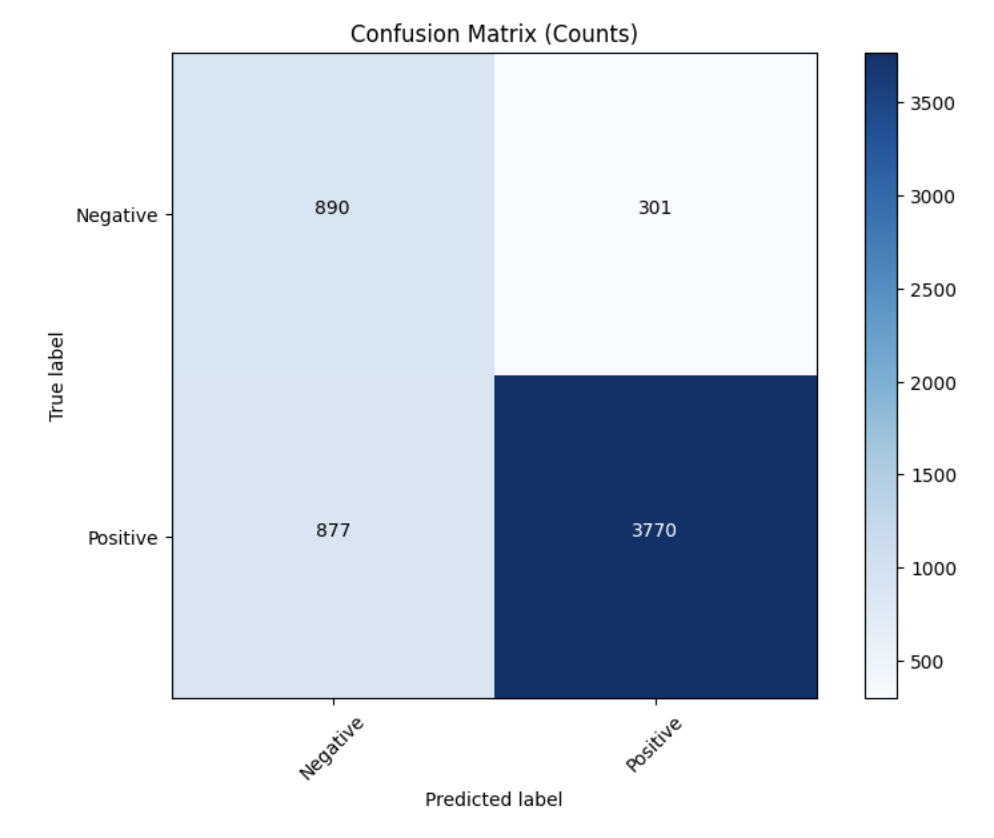
Macro F1 Score: 0.7333

Classification Report:

precision recall f1-score support

0 0.50 0.75 0.60 1191

1 0.93 0.81 0.86 4647



**Kaggle Score: 0.74061**

**Cutoff 2:**

Best Model: Linear SVM

Best parameters: {'C': 0.1, 'class\_weight': 'balanced', 'loss': 'squared\_hinge'}

Best cross-validation F1 (macro): 0.8024

Accuracy: 0.8129

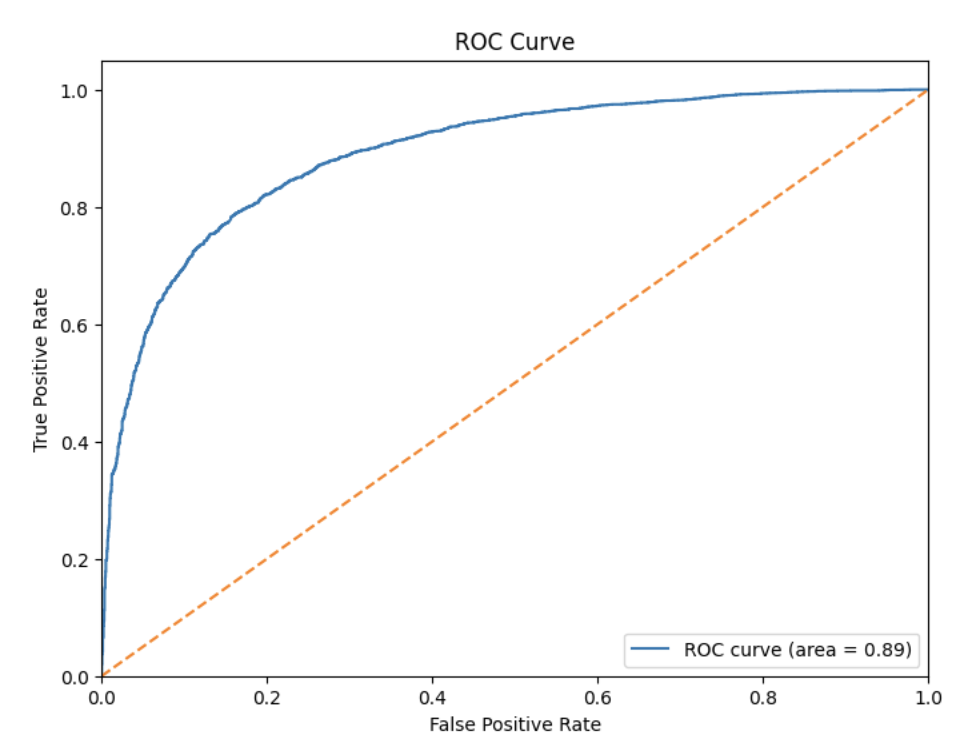
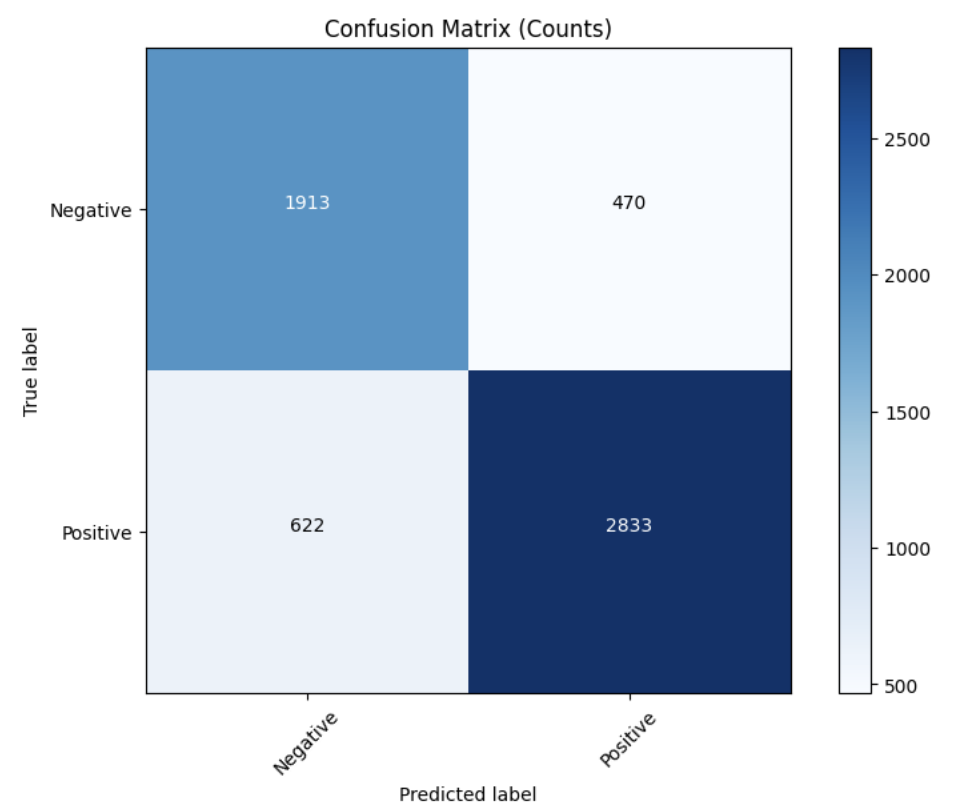
Macro F1 Score: 0.8082

Classification Report:

precision recall f1-score support

0 0.75 0.80 0.78 2383

1 0.86 0.82 0.84 3455



**Kaggle Score: 0.80083**

**Cutoff 3:**

Best Model: Logistic Regression

Best parameters: {'C': 1.0, 'class\_weight': 'balanced', 'solver': 'liblinear'}

Best cross-validation F1 (macro): 0.8152

Accuracy: 0.8359

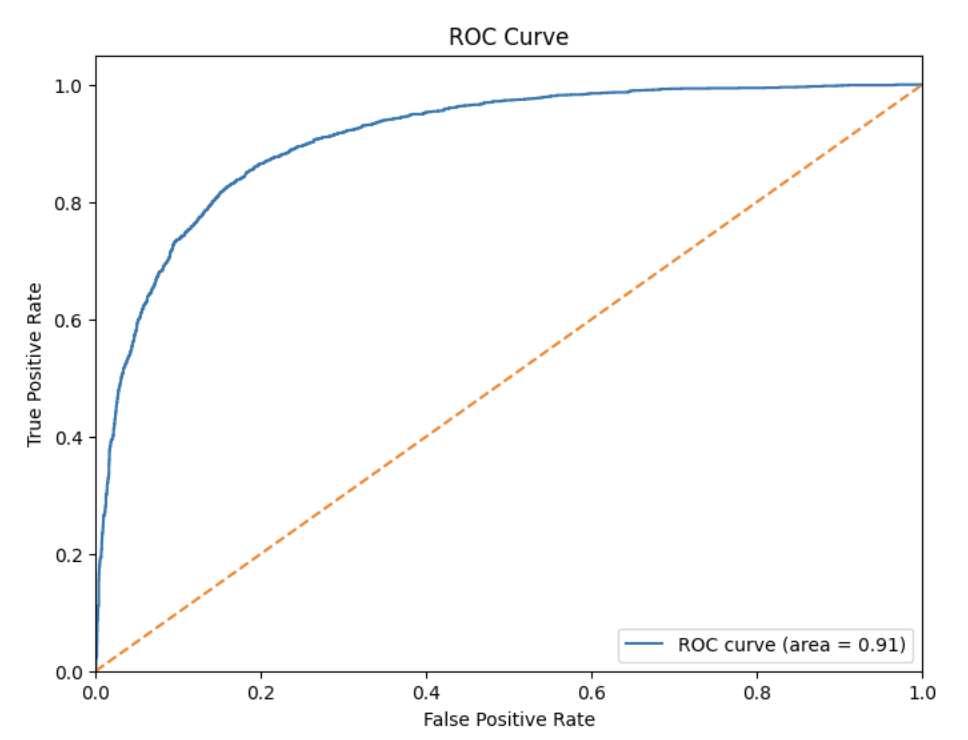
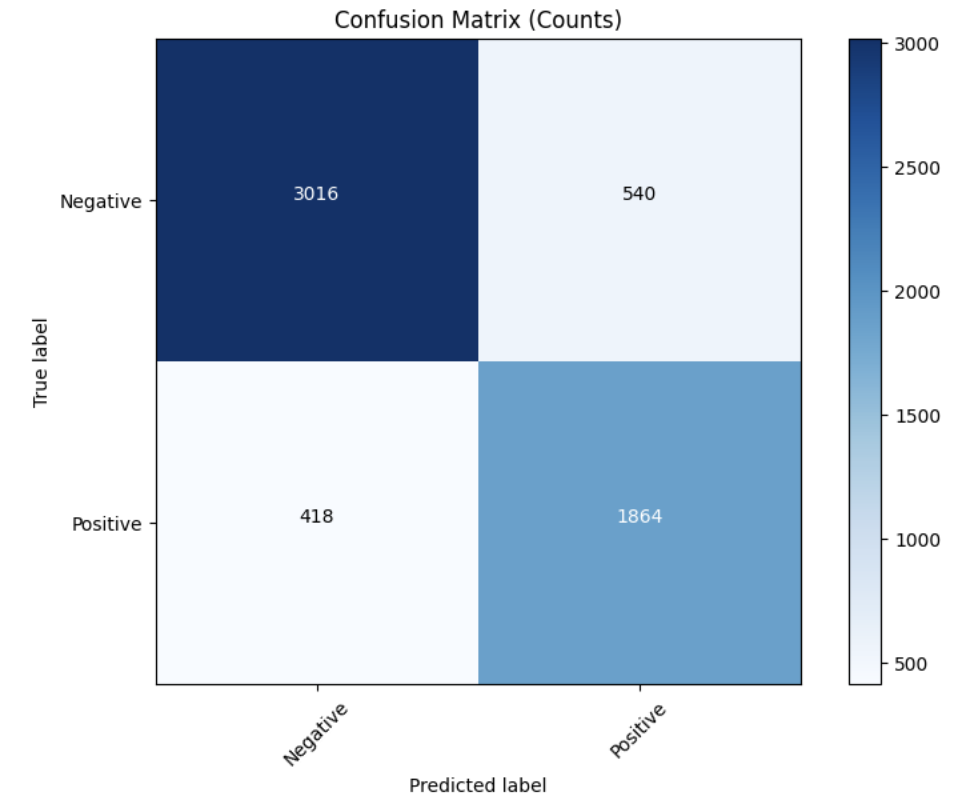
Macro F1 Score: 0.8293

Classification Report:

precision recall f1-score support

0 0.88 0.85 0.86 3556

1 0.78 0.82 0.80 2282

****

**Kaggle Score: 0.82202**

**Cutoff 4:**

Best Model: Logistic Regression

Best parameters: {'C': 1.0, 'class\_weight': 'balanced', 'solver': 'liblinear'}

Best cross-validation F1 (macro): 0.7685

Accuracy: 0.8338

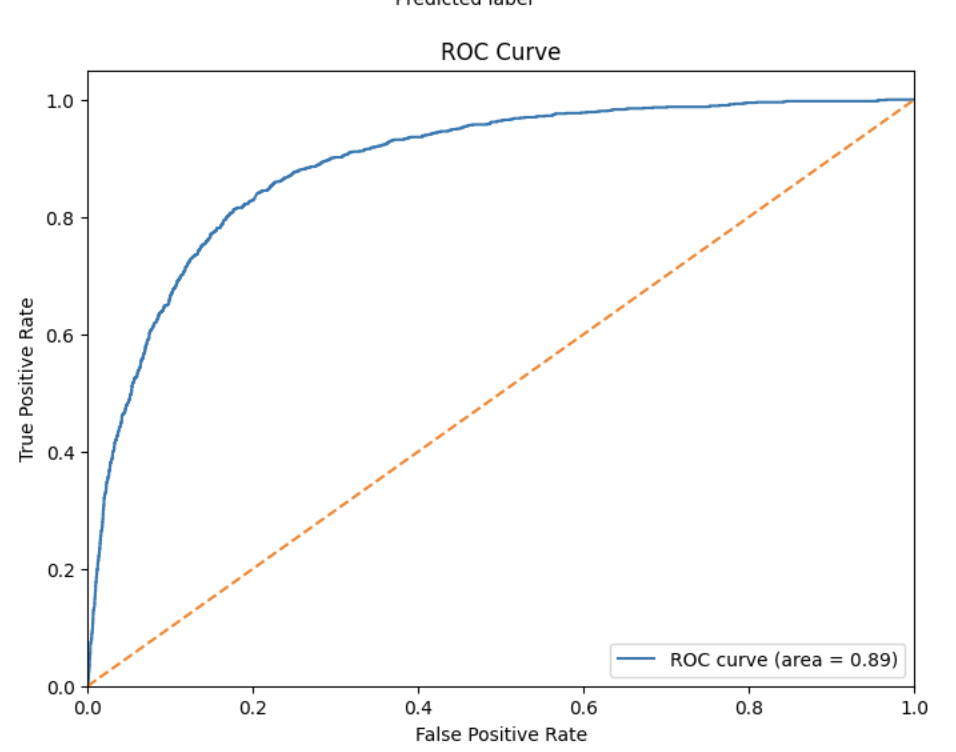
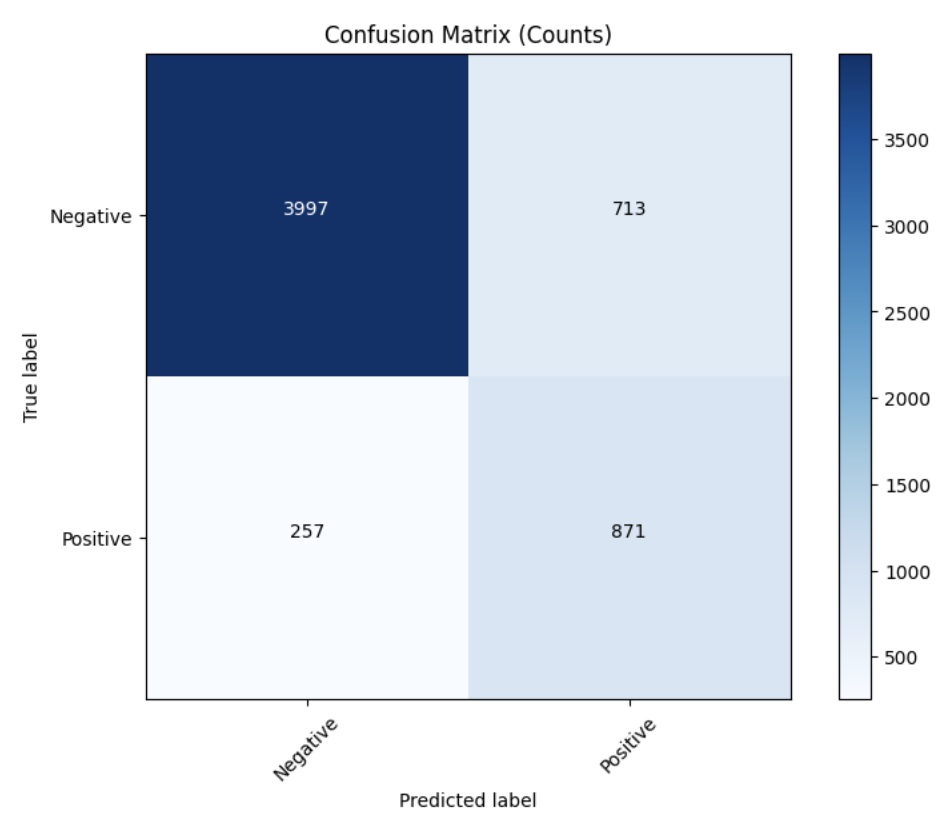
Macro F1 Score: 0.7671

Classification Report:

precision recall f1-score support

0 0.94 0.85 0.89 4710

1 0.55 0.77 0.64 1128



**Kaggle Score: 0.76273**

**Multiclass Classification Results**

Best Model: Logistic Regression

Best parameters: {'C': 1.0, 'class\_weight': 'balanced'}

Best cross-validation F1 (macro): 0.4841

Accuracy: 0.4976

Macro F1 Score: 0.4955

Classification Report:

precision recall f1-score support

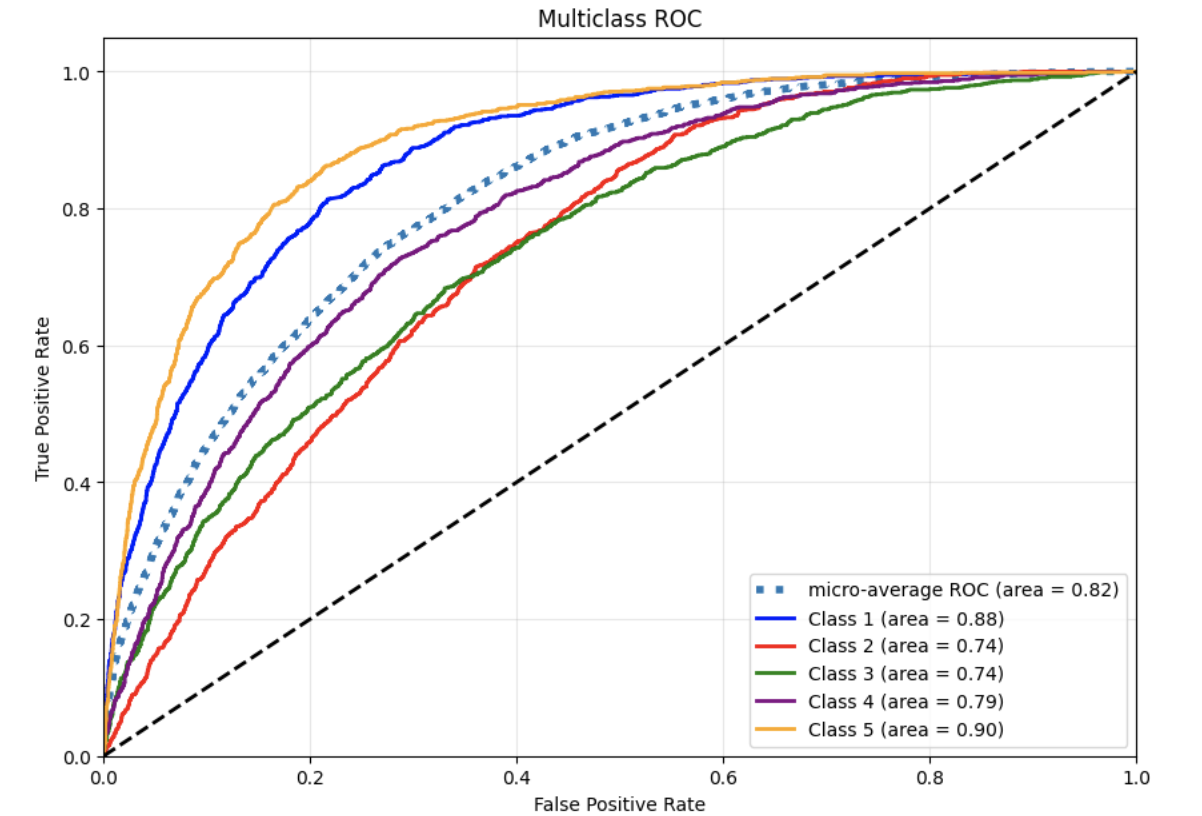
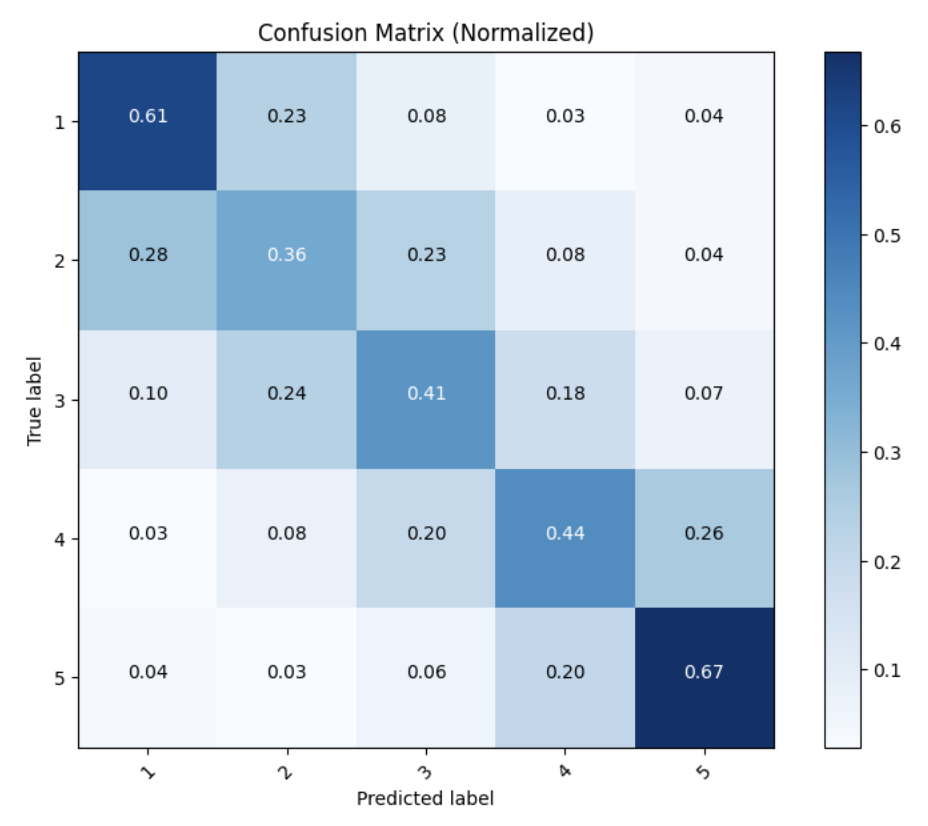
1 0.58 0.61 0.60 1192

2 0.39 0.36 0.37 1192

3 0.42 0.41 0.42 1172

4 0.47 0.44 0.45 1154

5 0.61 0.67 0.64 1128



### 

### **Clustering Results**

* **Optimal k**: 3
* **Silhouette Score**: 0.6160
* **Rand Index**: 0.0456
* **Observations**:
  + ARI is extremely low while silhouette score is high. Likely means that the clusters were well separated but did not reflect the actual categories well.

## **Discussion**

### **Summary of Key Findings**

* Logistic Regression was the best model for all classification experiments except for cutoff 2.
* All classifiers meet F1 baseline on test data
* Clustering model achieved baseline silhouette score but had very low ARI

### **Future Work**

* Future work should include revisiting the clustering model to improve ARI