Sentiment Analysis Model Transferability

Final Project Report IST 736

Wesley Stevens

Introduction

Sentiment classification has been a well-researched topic in the field of text mining and NLP. Many websites and products generate user reviews, and some get so many that looking through every single one is infeasible. Categorizing these reviews and determining if it is a positive or negative review can help the company determine if their product is well liked or what could be better about it. Further predicting the rating (1-5) a user would give based on the text can further separate sentiment for later analysis.

Researchers have been able to achieve good results on this sentiment analysis problem, but these models often only do well on the type of data they were trained for [1]. In this paper, we explore how well models do when transferred from one dataset to another. We will use the Kaggle musical instrument review dataset and determine how it transfers to the Kaggle movie review dataset in our analysis [2] [3]. This paper will seek to explore and answer these questions: How do known classification methods perform when doing sentiment analysis on a transferred domain? How can their performance be improved on this task?

Data Preprocessing

This transferred classification problem will seek to classify text from 1-5 where the classes are ratings. The data is retrieved from the Kaggle website, and we utilize the text and the rating given to build a classification model. There are over 10,000 data points in each dataset. We note that each product and each user that rates a product appears at least 5 times in each dataset [2].

We begin by exploring the data. The musical instruments dataset is a csv file with nine columns. These columns include the text and overall rating (which we will use for classification) as well as review ID, time reviewed, summary, etc. The first five datapoints are shown below. We note that although the “overall” rating column shows that each value is a float, we found that there are only whole numbers included in this dataset. This proves that the proposed classification is representable of the raw data.

```
1 data=pd.read_csv("./Musical_instruments_reviews.csv")
2 data.head()
```

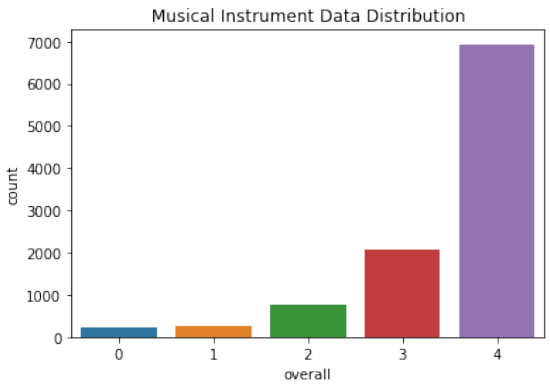
	reviewerID	asin	reviewerName	helpful	reviewText	overall	summary	unixReviewTime	reviewTime
0	A2IBPI20UZIR0U	1384719342	cassandra tu "Yeah, well, that's just like, u...	[0, 0]	Not much to write about here, but it does exac...	5.0	good	1393545600	02 28, 2014
1	A14VAT5EAX3D9S	1384719342	Jake	[13, 14]	The product does exactly as it should and is q...	5.0	Jake	1363392000	03 16, 2013
2	A195EZSQDW3E21	1384719342	Rick Bennette "Rick Bennette"	[1, 1]	The primary job of this device is to block the...	5.0	It Does The Job Well	1377648000	08 28, 2013
3	A2C00NNG1ZQQG2	1384719342	RustyBill "Sunday Rocker"	[0, 0]	Nice windscreen protects my MXL mic and preven...	5.0	GOOD WINDSCREEN FOR THE MONEY	1392336000	02 14, 2014
4	A94QU4C90B1AX	1384719342	SEAN MASLANKA	[0, 0]	This pop filter is great. It looks and perform...	5.0	No more pops when I record my vocals.	1392940800	02 21, 2014

A quick examination shows that there are a few null entries. We will deal with these by simply dropping the rows. Doing this will not skew our dataset since there are less than 10 null values in the columns we are interested in.

```
[11]  1 data.isna().sum()

reviewerID          0
asin                0
reviewerName       27
helpful             0
reviewText          7
overall             0
summary             0
unixReviewTime      0
reviewTime          0
dtype: int64
```

Lastly, we see in the figures below the support of every rating option for both datasets. This shows us that the musical instrument review dataset is skewed toward higher ratings with very little support for very negative reviews. We also see that the movie review dataset is skewed toward mediocre ratings (2-4), but it does have far more support for every category. We will have to take this into consideration as we build our models, such as down sampling the categories with large support and up-sampling from the categories with little support during training. We will split the data 80:20 for training and testing respectively.



Data distribution for the Musical Instruments Review dataset

Data distribution for the Movie Review dataset

Vectorization

In our experiments, we use sklearn's "CountVectorizer" method to extract features of the raw text to make it consumable for our models [4]. We compare vectorizing with unigrams, bigrams and unigrams, and stemming with the methods we use to produce classifications. Stemming is done through nltk's "EnglishStemmer," which snowball-stems English words [5].

Classifiers

Multinomial Naïve Bayes

Multinomial Naïve Bayes (NB) is an NB classifier for multinomial models [6]. NB works by applying Bayes' theorem with strong independence assumptions between the features. This makes it suitable for classification with discrete features such as word counts for text classification. The multinomial distribution normally requires integer feature counts.

Support Vector Machines

Support Vector Machines (SVM) take the data and optimally determine boundary lines to linearly separate the data into categories [7]. Algorithms such as the Kernel Method may decrease dimensionality, thereby allowing these SVMs to accurately draw boundary lines to fit non-linear data. We will use it to separate text that is indicative of high or low ratings.

XGBoost

eXtreme Gradient Boosting (XGBoost) is a supervised learning algorithm that uses an ensemble of boosted decision trees to determine the boundary lines between classifications [8]. It is well known for its training speed, versatility, and awareness of sparse data, which makes it a good choice for our imbalanced dataset.

Results

Baseline

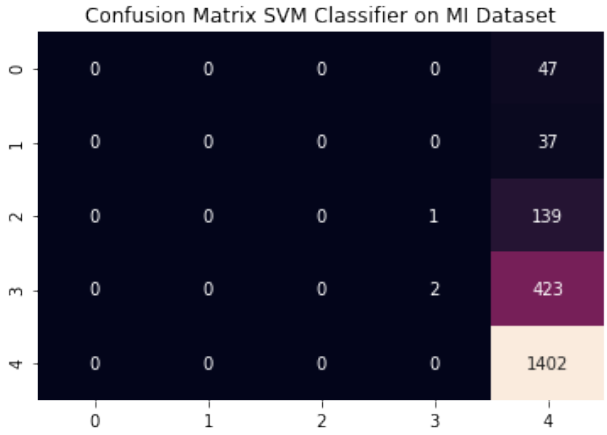
As a baseline, we computed the performance of the SVM, Multinomial NB, and XGBoost classifiers using the default values. We fit the vectorizer with the training data from both datasets so that the consumable form of the data from each set would stay consistent. If we had not done so, then the separate consumable forms of each dataset would have looked the same but each indices would refer to different vectorized words. This fitting of the vectorizer to both datasets will not impact the true transferability performance of these models, however, since the words that do not exist in both datasets will be reflected in the vectorized data as a zero and will not have any weight in training the models. The following table shows the baseline accuracy for the classifiers on each dataset using a stemmed vectorizer, a unigram vectorizer, a bigram and unigram vectorizer, and a bigram stemming vectorizer. Each result is the average of a 3-fold cross validation.

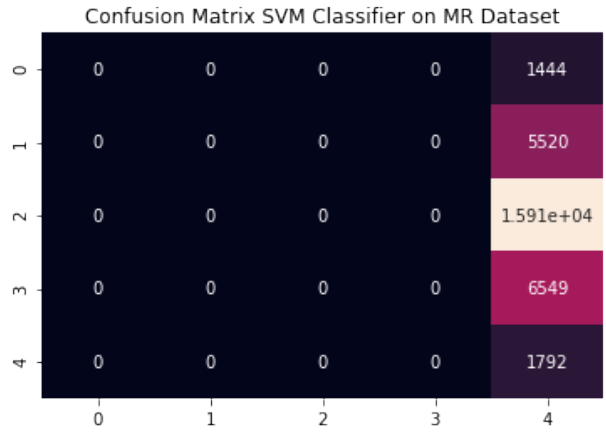
Vectorizer	XGBoost MI	SVM MI	NB MI	XGBoost MR	SVM MR	NB MR
Stemmer	70.16%	68.45%	66.94%	5.77%	5.74%	16.91%
Unigram	68.94%	68.41%	66.07%	5.74%	5.74%	15.22%
1-2 gram	69.20%	68.40%	68.50%	5.74%	5.74%	16.76%
1-2 gram & Stemming	68.89%	68.45%	66.94%	5.74%	5.74%	16.91%

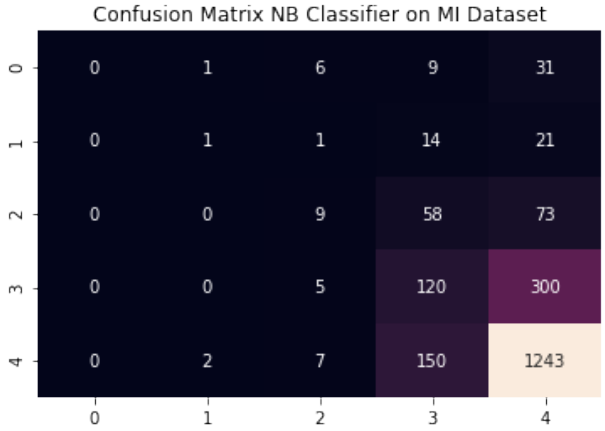
Table 1. Accuracies of each model with varied vectorizers. MI stands for the Musical Instruments dataset whereas MR stands for the Movie Review dataset.

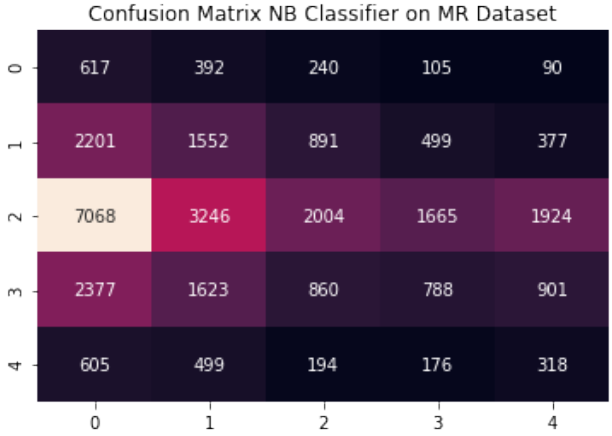
Table 1 shows us that the stemming vectorizer performs the best overall between each model. The other vectorizers perform similarly well, so for the rest of our experiments, we will use this

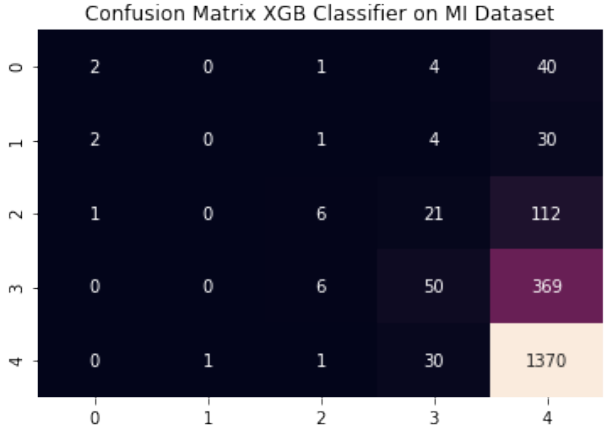
vectorizer. Confusion matrices for the models using the stemming vectorizer are shown below. As seen in the figure, these models seem to be classifying everything as a high rating. This is most likely caused by the imbalanced data. Recall that the y axis is the truth and the x axis is the prediction.

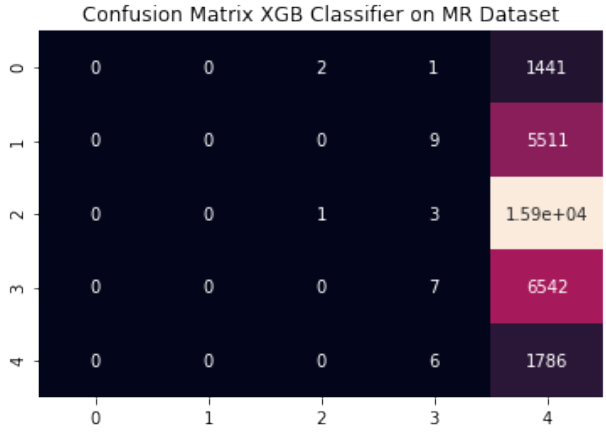












Improving by Equalizing Training Data

To equalize the data, we up-sample the data points with little support, namely the data with corresponding ratings 0-3. We then down-sample the data points with overwhelming support,

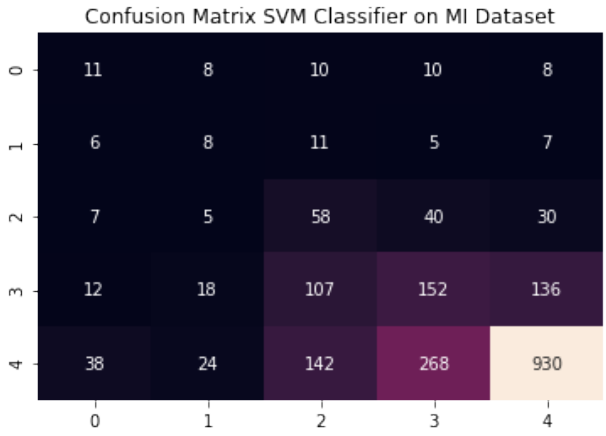
namely data with corresponding ratings 4 and 5. The new distribution of data is equalized so that every category has a support of 1000. In our tests, we found this to give the best results.

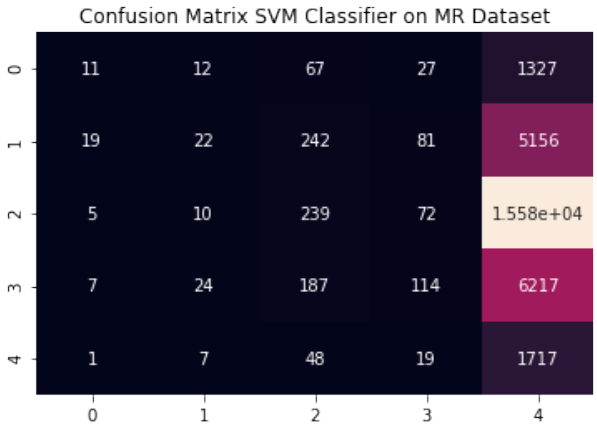
Training, testing, and transferring this model, we record the accuracy scores below. As seen in Table 2, each model performs slightly worse, but the score on the movie review dataset is increased.

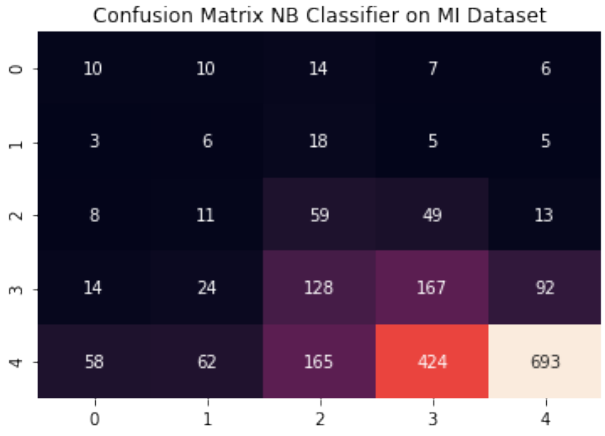
XGBoost MI	SVM MI	NB MI	XGBoost MR	SVM MR	NB MR
54.56%	56.51%	45.59%	8.37%	6.74%	17.45%

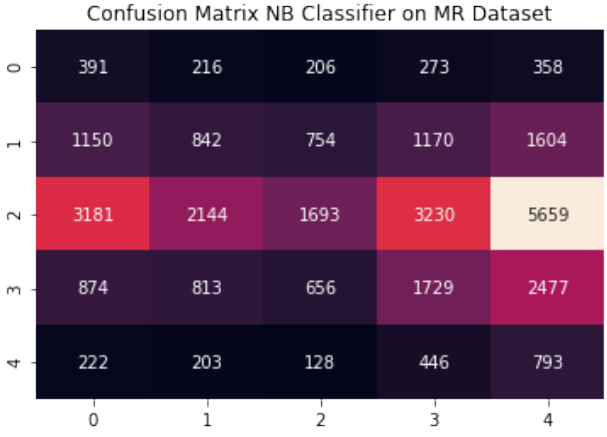
Table 2. Accuracy scores of models trained on the equalized data

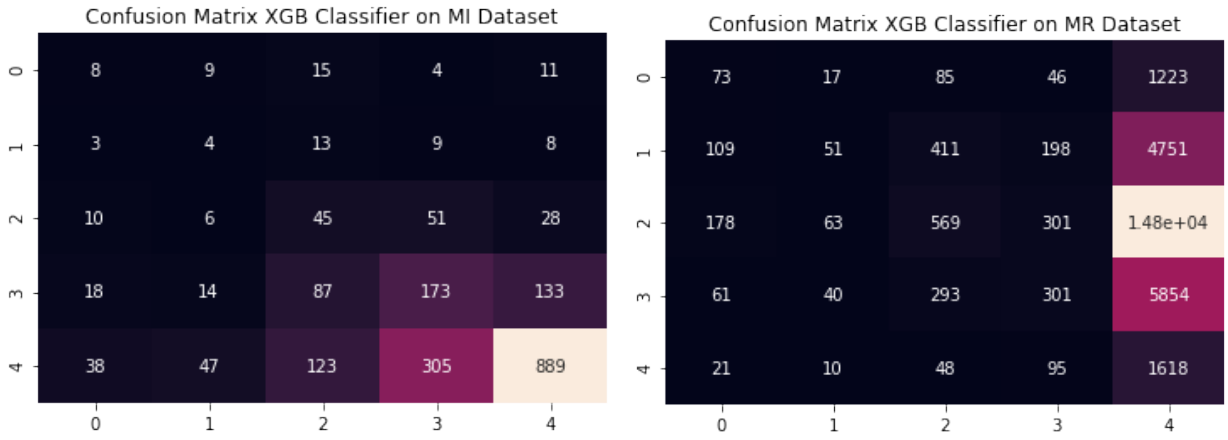
A look at the confusion matrices also tells us that these classifiers are better overall, as they do not categorize everything as a “Strong Positive” review but make informed predictions. Of note, the SVM and XGBoost classifiers still classify most everything as a “Strong Positive” review, but there is an improvement as some of the other classes get predictions.











Improving by Including Data from the MR Dataset in Training

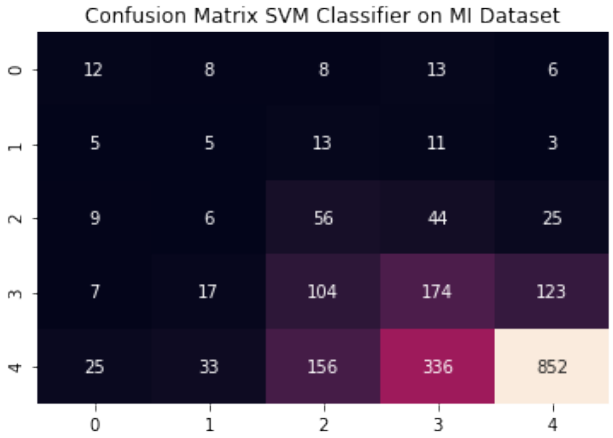
Next, we will show how including training data from the exterior dataset improves model transference. We will use the equalized dataset for our evaluation. We take 1000 samples from each dataset per class for training. The "% Data" column in Table 3 represents the percentage of data used for training of the pre-allocated 1000 datapoints per class. The data in Table 3 shows the accuracy scores of the models with varied amounts of data used from the MR set along with the 1000 datapoints from the MI set per class.

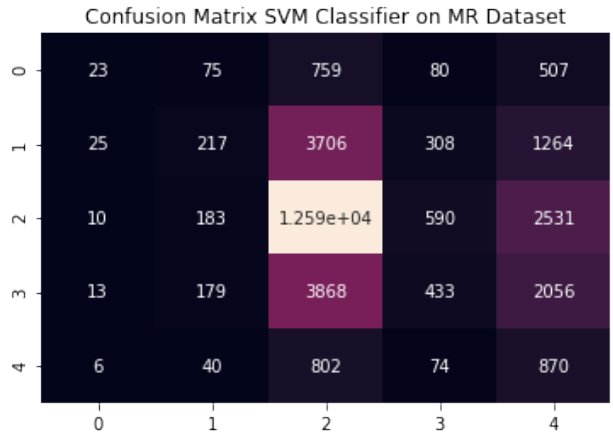
% Data	XGBoost MI	SVM MI	NB MI	XGBoost MR	SVM MR	NB MR
0%	54.56%	56.51%	45.59%	8.37%	6.74%	17.45%
10%	54.31%	54.46%	49.98%	12.12%	10.28%	19.24%
25%	54.02%	54.12%	48.03%	40.56%	36.11%	19.07%
50%	51.78%	52.95%	48.61%	44.37%	50.13%	19.85%
100%	51.43%	51.10%	45.10%	45.10%	50.57%	19.82%

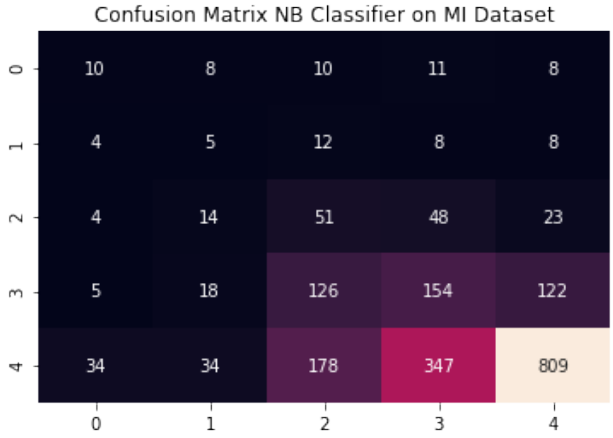
Table 3. Accuracies of each model with varied vectorizers. MI stands for the Musical Instruments dataset whereas MR stands for the Movie Review dataset.

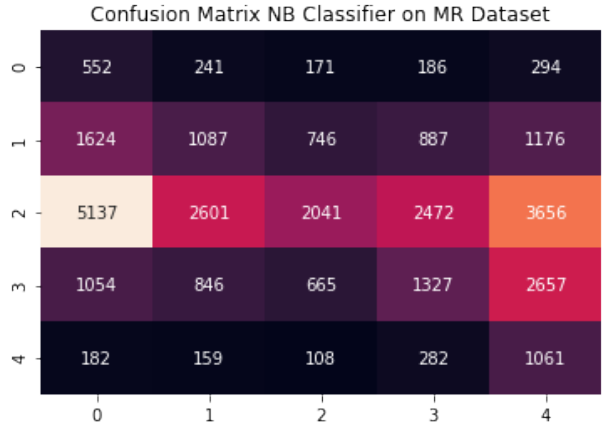
As seen in the table, with only a quarter of the data from the new dataset, the accuracies on the MR dataset are comparable to that of the accuracies on the MI dataset, which we trained the models on. We also see that by doing this, the accuracy of the models on the MI dataset only go down by a few percentage points. Overall, the SVM classifier seems to perform the best and the most consistently, with the XGBoost classifier close behind. The Multinomial NB classifier doesn't seem to be improving despite the new data.

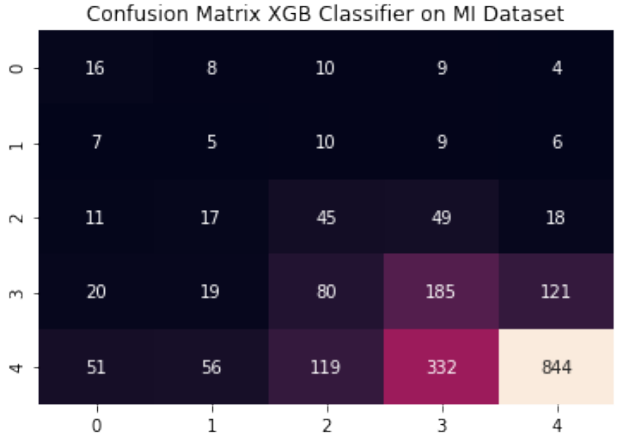
We show the confusion matrices of the 25% data test in the figure below. We see that the classifiers tend to classify most things into class 4 for the MI set and class 3 for the MR set. This is representative of the skewed data for each dataset. We surmise that the classifiers have distinguished between the language in each dataset and found a solution that allows for the best accuracy across both datasets.

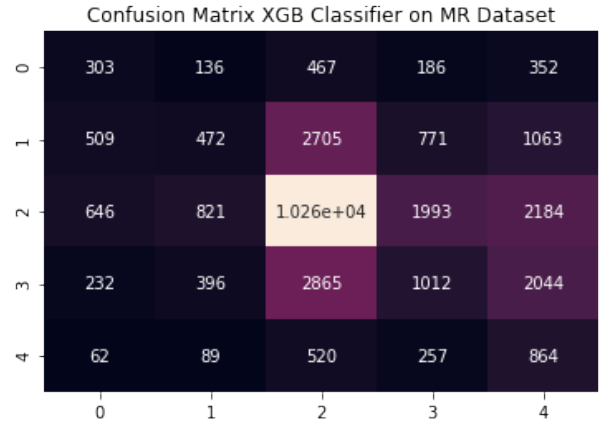












Improving by Removing Unique Words

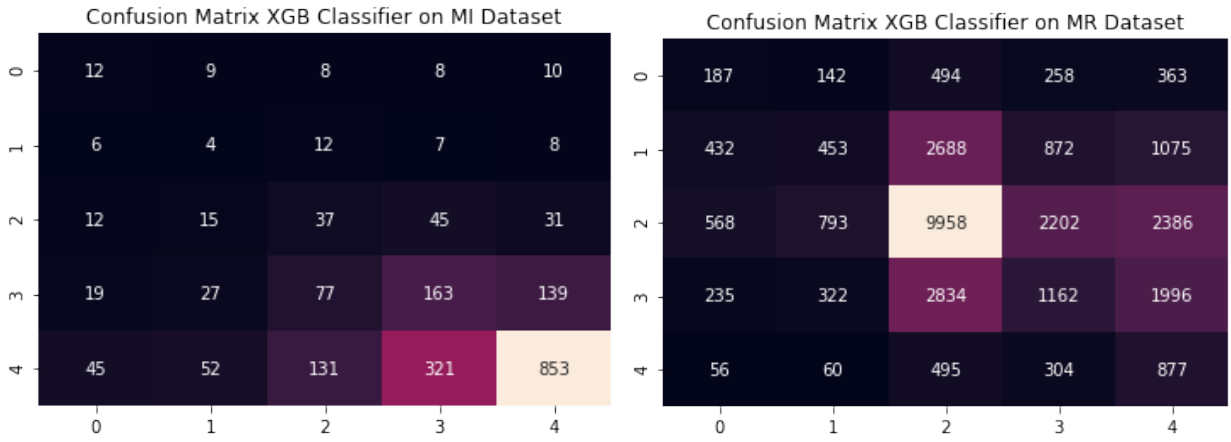
In this experiment, we remove the unique words that are not in both datasets. This should make transferability easier since the models can then build their decision boundaries on information that is found in both datasets. Table 4 shows the accuracies on each of the models and test data by using purely the data from the MI dataset to train and it also shows the accuracies when a fraction of the MR dataset is added to the training set.

% Data	XGBoost MI	SVM MI	NB MI	XGBoost MR	SVM MR	NB MR
0%	69.53%	68.41%	54.17%	5.81	5.74%	16.40%
25%	52.12%	55.38%	48.66%	40.49%	39.57%	18.70%

Table 4. Accuracies of each model with varied vectorizers. MI stands for the Musical Instruments dataset whereas MR stands for the Movie Review dataset.

As seen in Table 4, the models perform well on the MI data when no MR data is added into the training set, but it seems that removing the unique words had little to no impact on the performance of these classifiers. Additionally, we see no improvement to the MR data inclusion during training with this method. We suppose that the similar results to the previous methods may be an indicator that this method is equivalent in effect to the way we vectorized the data, and thus has no effect on the performance of the classifiers.

The following figure shows the confusion matrices of the results when using 25% of the MR data in training. We see that these are very similar to the confusion matrices in the previous experiment.



Conclusion

Sentiment analysis is an important tool for companies to understand their user feedback, but training an accurate model is expensive and difficult. Ideally, a model that would transfer across domains would be used, but the problem of domain transfer is even more difficult, and is one many researchers have attempted to solve.

In this paper, we compared sentiment analysis across a transferred domain using three different classifiers. We found that model performance across domains is dismal at best, even with a well performing model. However, we also found that there are ways to improve a model's performance for this domain transference task. These methods include balancing the dataset and including a small portion of data from the external dataset during training. We found that removing unique words from the dataset before training may also help performance. On our best attempt, we were able to achieve a comparable accuracy from the initial dataset to the second dataset with only a 5% performance loss on the initial dataset.

References

[1] J. Meng, Y. Long, Y. Yu, D. Zhao and S. Liu, "Cross-Domain Text Sentiment Analysis Based on CNN_FT Method," Dalian Minzu University, 2019.

[2] "Amazon Musical Instruments Reveiws," Kaggle, [Online]. Available: https://www.kaggle.com/eswarchandt/amazon-music-reviews. [Accessed 6 8 2020].

[3] "Sentiment Analysis on Movie Reviews," Kaggle, [Online]. Available: https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews. [Accessed 6 8 2020].

[4] "sklearn.feature_extraction.text.CountVectorizer," scikit learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html. [Accessed 6 8 2020].

[5] "nltk.stem package," NLTK, [Online]. Available: https://www.nltk.org/api/nltk.stem.html. [Accessed 6 8 2020].

[6] "sklearn.naive_bayes.MultinomialNB," scikit learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html?highlight=multinomial%20nb#sklearn.naive_bayes.MultinomialNB. [Accessed 6 8 2020].

[7] "sklearn.svm.SVC," scikit learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html. [Accessed 6 8 2020].

[8] "XGBoost Documentation," [Online]. Available: https://xgboost.readthedocs.io/en/latest/. [Accessed 6 8 2020].