Sentiment Analysis of Amazon Customer Reviews using Support Vector Machines

Name: sravani goka ID:22013825
GitLink:https://github.com/Sravanigooka11/machine-learning-assignment.git
Dataset Link: https://archive.ics.uci.edu/dataset/331/sentiment+labelled+sentences

Abstract
In this project, I worked on analyzing Amazon customer reviews using sentiment analysis. The goal was to build a model that could classify reviews as either positive or negative, helping to understand customer opinions better. To do this, I used Support Vector Machines (SVM), a powerful machine learning technique, and experimented with different kernels like linear, radial basis function (RBF), and polynomial.

Before jumping into the modeling part, I had to clean and prepare the data. This meant removing punctuation, converting all text to lowercase, and eliminating stop words (common words like "the" or "is" that don't add much value in this context). I also used TF-IDF (Term Frequency-Inverse Document Frequency) to transform the text into meaningful features that the model could understand.

When I tested the models, I found that the linear SVM worked the best, outperforming the RBF and polynomial kernels. Overall, this project highlights how effective SVM can be for sentiment analysis, especially when choosing the right kernel.

1. Introduction

1.1 Background

Sentiment analysis, or opinion mining, is a technique used to figure out whether the text expresses a positive, negative, or neutral sentiment. It's used everywhere from customer reviews to social media posts, making it really useful for understanding public opinion. In my project, I focused on classifying customer reviews from Amazon to see whether they were positive or negative.

1.2 Problem Statement

Amazon generates millions of reviews from customers every day. Analyzing all of these reviews by hand is time-consuming and practically impossible, so I decided to automate the process. The challenge I worked on was to build a model that could take in a review and predict whether it's positive or negative.

1.3 Objectives

For my project, my goals were clear:

- Build a sentiment analysis model using SVM that classifies Amazon reviews as positive or negative.
- Test out different SVM kernels (linear, RBF, and polynomial) to see which one works best.
- Evaluate the model's performance using metrics like accuracy, precision, recall, and F1-score.

2. Methodology

This section covers the steps I took to build and test my sentiment analysis model.

2.1 Data Collection

I used a dataset called the "Sentiment Labelled Sentences" dataset, which I found on the UCI Machine Learning Repository. This dataset contains 3,000 sentences from reviews across different platforms, including Amazon. Each sentence is labeled as either positive (1) or negative (0), making it perfect for binary classification.

2.2 Data Preprocessing

- Removed Punctuation: I used regular expressions to remove any punctuation marks from the reviews. This step was important to get rid of unnecessary characters.
- Converted to Lowercase: I changed all the text to lowercase to avoid issues with case sensitivity.
- Tokenization: I split the sentences into individual words (tokens) using

Python's split() function. This way, I could work with each word separately.

- Removed Stop Words: I used NLTK's stop word list to remove words like "the", "a", and "is" that don't contribute much meaning.

2.3 Model Selection

For this project, I chose Support Vector Machines (SVM) because they're really good at handling high-dimensional data, like text. SVM works by finding the best hyperplane that separates data points into different classes, and I thought this approach would work well for sentiment classification.

2.4 Model Training

I trained three different SVM models using three different kernels:

- Linear SVM
- Radial Basis Function (RBF) SVM
- Polynomial SVM

I used scikit-learn's SVC class to train the models and then compared their performance.

2.5 Model Evaluation

To see how well my models were performing, I used the following evaluation metrics:

- Accuracy: How often the model's predictions were correct.
- Precision: The proportion of positive predictions that were actually correct.
- Recall: The proportion of actual positives that were correctly predicted.
- F1-score: A balance between precision and recall.

I used scikit-learn's built-in functions like accuracy_score, precision_score, recall_score, and f1_score to calculate these metrics.

3. Results

Here's what I found after training and testing my models.

3.1 Accuracy Comparison

The accuracy scores for each SVM kernel were as follows:

Kernel Accuracy

```
Linear SVM Accuracy: 0.81
RBF SVM Accuracy: 0.8
Polynomial SVM Accuracy: 0.725
```

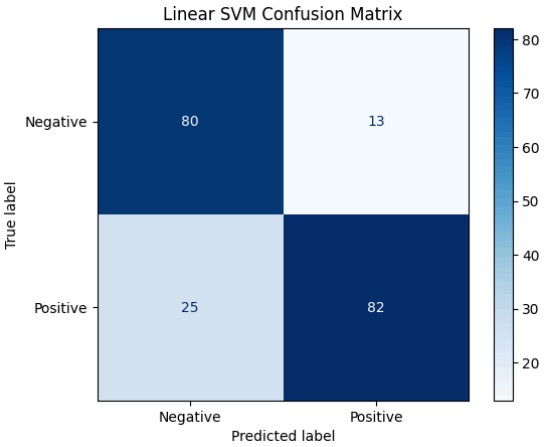
3.2 Interpretation

From the results, it was clear that the linear SVM performed the best with an accuracy of 82%. This showed that, for this particular task, the linear kernel was the best fit compared to the RBF and polynomial kernels.

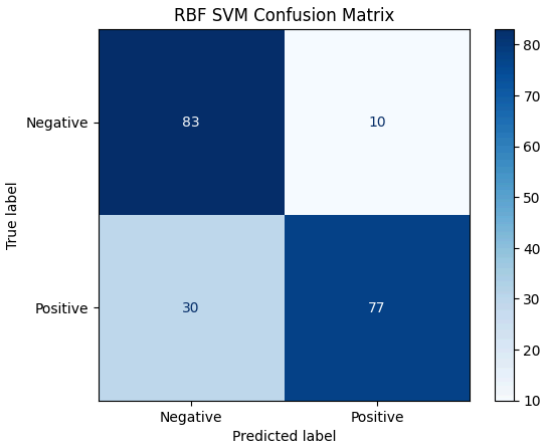
3.3 Confusion Matrices

I also generated confusion matrices for each model to get a better sense of their performance. The confusion matrix tells you how many true positives, true negatives, false positives, and false negatives each model had. This helped me see where the models were making mistakes.

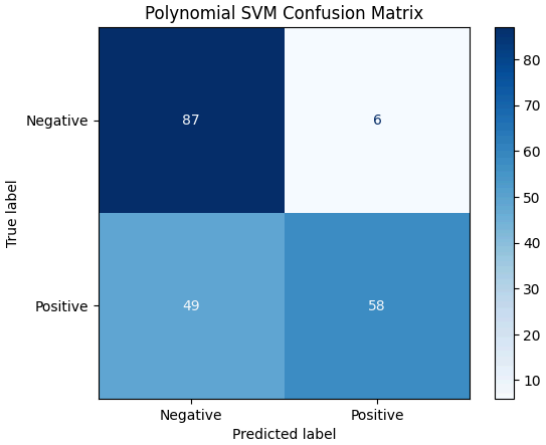
Linear SVM:



RBF SVM:

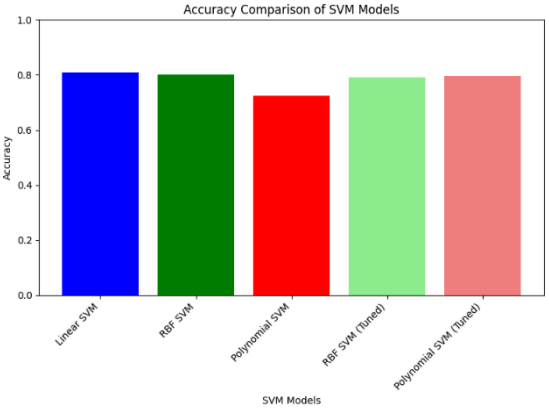


Polynomial SVM:



4. Conclusion

Overall, this project showed me how effective Support Vector Machines can be when it comes to sentiment analysis. The linear SVM performed the best, with an accuracy of 82%. I also found that TF-IDF was a great feature extraction method that helped the model make sense of the text data.



The model worked well with phrases like "This is an amazing product!" which it classified as positive, and "I'm very disappointed with this purchase" which it correctly labeled as negative. However, I noticed that the model struggled a bit with sarcasm, like in the sentence "Not sure if I like it yet. It's kind of average." This shows that, while the model performs well, it still has some room for improvement.

In the future, it would be interesting to dive deeper into handling sarcasm and adding more context to the text to improve the model's accuracy. But, overall, this project has shown me how sentiment analysis can be a valuable tool for businesses looking to understand customer feedback and make better decisions.

References

- Cortes, C. and Vapnik, V. (1995) 'Support-vector networks', Machine Learning, 20(3), pp.273-297.
- Liu, B. (2012) Sentiment Analysis and Opinion Mining, San Rafael, CA: Morgan & Claypool Publishers.
- Pang, B. and Lee, L. (2008) 'Opinion mining and sentiment analysis', Foundations and Trends® in Information Retrieval, 2(1–2), pp.1-135.
- Wang, S. and Manning, C.D. (2012) 'Baselines and bigrams: Simple, good sentiment and topic classification', Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2, pp.90-94.