CS 349
Final Project

CDs & Vinyls Dataset: Individual Task

Ian Shi

Individual Assignment Goals

Improve Group Task Results

In the individual portion of the assignment, I look to continue to add to the work done in the group portion and improve the model predictions.

Apply Neural Networks to Task

Another goal I aim to accomplish is to apply and use the NN tools learned in class to this project. I hope this will not only enhance my model, but also my understanding of deep learning

Individual Assignment Steps

01

Continue to implement and refine existing and new features

02

Use TF-IDF as another form of sentiment analysis

03

Use sklearn Pipeline to perform different tasks with multiple parameters

04

Implement NN with sklearn MLPClassifier

Feature Engineering

In group tasks 1 and 2, I implemented 17 product-focused features that spanned unique reviews, nltk VADER sentiment analysis, normalized time between reviews, max review upvote, and percentage of reviews from verified users. Below are new features added for the individual assignment.
Furthermore, all features were scaled to values between 0 and 1--something we did not do as a group

1

Length of Summary

Average number of words contained in each summary per product

2

TF-IDF

Using sklearn's tfidf vectorizer for another form of sentiment

3

of Verified Reviews

Total raw number of verified reviewers per product

4

Most Recent Review

Time (normalized) from most recent review

New Algorithms & Methods

1. **New Feature Generation**
   RF, Decision Tree, LF algorithms using new features
2. **TF-IDF**
   TF-IDF using sklearn
3. **Pipeline & ColumnTransformer**
   Pipeline & ColumnTransformer to apply TF-IDF
4. **MLPClassifier & Keras**
   Neural Network with sklearn MLPClassifier & Keras

Hyperparameter Optimization

Recursive Feature Elimination

- Sklearn's RFECV function was used to to recursively test which features chosen during our feature engineering step should be prioritized in our model
- Each feature is tested on a 10-fold cross-validation
- Feature rankings are recorded, giving us a list of features for each model that should be used
- Created RFE functions for classifiers w/o RFECV support

Grid Search

- Sklearn's GridSearchCV function was used to iteratively test different hyperparameters for each relevant model
- Each hyperparameter pair was tested on a 10-fold cross-validation, with their mean F1 score recorded
- Collected data was exported to a .csv file where we could analyze which set of hyperparameters gave us the best results

Randomized Search

- Sklearn's RandomizedSearchCV function was used as another optimization method
- Compared to Grid Search (which tests every combination), Randomized Search randomly selects combinations from a given subset. This is beneficial when testing on larger datasets like ours
- Each hyperparameter pair was tested on a 10-fold cross-validation, with their mean F1 score recorded

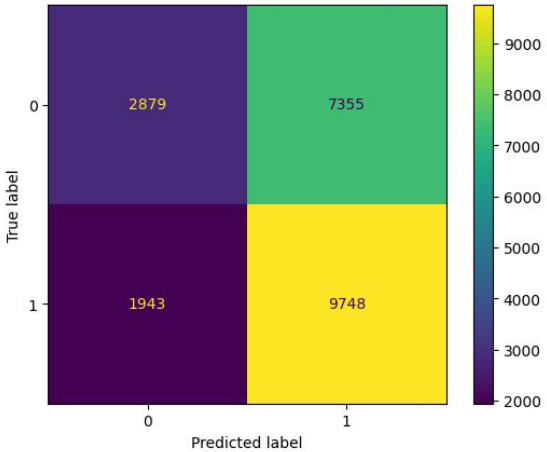
1. New Feature Generation

Previous classification methods such as RF, Decision Tree, and Late Fusion were performed on the dataset with the addition of the new features

Random Forest

max_depth: 1 | n_estimators: 49

Features : all

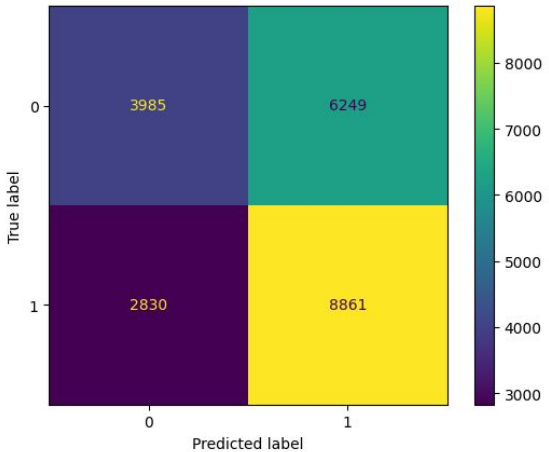


Precision: 0.57 | Recall: 0.83 | F1 Score: 0.68

Decision Tree

max_depth : 4

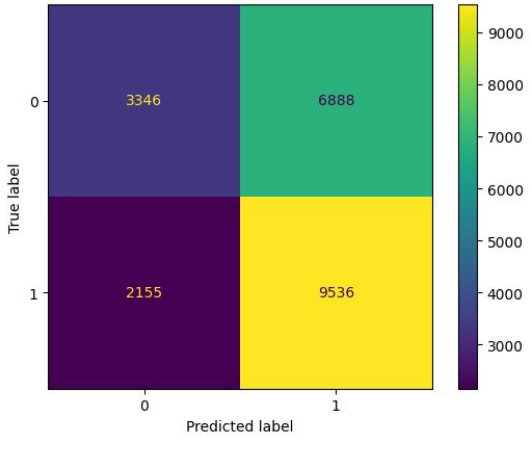
Features : ['reviewerID', 'verified', 'rev_negSentiment', 'summ_negSentiment', 'rev_posNegRatio', 'normalized_time', 'norm_time_diff']



Precision: 0.59 | Recall: 0.76 | F1 Score: 0.66

Late Fusion

Random Forest, Decision Tree, Logistic Regression

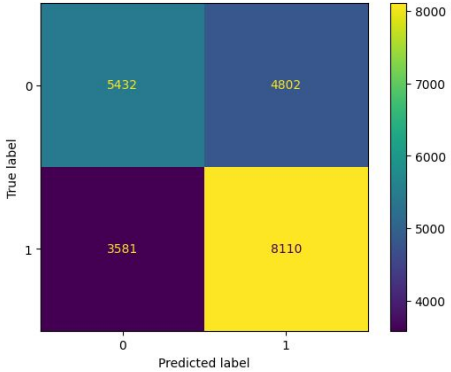


Precision: 0.58 | Recall: 0.83 | F1 Score: 0.685

2. TF-IDF

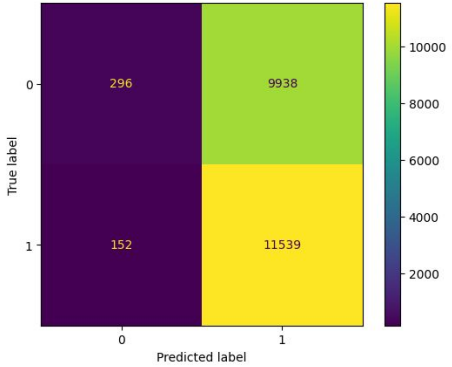
Scikit Learn TFIDF was introduced and used to evaluate review sentiment

TFIDF Logit



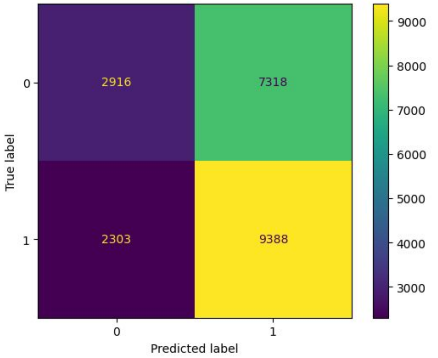
Precision: 0.63 | Recall: 0.69 | F1 Score: 0.66

TFIDF RF



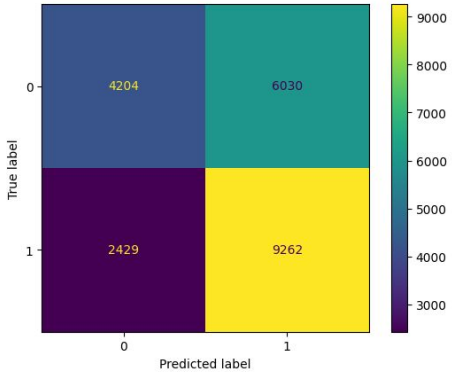
Precision: 0.54 | Recall: 0.99 | F1 Score: 0.70

TFIDF Tree



Precision: 0.56 | Recall: 0.80 | F1 Score: 0.66

TFIDF Late Fusion*



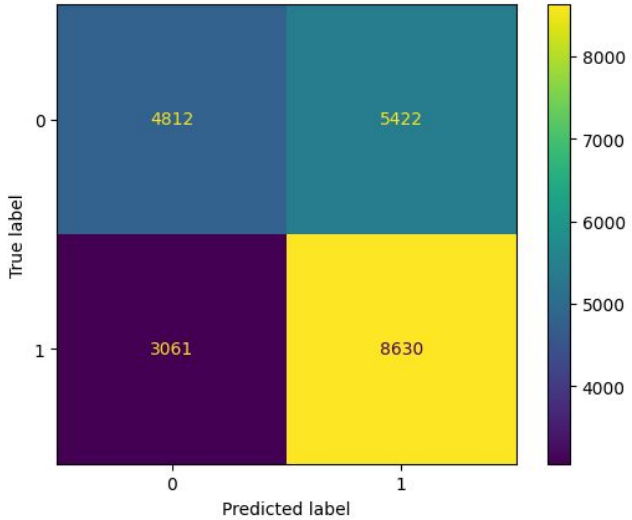
Precision: 0.61 | Recall: 0.79 | F1 Score: 0.70

*Model used

3. Pipeline & ColumnTransformer

Sklearn’s pipeline and column transformer feature was used to test TF-IDF along with the other features

Pipeline + RF



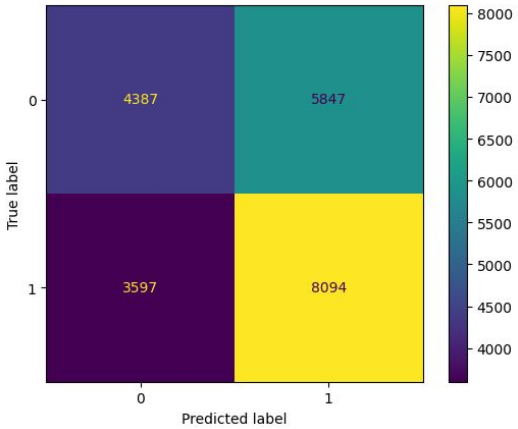
Precision: 0.61 | Recall: 0.74 | F1 Score: 0.67

4. MLPClassifer & Keras

Using sklean MLPClassifer and Keras to implement neural networks

MLPClassifier

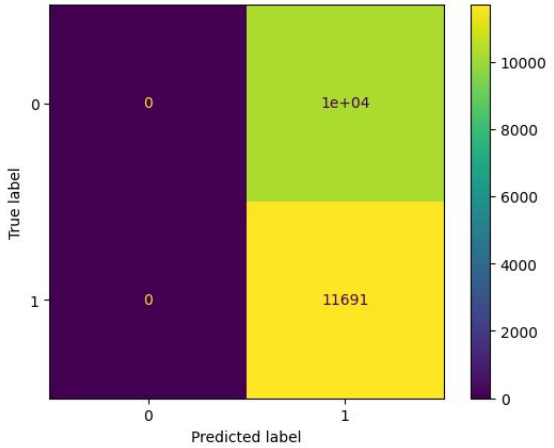
Activation: ‘relu’ | alpha: 0.001 | layer_sizes: (1024, 512, 256, 128)



Precision: 0.58 | Recall: 0.69 | F1 Score: 0.63

Keras

Activation: ‘sigmoid’ | SGD | Dense: (16, 8, 4, 2)



Precision: 0.53 | Recall: 1.00 | F1 Score: 0.69

*Using Keras with F1 score as primary metric leads to prediction of all 1’s after all epochs

Learnings

Feature Engineering

- Feature engineering is one of the most critical aspects of building a solid classifier
- The success of our sklearn implemented classification or NN networks ultimately depends on the values of our features. Thus, in order to increase and improve the model, one of the first things to do is refine and adjust our features
- Feature scaling is also important to ensure the scale of each feature is along the same axis, and no single feature dominates

Hyperparameter Optimization

- For classification methods such as RandomForest and DecisionTree, hyperparameter optimization can be done rather simply using Grid Search or Randomized Search with the tradeoff being time and complexity
- However, optimization with MLP becomes more difficult, especially in regards to the number of hidden layers
- Researching online, there's relatively little "general" layer rules that apply to any dataset
- However, from testing, it seems multiple smaller layers works better than a smaller number of large layers

Machine Learning

- Machine learning is *hard*. There's a lot of different aspects that can go wrong, and little things that can have big outcomes on the result
- It's important to check the confusion matrices and evaluate model metrics in a variety of ways. For example, on the training set, predicting all 1's will give an F1 score of 0.69, but this doesn't translate well to other datasets.

Next Steps

Neural Networks & Deep Learning

- Continue to explore Keras and neural networks. Optimizing layer depth and is difficult and cumbersome, but also important in tuning the best classifier
- With more time, try out different options keras and tensorflow have, explore more techniques and metrics to measure model performance.

Sentiment Analysis

- Sentiment analysis was one of the biggest “black-box” unknowns during the preprocessing stage
- Trying different models such as spaCY instead of nltk could product different or more accurate results

Feature Engineering

- The features I chose gave mean results that made it hard to distinguish between “awesome” and “not awesome” products.
- From the Keras performance, it seemed like the nominal features I choose (excluding TF-IDF), gave its highest F1 scores when we predicted all 1’s.
- Sign that more distinguishing features are needed