NLP 220 Assignment 2

Ishika Kulkarni
ikulkar1@ucsc.edu

Abstract

This report addresses a multiclass classification problem on an E-commerce dataset that contains categories of items and their respective descriptions. We use classifiers like SVMs (Support Vector Machine), Logistic Regression, and Decision Trees for the same after extracting features from the dataset.

1 Introduction

This assignment aims to develop a machine learning model to classify E-commerce products into predefined categories. We explore several classification techniques, such as the One-vs-Rest (OvR) strategy. We use Bag of Words (BoW), Term Frequency - Inverse Document Frequency (TF-IDF), and n-grams for feature engineering.

As mentioned in the abstract, the classifiers used were SVMs, Logistic Regression, and Decision Trees. Grid Search CV was used on each of them to perform hyperparameter tuning, and it was evaluated based on metrics like precision, accuracy, macro F1, and time taken for prediction.

2 Dataset

Let's dive into the dataset. The E-commerce dataset contains two columns - Category and Description with 50423 rows.

	Category	Description
0	Household	SAF 'Floral' Framed Painting (Wood, 30 inch x ...
1	Household	SAF 'UV Textured Modern Art Print Framed' Pain...
2	Household	SAF Flower Print Framed Painting (Synthetic, 1...
3	Household	Incredible Gifts India Wooden Happy Birthday U...
4	Household	Pitaara Box Romantic Venice Canvas Painting 6m...

Figure 1: Dataset

The data types, as we can see, are non-null objects.

```
<class 'pandas.core.frame.DataFrame'>
Index: 50423 entries, 0 to 50423
Data columns (total 2 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Category     50423 non-null  object
 1   Description  50423 non-null  object
dtypes: object(2)
memory usage: 1.2+ MB
```

Figure 2: Dataset Info

Before going to the following steps, we split the data into Train, Validation, and Test with random states initialized as 42 and shuffling as accurate.

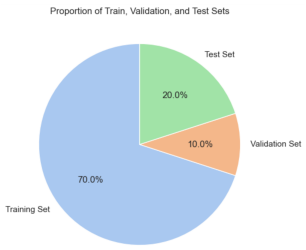

Figure 3: Dataset Info

3 Data Pre-Processing and Feature Engineering

We process the data before using it to remove noise like special characters, extra spaces, and random symbols.

```
6702     Okayji Cotton Heat Proof Microwave Oven Gloves...
12642    Philips HD Litre Electric Kettle Multicolor Pr...
18686    Bosch Hand Tool Kit Blue Pieces All the tools ...
19922    The Incredible History of Indias Geography Abo...
16169    Voltas Ton Star Window AC Copper CYA CZP White...
19602     Music Scales For Composers Improvising Musicians
17086    JP Hardware Brass Door Knob Pack of Suitable f...
9263     Kosh Cake Decorating ToolsNonStick Square and ...
34628                    Rupa Thermocot Mens Thermal Set
41368    Seagate TB Backup Plus Slim Red USB External H...
Name: Description, dtype: object
```

Figure 4: Processed Data

The distribution for categories across the sets is as follows:

4 Class Distribution

The distribution of categories across the training, validation, and test datasets is shown in the table below:

Category	Training Set	Validation Set	Test Set
Household	13,464	1,963	3,885
Books	8,337	1,145	2,338
Electronics	7,443	1,067	2,111
Clothing & Accessories	6,052	866	1,752

Table 1: Class distribution across the Training, Validation, and Test datasets.

The visual representation for the same is as follows:

Figure 5: Training Set

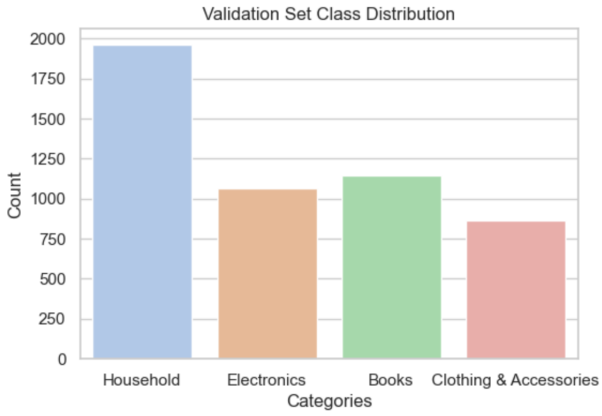

Figure 6: Validation Set

For feature engineering, three methods were used,

Figure 7: Test Set

- **Bag of Words (BoW)**: Transforms text into a matrix where each column represents a word in the vocabulary and each row corresponds to a document, with values representing word frequencies in that document.
- **Term Frequency-Inverse Document Frequency (TF-IDF)** [1]: Adjusts word frequency based on how common or rare the word is across all documents, giving higher weights to words unique to specific documents.
- **N-Grams (Bi-grams)**: Captures relationships between consecutive words (i.e., pairs of words) to provide context, which is useful for understanding phrases within the text.

5 Result for Feature and Model Evaluation

5.1 Feature-to-Feature Comparison

In this section, we compare how different feature sets (Bag of Words, TF-IDF, and N-grams) impact the performance of various machine learning models. We focus on crucial evaluation metrics such as accuracy, precision, macro F1 score, and model training time.

5.1.1 Key Observations

- **Bag of Words (BoW)**:
  - *Best Model*: Logistic Regression with BoW (Accuracy: 0.9747, Precision: 0.9747, F1 Score: 0.9744, Time: 28.16 seconds).
  - *SVM with BoW*: Performed well with an accuracy of 0.9729 (Precision: 0.9725,

[1] TF-IDF Documentation

Recall: 0.9730, F1 Score: 0.9727, Time: 6.28 seconds), indicating strong performance in terms of both speed and accuracy.

- **TF-IDF**:
  - *Best Model*: SVM with TF-IDF (Accuracy: 0.9787, Precision: 0.9794, F1 Score: 0.9786, Time: 1.46 seconds), demonstrating the highest performance across all models and feature sets.
  - The SVM with TF-IDF outperformed all other models in accuracy and precision while also being significantly faster than Decision Tree models.
  - *Logistic Regression with TF-IDF*: Showed good performance (Accuracy: 0.9663, Precision: 0.9687), but still fell short compared to the SVM with TF-IDF.

- **N-grams**:
  - *Best Model*: SVM with N-grams (Accuracy: 0.9534, Precision: 0.9545, F1 Score: 0.9525, Time: 31.68 seconds). This model showed decent performance but was slower than both BoW and TF-IDF approaches.
  - *Decision Tree with N-grams*: Had the lowest accuracy (0.9189) and the longest training time (121.98 seconds), indicating it was the least effective choice among the evaluated models and features.

5.1.2 Summary of Feature Comparison

Model	Best Feature Set	Accuracy	Precision	Macro F1	Time(s)
SVM	TF-IDF	0.9787	0.9794	0.9786	1.46
Logistic Regression	Bag of Words	0.9747	0.9747	0.9744	28.16
Decision Tree	Bag of Words	0.9485	0.9505	0.9491	19.89

Table 2: Model-to-Model Comparison: Performance across different models.

5.2 Model-to-Model Comparison

In this section, we compare the performance of three different models: **SVM**, **Logistic Regression**, and **Decision Tree**, across three feature sets (BoW, TF-IDF, and N-grams).

5.2.1 Key Observations

- **SVM**:
  - *Best Model*: SVM with TF-IDF (Accuracy: 0.9787, Precision: 0.9794, F1 Score: 0.9786, Time: 1.46 seconds). This model demonstrates superior performance in terms of both accuracy and speed, making it the most efficient choice.

- **Logistic Regression**:
  - *Best Model*: Logistic Regression with Bag of Words (Accuracy: 0.9747, Precision: 0.9747, F1 Score: 0.9744, Time: 28.16 seconds). This model performs well but is slower and less accurate compared to the SVM with TF-IDF.

- **Decision Tree**:
  - *Best Model*: Decision Tree with Bag of Words (Accuracy: 0.9485, Precision: 0.9505, F1 Score: 0.9491, Time: 19.89 seconds). Decision Trees showed the lowest accuracy among the models evaluated, highlighting their relative inefficiency for this task.

5.2.2 Summary of Model Comparison

Model	Best Feature Set	Accuracy	Precision	Macro F1	Time(s)
SVM	TF-IDF	0.9787	0.9794	0.9786	1.46
Logistic Regression	Bag of Words	0.9747	0.9747	0.9744	28.16
Decision Tree	Bag of Words	0.9485	0.9505	0.9491	19.89

Table 3: Model-to-Model Comparison: Performance across different models.

5.2.3 Confusion Matrices

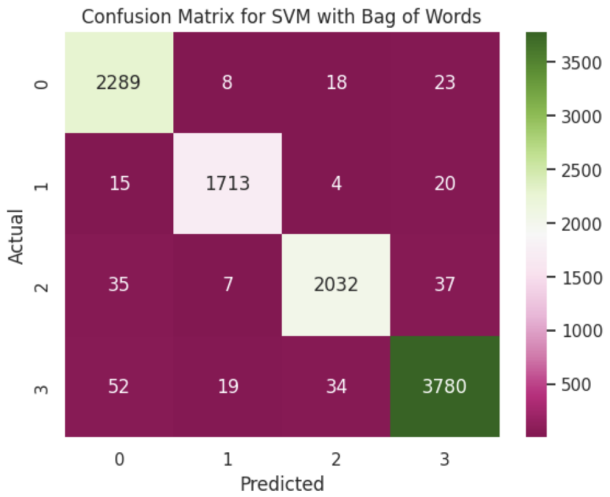

Figure 8: SVM-BoW

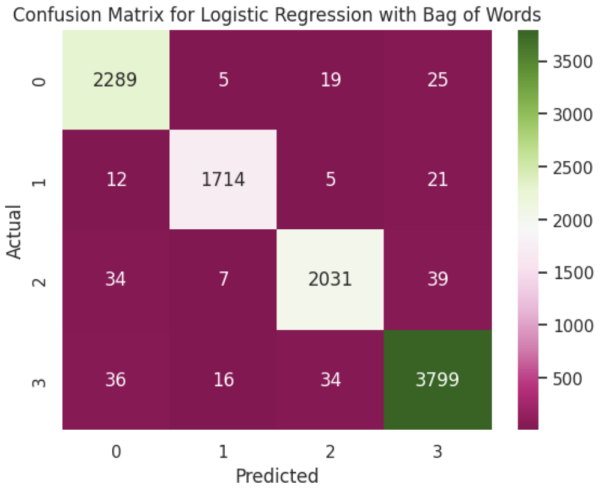

Figure 11: LR-BoW

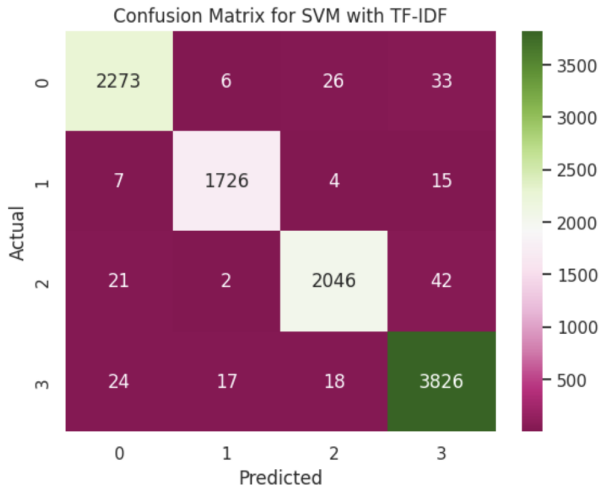

Figure 9: SVM-TFIDF

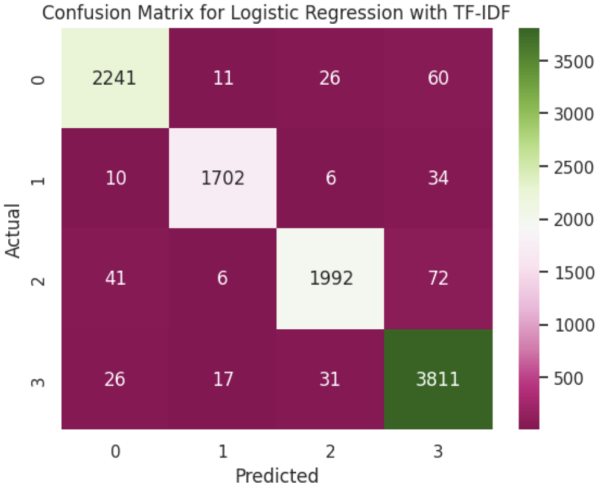

Figure 12: LR-TFIDF

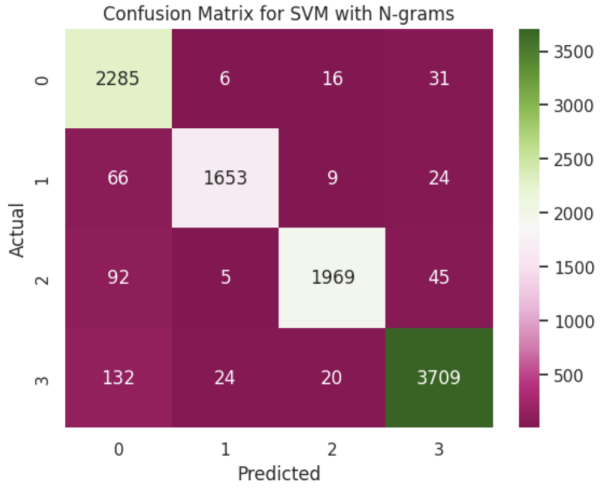

Figure 10: SVM-NGram

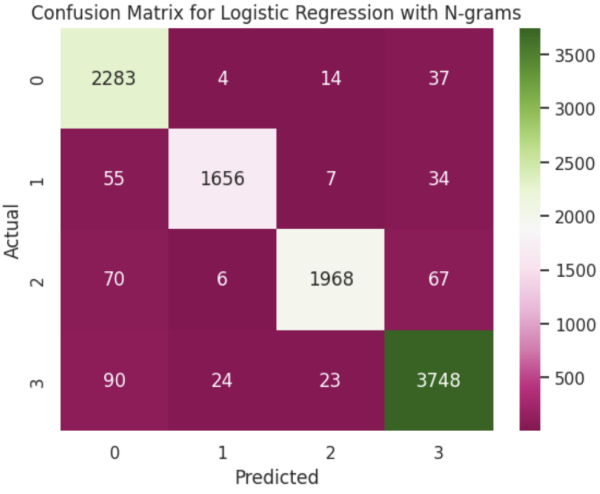

Figure 13: LR-NGram

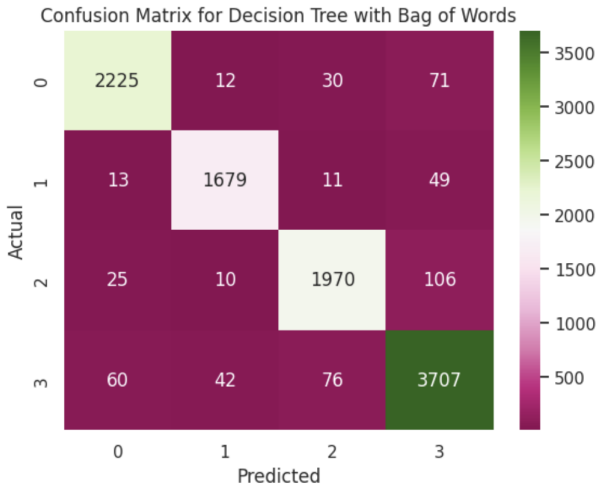

Figure 14: DT-BoW

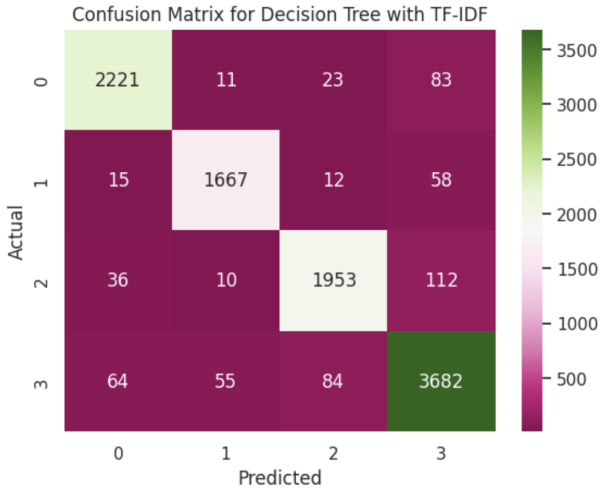

Figure 15: DT-TFIDF

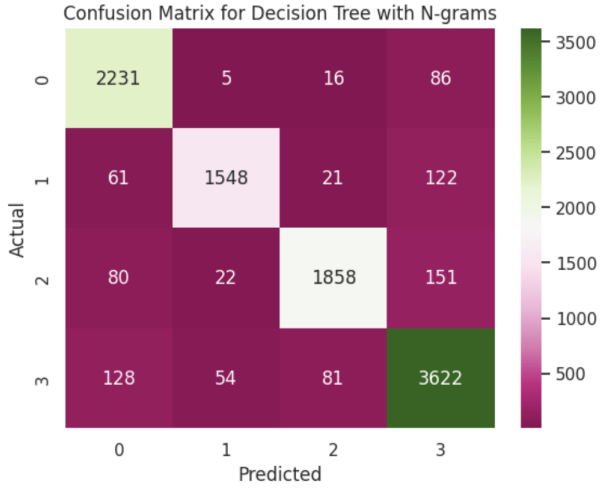

Figure 16: DT-NGram

5.3 Best Model and Feature Combination

Best Overall Combination: SVM with TF-IDF.

- **Accuracy**: 0.9787
- **Precision**: 0.9794
- **Macro F1**: 0.9786
- **Time**: 1.46 seconds

This combination provides the highest accuracy, precision, and F1 score among all tested models and feature sets. It also boasts a competitive training time, making it the most effective model-feature combination for this classification task.

5.4 Conclusion

6 Hyperparameter Tuning

Hyperparameter tuning is finding the optimal set of hyperparameters for a given machine-learning model. This is done using techniques like grid search or random search, where multiple combinations of hyperparameters are tested, and the one yielding the best performance on the validation set is selected.

Here, we use Grid Search.

6.1 Model Evaluation

6.1.1 LinearSVC (Support Vector Classification)

Parameter	Value
C	1
loss	'squared_hinge'
Best Cross-Validation Accuracy	0.9753

Table 4: LinearSVC Best Hyperparameters and Cross-Validation Accuracy

Performance on Test Set:

Metric	Value
Accuracy	0.9787
Precision	0.9794
Recall	0.9778
Macro F1	0.9786

Table 5: LinearSVC Test Set Performance

The LinearSVC model demonstrates excellent performance, achieving high accuracy and macro F1 score. The hyperparameter tuning with a C value of 1 and a squared hinge loss function has helped enhance the model's generalization on unseen data.

6.1.2 Logistic Regression

Parameter	Value
C	10
penalty	'l2'
solver	'lbfgs'
Best Cross-Validation Accuracy	0.9738

Table 6: Logistic Regression Best Hyperparameters and Cross-Validation Accuracy

Performance on Test Set:

Metric	Value
Accuracy	0.9776
Precision	0.9782
Recall	0.9761
Macro F1	0.9771

Table 7: Logistic Regression Test Set Performance

Logistic Regression also performed well, achieving comparable accuracy and macro F1 scores to LinearSVC. Using the lbfgs solver and a C value of 10 helped optimize the model for better performance on the validation and test data.

6.1.3 Decision Tree Classifier

Parameter	Value
criterion	'gini'
max_depth	15
min_samples_leaf	1
min_samples_split	2
Best Cross-Validation Accuracy	0.8071

Table 8: Decision Tree Best Hyperparameters and Cross-Validation Accuracy

Performance on Test Set:

Metric	Value
Accuracy	0.8028
Precision	0.8529
Recall	0.7864
Macro F1	0.8059

Table 9: Decision Tree Test Set Performance

Despite its lower cross-validation accuracy, the Decision Tree classifier demonstrated relatively high precision on the test set (Precision: 0.8529).

Its overall performance, however, is somewhat lacking compared to the other models, with a lower accuracy (0.8028) and macro F1 score (0.8059). This suggests that the model may benefit from further hyperparameter tuning, feature engineering, or other improvements to enhance its generalization and efficiency in predicting outcomes.

6.2 Conclusion

- **Best Model**: The SVM with TF-IDF (using LinearSVC) outperformed the other models, with an accuracy of 0.9619 and a precision of 0.9632. Logistic Regression also performed well, but LinearSVC showed a slight edge in both accuracy and F1 score.
- **Model Recommendations**: For tasks requiring high accuracy and robust performance, LinearSVC with TF-IDF is the preferred model, as it achieved the highest accuracy and precision. Logistic Regression with BoW is another strong candidate, particularly when speed is important. The Decision Tree, though interpretable, underperforms in comparison (Accuracy: 0.7815, Precision: 0.8853) and may require further hyperparameter tuning and feature engineering to improve performance.
- **For Speed and Simplicity**: If training time is absolutely critical and you can sacrifice a very small drop in accuracy, Logistic Regression with Bag of Words (BoW) could be considered. It offers fast inference time and decent performance, with an accuracy of 0.9747. However, its training time of 28.16 seconds is a major drawback when compared to other models like SVM.
- **For Interpretability**: If model interpretability is more important and speed is not as critical, Decision Tree with BoW might be useful. It offers easy interpretability and relatively simple decision-making logic. However, its accuracy (0.9485) and training time (19.89 seconds) are not as competitive when compared to other models like SVM with TF-IDF.

7 OneVsRest Exploration

7.1 Introduction to OneVsRest Exploration

The OneVsRest (OvR) strategy is a well-known approach for tackling multiclass classification problems by decomposing them into multiple binary classification tasks. In this method, each class is treated as "positive" while the remaining classes are considered "negative." This transformation allows us to leverage binary classifiers to solve multiclass problems effectively. The evaluation of this approach involves computing metrics like Precision, Recall, and F1 Score for each class and averaging these metrics to assess overall performance.

7.2 Implementation

To evaluate our model using the OneVsRest strategy, we followed these steps:

- We selected the best-performing model from our prior analysis and wrapped it with `OneVsRestClassifier` from the `sklearn` library.
- The model was trained using the transformed OneVsRest strategy on the training data set $(X_{\text{train\_best}}, y_{\text{train\_best}})$.
- For each class i, a binary classifier was constructed, and we obtained the predicted probabilities for the test dataset.
- For each binary classification task, we calculated the following metrics:
  - **Precision:** Measures the ratio of accurate positive predictions to the number of optimistic predictions made.
  - **Recall:** Indicates the ratio of accurate optimistic predictions to the actual number of positive instances in the test set.
  - **F1 Score:** The harmonic mean of Precision and Recall provides a balanced measure even when dealing with class imbalance.
- We also plotted the Precision-Recall and ROC curves to visually evaluate the model's performance for each class.

7.3 Results

The following evaluation metrics were obtained for each class:

Table 10: Evaluation Metrics for each class (OneVs-Rest)

Class	F1 Score	Precision	
Books	0.98699	0.98815	0
Clothing & Accessories	0.98993	0.98882	0
Electronics	0.98051	0.97977	0
Household	0.98139	0.98018	0

The visualizations for Precision-Recall and ROC are as follows:

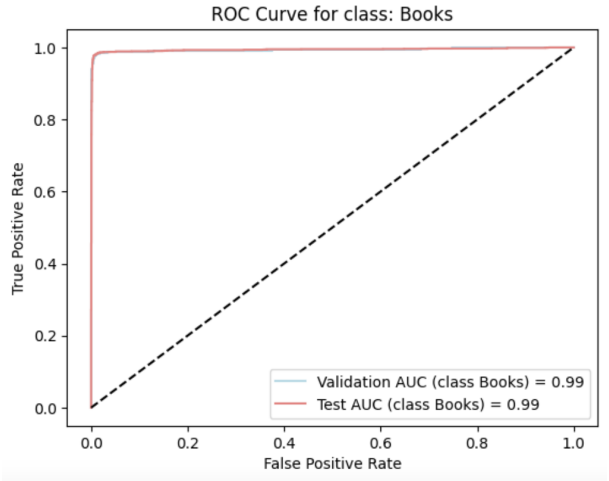

Figure 17: ROC Class: Books

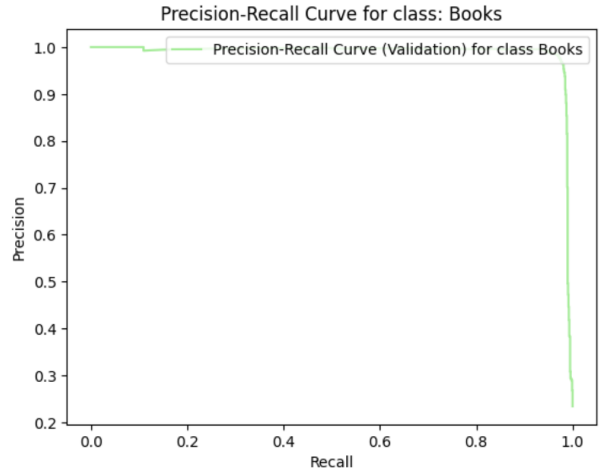

Figure 18: PR Curve Class: Books

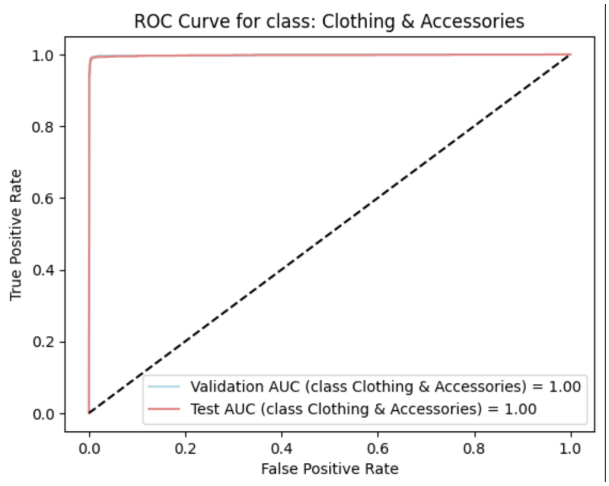

Figure 19: ROC Class: Clothing & Accessories

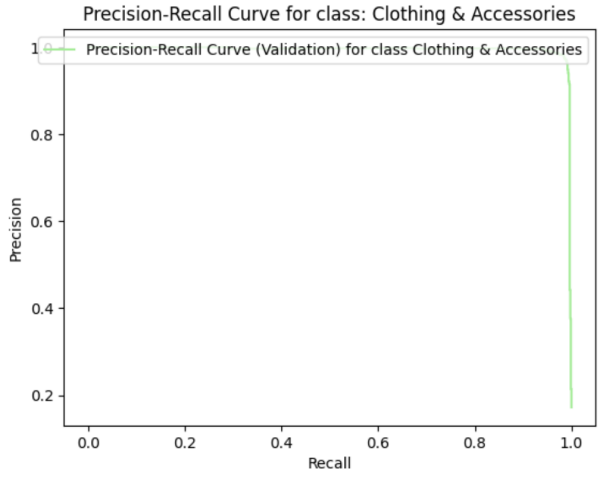

Figure 20: PR Curve Class: Clothing & Accessories

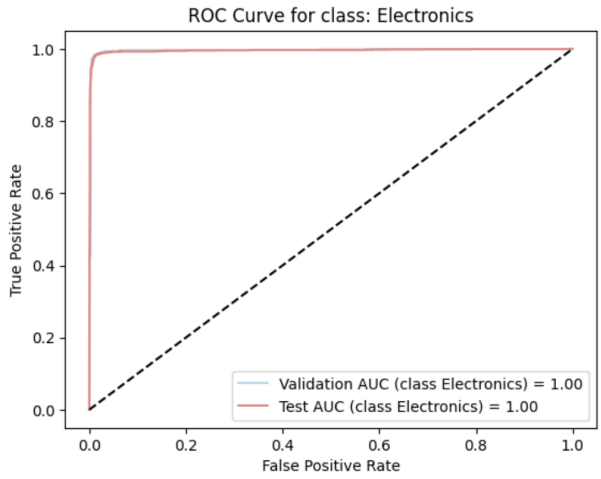

Figure 21: ROC Class: Electronics

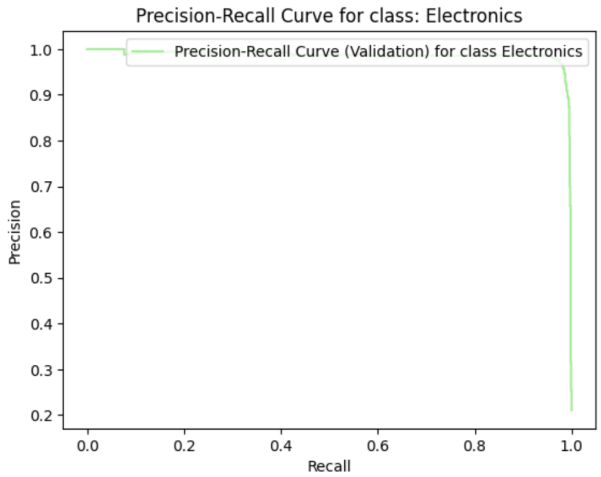

Figure 22: PR Curve Class: Electronics

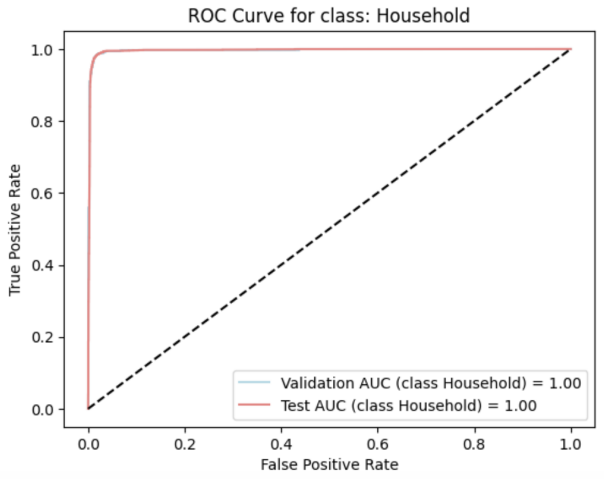

Figure 23: ROC Class: Household

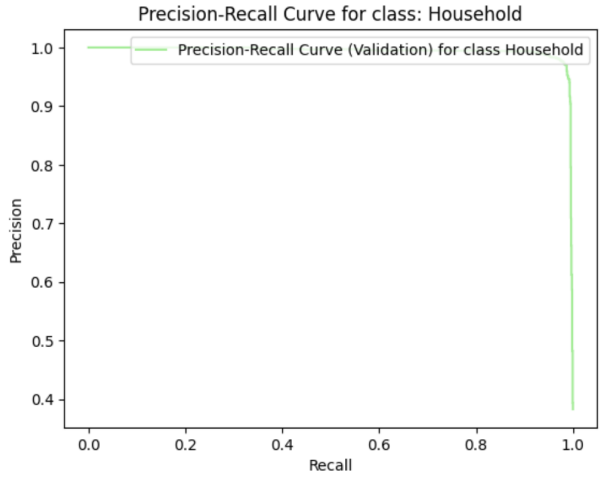

Figure 24: PR Curve Class: Household

7.4 Analysis

From the evaluation metrics:

- **Books:** Achieved the highest recall score of 0.98585, indicating that the model identified most of the positive instances for this class. The F1 score was also good at 0.98699, suggesting a balance between precision (0.98815) and recall.
- **Clothing & Accessories:** The model performed well with an F1 score of 0.98993. The precision (0.98882) and recall (0.99105) metrics were balanced, reflecting the effective identification of positive samples with occasional false positives.
- **Electronics:** Demonstrated a solid F1 score of 0.98051. Both precision (0.97977) and recall (0.98126) were high, showing competent performance distinguishing this class from others.
- **Household:** While this class had slightly lower metrics than the others, it still showed robust performance with an F1 score of 0.98139. Precision (0.98018) and recall (0.98266) were similar, suggesting good overall classification.

The results indicate strong performance across all classes, with minimal variance in F1 scores, showing the model's consistency in handling different classes effectively.

7.5 Conclusion

The OneVsRest strategy effectively evaluated the multiclass classification problem, breaking it into more straightforward binary classification tasks. The model demonstrated high F1 scores across all classes, indicating a satisfactory balance between precision and recall. The visualizations (ROC and Precision-Recall curves) confirmed the model's strong performance, significantly separating it from the chance level.

8 Conclusion

In conclusion, the SVM with TF-IDF emerged as the top-performing model for this multiclass classification task, achieving the highest accuracy (0.9787), precision (0.9794), and macro F1 score (0.9786).

Logistic Regression also demonstrated strong results, particularly when combined with TF-IDF, yielding an accuracy of 0.9776 and precision of 0.9782.

Overall, the combination of TF-IDF features with SVM proved to be the most efficient and accurate approach for classifying E-commerce products, highlighting its effectiveness in this domain.

References

[1] Scikit-learn Documentation. (n.d.). *Vectorization of Text Data*. Retrieved from https://scikit-learn.org/stable/modules/feature_extraction.html#text-feature-extraction

[2] Raschka, S. (2015). *Python Machine Learning: Machine Learning and Deep Learning with Python* (Chapter 5: Text Classification with Scikit-learn).

[3] DataCamp. (n.d.). *Introduction to Support Vector Machines*. Retrieved from https://www.datacamp.com/community/tutorials/svm-classification-python

[4] Google Cloud Documentation. (2024). *Text Classification with AutoML Natural Language*. Retrieved from https://cloud.google.com/natural-language/automl/docs

[5] McKinney, W. (2018). *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython* (Chapter on Data Wrangling).