Performance comparison of classification algorithms for Sentiment Analysis and Domain Adaptation on Amazon Reviews Dataset

Anirudh Kamalapuram Muralidhar, Mohit Galvankar, Srijhari Katragadda, Venkata Koppu

***Abstract*—The main aim of this paper is to compare the working various algorithms such as SVM, Naive Bayes and Logistic Regression and how well a model trained in one domain can be made effective in another domain (Domain adaptation). The motivation for this paper is the increase in importance of text classification and sentiment analysis, there is always some sentiment associated with each sentence that we breath. So developing a generalized model which can work across all domains is still a challenging task in hand. This paper tries to make one generalized model which can work well with amazon review dataset and twitter dataset.**

***Keywords*—Sentiment Analysis, Domain Adaptation, Naïve Bayes, SVM, Logistic Regression, Amazon, Twitter.**

I. Introduction

Sentiment analysis of data is holding higher importance these days, every breathe of us has a sentiment associated, this project develops classifier model for sentiment classification and domain adaptation.

Domain adaptation plays an important role too since, there can be situations where model developed using one particular data finds its application in other domain, this is because training data on each domain can be costly and in effective. Sentiment Analysis is the process of determining the emotional tone behind a series of words, used to gain an understanding of the the attitudes, opinions and emotions expressed within an online mention. It is extremely useful in social media monitoring as it allows us to gain an overview of the wider public opinion behind certain topics.

Domain adaptation is a field associated with machine learning and transfer learning. The main purpose to perform Domain Adaptation is to store the knowledge gained while solving one problem and to apply this knowledge to a different but related problem. This paper shows how a model trained on Amazon Reviews data set can be adapted to another test data set like Twitter.

A. Why Sentiment Analysis?

Humans are fairly intuitive when it comes to interpreting the tone of a piece of writing.

Consider the following sentence: My flights been delayed. Brilliant!

Most humans would be able to quickly interpret that the person was being sarcastic. We know that for most people having a delayed flight is not a good experience (unless theres a free bar as recompense involved). By applying this contextual understanding to the sentence, we can easily identify the sentiment as negative.

Without contextual understanding, a machine looking at the sentence above might see the word brilliant and categorize it as positive.

Hence, sentiment analysis is important to enable social success, which impacts search success, be clear about the metrics and the tools you deploy to capture, organize, and report those metrics. Customer optimization relies strongly on analysing the sentiments.

B. Domain Adapatation and its importance

With such widely-varying domains, researchers and engineers who build sentiment classification systems need to collect and curate data for each new domain they encounter. Even in the case of market analysis, if automatic sentiment classification were to be used across a wide range of domains, the effort to annotate corpora for each domain may become prohibitive, especially since product features change over time. We envision a scenario in which developers annotate corpora for a small number of domains, train classifiers on those corpora, and then apply them to other similar corpora.

However, this approach raises two important questions. First, it is well known that trained classifiers lose accuracy when the test data distribution is significantly different from the training data distribution 1. Second, it is not clear which notion of domain similarity should be used to select domains to annotate that would be good proxies for many other domains. We propose solutions to these two questions and evaluate them on a corpus of reviews for music category from Amazon.

Types of Domain Adaptation:

There are several contexts of domain adaptation. They differ in the informations considered for the target task.

1.The **supervised domain adaptation**: all the examples considered are supposed to be labeled

M. Shell is with the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, 30332 USA e-mail: (see http://www.michaelshell.org/contact.html).

J. Doe and J. Doe are with Anonymous University.

Manuscript received April 19, 2005; revised January 11, 2007.

2. The **unsupervised domain adaptation**: the learning sample contains a set of labeled source examples, a set of unlabeled source examples and an unlabeled set of target examples.

3. The **semi-supervised domain adaptation**: in this situation, we also consider a "small" set of labeled target examples.

Despite of the progresses by the existing approaches, they so far have only been limited to macroscopically examining the distribution similarity by tuning to statistical properties of the samples as a whole—when comparing distributions, all the samples are used. This notion is stringent, as it requires all discrepancies to be counted for and forces learning inefficiently (or even erroneously) from hard cases which might be just outliers to the target domains. In contrast, the key insight that not all instances are created equally in terms of adaptability. Thus, examining distribution similarity microscopically at the instance level using landmarks. Landmarks are defined as a subset of labeled instances from the source domain. These instances are distributed similarly to the target domain. As a result, they are expected to function as a conduit connecting the source and target domains to facilitate adaptation.

In addition, an interesting problem in domain adaptation: given multiple source domains and a target domain, how can we select which source domain to pair with the target domain? This is an especially important problem to address in order to apply domain adaptation to real-world problems. This problem is addressed proposing a rank of domains (ROD) measure that ranks source domains based on how suitable they are for a source-trained classifier to adapt to the target domain. The metric integrates two pieces of information: geometrically how much the subspaces of the source and the target domains overlap, and statistically how similarly the target and source data are distributed in the subspaces.

II. Related Work

Sentiment classification has advanced considerably since the work of Pang et al. (2002), which we use as our baseline. Thomas et al. (2006) use discourse structure present in congressional records to perform more accurate sentiment classification. Pang and Lee (2005) treat sentiment analysis as an ordinal ranking problem. In our work we only show improvement for the basic model, but all of these new techniques also make use of lexical features. Thus we believe that our adaptation methods could be also applied to those more refined models.

While work on domain adaptation for sentiment classifiers is sparse, it is worth noting that other researchers have investigated unsupervised and semisupervised methods for domain adaptation. The work most similar in spirit to ours that of Turney (2002). He used the difference in mutual information with two human-selected features (the words excellent and poor) to score features in a completely unsupervised manner. Then he classified documents according to various functions of these mutual information scores. We stress that our method improves a supervised baseline. While we do not have a direct comparison, we note that Turney (2002) performs worse on movie reviews than on his other datasets, the same type of data as the polarity dataset.

We also note the work of Aue and Gamon (2005), who performed a number of empirical tests on domain adaptation of sentiment classifiers. Most of these tests were unsuccessful. We briefly note their results on combining a number of source domains. They observed that source domains closer to the target helped more. In preliminary experiments we confirmed these results. Adding more labeled data always helps, but diversifying training data does not. When classifying kitchen appliances, for any fixed amount of labeled data, it is always better to draw from electronics as a source than use some combination of all three other domains. Domain adaptation alone is a generally well-studied area, and we cannot possibly hope to cover all of it here. We are able to significantly outperform basic structural correspondence learning (Blitzer et al., 2006). We also note that while Florian et al. (2004) and Blitzer et al. (2006) observe that including the label of a source classifier as a feature on small amounts of target data tends to improve over using either the source alone or the target alone, we did not observe that for our data. We believe the most important reason for this is that they explore structured prediction problems, where labels of surrounding words from the source classifier may be very informative, even if the current label is not. In contrast our simple binary prediction problem does not exhibit such behavior. This may also be the reason that the model of Chelba and Acero (2004) did not aid in adaptation. Finally we note that while Blitzer et al. (2006) did combine SCL with labeled target domain data, they only compared using the label of SCL or non-SCL source classifiers as features, following the work of Florian et al. (2004). By only adapting the SCLrelated part of the weight vector v, we are able to make better use of our small amount of unlabeled data than these previous techniques.

III. Data (Collection and Labeling)

A. Amazon Review dataset

In order to achieve our goals, we've used the Amazon review dataset. This dataset contains product reviews and metadata from Amazon, including 142.8 million reviews spanning May 1996 - July 2014.

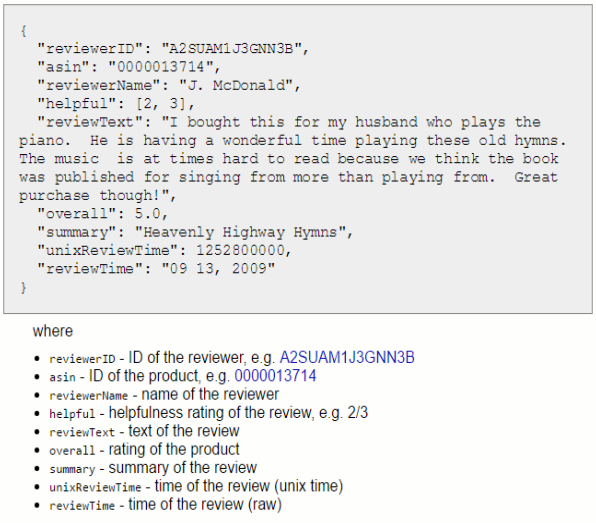
The review dataset is properly segregated into different domain categories. Some of which are listed above :-

This dataset includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs).

Following is a snippet of a review in json format :-

TABLE I. MY CAPTION

Category	Number of Reviews
Books	22,507,155
Electronics	7,824,482
Movies and TV	4,607,047
CDs and Vinyl	3,749,004
Clothing, Shoes and Jewelry	5,748,920
Home and Kitchen	4,253,926
Kindle Store	3,205,467
Sports and Outdoors	3,268,695
Health and Personal Care	2,982,326
Grocery and Gourmet Food	1,297,156
Musical Instruments	500,176



```
{
  "reviewerID": "A2SUAM1J3GNN3B",
  "asin": "0000013714",
  "reviewerName": "J. McDonald",
  "helpful": [2, 3],
  "reviewText": "I bought this for my husband who plays the
piano.  He is having a wonderful time playing these old hymns.
The music  is at times hard to read because we think the book
was published for singing from more than playing from.  Great
purchase though!",
  "overall": 5.0,
  "summary": "Heavenly Highway Hymns",
  "unixReviewTime": 1252800000,
  "reviewTime": "09 13, 2009"
}
```

where

- `reviewerID` - ID of the reviewer, e.g. A2SUAM1J3GNN3B
- `asin` - ID of the product, e.g. 0000013714
- `reviewerName` - name of the reviewer
- `helpful` - helpfulness rating of the review, e.g. 2/3
- `reviewText` - text of the review
- `overall` - rating of the product
- `summary` - summary of the review
- `unixReviewTime` - time of the review (unix time)
- `reviewTime` - time of the review (raw)

For our analysis, we will use only the positive and negative class. So the ratings which are given as 1, 2 are considered as negative class and the rating 3, 4, 5 are considered as positive class for our analysis. For our analysis from the amazon review dataset we will have data with 11922 positive class and 9078 negative class.

B. Twitter dataset

We use the twitter dataset as our out domain dataset and the model that is being built with the amazon dataset will be tested on the twitter dataset.

The data has been collected by the Stanford university where we have about 1.6 lakhs tweets where we consider class ”0” as negative and class ”2” and ”4” as positive sentiment for classification. For the analysis we will have twitter data with 3997 positive class and 4001 with negative class.

The various columns are explained as follows
1 - the polarity of the tweet (0 = negative, 2 = neutral, 4 = positive)
2- the tweet ID(2087)
3 - Tweet date (Sat May 16 23:58:44 UTC 2009)
4 - the query
5 - Username that tweeted (robotickilldozr)
6 - the tweet text (@sachinstr wow wat a game!!)

IV. TOOLS

Python programming language has been used to perform these tasks with mainly the usage of packages such as scikit learn, pandas, numpy and gensim.

The graphs obtained are done using the package matplotlib from Python.

The main concepts used in this analysis:

Word2Vec: Word2vec is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks, that are trained to reconstruct linguistic contexts of words: the network is shown a word, and must guess which words occurred in adjacent positions in an input text. The order of the remaining words is not important (bag-of-words assumption). After training, word2vec models can be used to map each word to a vector of typically several hundred elements, which represent that word’s relation to other words. This vector is the neural network’s hidden layer. Word2vec relies on either skip-grams or continuous bag of words (CBOW) to create neural word embeddings. It was created by a team of researchers led by Tomas Mikolov at Google. The algorithm has been subsequently analysed and explained by other researchers.

Part of Speech Tags: In corpus linguistics, part-of-speech tagging (POS tagging or POST), also called grammatical tagging or word-category disambiguation, is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition and its contexti.e., its relationship with adjacent and related words in a phrase, sentence, or paragraph. A simplified form of this is commonly taught to school-age children, in the identification of words as nouns, verbs, adjectives, adverbs, etc.

Once performed by hand, POS tagging is now done in the context of computational linguistics, using algorithms which associate discrete terms, as well as hidden parts of speech, in accordance with a set of descriptive tags. POS-tagging algorithms fall into two distinctive groups: rule-based and stochastic. E. Brill’s tagger, one of the first and most widely used English POS-taggers, employs rule-based algorithms.

V. DATA PRE-PROCESSING

Pre-processing is one of the main process in developing a good text classifier. So for this, few pre-processing steps are being done which helps in improving the accuracy of the classifier.
1. Remove the HTML entities from the data.
2. Remove all special characters and digits from the data.
3. Convert all characters to lower case in the reviews.
4. Add Part of Speech tags in the reviews, so show the effectiveness we run the classifier on data with and without

Part of Speech tags.

The above steps are done for both twitter and amazon review datasets.

```
Out[9]: "Just when I thought Enya couldn't possibly get any better, I found &quot;The
Memory of Trees&quot;.  &quot;Trees&quot; is Enya's best album, and like
&quot;Watermark&quot; and &quot;Shepherd Moons&quot;, each song is  a gem.  No other music
conveys beauty, love, joy and peace the way Enya's  does, and &quot;Trees&quot; is no
exception.  Enya is a world treasure, and  &quot;The Memory of Trees&quot; is a must-have
for anyone who loves her  music."
```

```
Out[13]: 'just_RB thought_VBN enya_RB couldnt_NN possibly_RB get_VB better_JJR found_VBN
memory_NN trees_NNS trees_NNS enyas_VBP best_JJS album_NN like_IN watermark_NN shepherd_NN
moons_NNS song_VBP gem_NN music_NN conveys_NNS beauty_VBP love_VB joy_NN peace_NN way_NN
enyas_FW trees_NNS exception_VBP enya_JJ world_NN treasure_NN memory_NN trees_NNS
musthave_VBP anyone_NN loves_NNS music_NN'
```

VI. Method and data splitting

A. Supervised Machine Learning

We used Supervised Machine Learning inorder to classify the processed data.

Supervised learning is the machine learning task of observing a function from labeled trained data. The training data has a set of training examples. In supervised learning, each example is a pair consisting of an input vector and an output value. A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly measure the class labels for unlabeled data. This requires the learning algorithm to develop a generalized model from the training data to unseen situations.

Types of classifiers used :-

Support Vector Machines
In machine learning, support vector machines are supervised learning models with learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked for belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

The following parameters are used when implemented in Python:

1. The penalty parameter C is set to 1.
2. The kernel method used is 'rbf' (radial basis function kernel).

3. The shrinking paramter is set to True, where it will use heuristic shrinking.

4. The tolerance stopping parameter (tol) is set to value 1e-3.

5. The class weight parameter is set uniform to all the class labels available.

Naive Bayes Classifier
In machine learning, naive Bayes classifiers is a simple probabilistic classifiers based on applying Bayes' theorem with independence assumptions between the features.
Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem.
Bayes theorem provides a way of calculating the posterior probability, P(c|x), from P(c), P(x), and P(x|c). Naive Bayes classifier assume that the effect of the value of a predictor (x) on a given class (c) is independent. This assumption is called class conditional independence.

The following parameters are used when implemented in Python:

1. No parameter is to be used when we use a Gaussian Naive Bayes classifier with python.

Logistic Regression
In statistics, logistic regression is a regression model where the dependent variable (DV) is categorical.Logistic regression measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function, which is the cumulative logistic distribution.

The following paramters are used when implemented in Python:

1. The penalty parameter is set to be the default value 'l2'.

2. The regularization strength parameter is set to the default value 1.

3. The solver parameter specifies the algorithm involved which is set to the default value 'liblinear'.

4. The tolerance (tol) paramter is set to 1e-3.

5. max_iter: The maximum number iterations taken to get converge value is set to default value 100.

B. Data Splitting

1. We have used 10-fold cross validation to test our results.
2. For domain adaptation part, we have developed a model that can adapt to both amazon and twitter and it is tested on 40% of the data.

C. Experimental setup

1. First we develop model from amazon data and test them with 10 fold cross validation without POS.
2. The above step is repeated again with POS tags and both of them are compared.
3. We also do step 2 on twitter data this time.
4. We then develop a model with amazon data and test them on twitter data.
5. Next, we develop a model from random sampling of both amazon and twitter data.
6. This model is tested on both amazon and twitter data and the performance of domain adaptation are compared.

VII. Evaluation

A. Performance Metrics

The evaluation metrics that are used as follows.
1. Accuracy : It is defined as the closeness of a measure when compared to the true value. Its formula is given by
Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$

2. Precision: It is defined as the probability of a randomly selected document being relevant. Its formula is given by
Precision = $\frac{TP}{TP+FP}$

3. Recall: It is defined as randomly selected relevant document is retrieved in the search. Its formula is given by
Recall = $\frac{TP}{TP+FN}$

4. F-1 Measure: It takes into account of both the precision and the recall by calculating their harmonic mean. Its formula is given by
F-1 = $\frac{2*(Precision*Recall)}{(Precision+Recall)}$

5. Confusion Matrix: It is defined as the matrix which shows the performance of the algorithm. Each row in the matrix is an instance in the actual class and each column corresponds to the predicted class.

6. ROC curve: The ROC curve is a curve plotted between false positive rate and true positive rate of the data, the area under the curve shows the performance of the classifier. Higher the area under the curve, better the classifier is.

TABLE II. Results for various algorithm on amazon review dataset with 10 fold CV

Algorithm	Accuracy	Precision	Recall	F-1
SVM	71.53	0.72	0.72	0.72
Naive Bayes	66.96	0.68	0.67	0.67
Logistic	70.77	0.71	0.71	0.71

TABLE III. Confusion Matrix for SVM

	Negative	Positive
Negative	6200	2878
Positive	3099	8823

VIII. Results

A. Results for various classifiers tested on amazon review dataset without POS

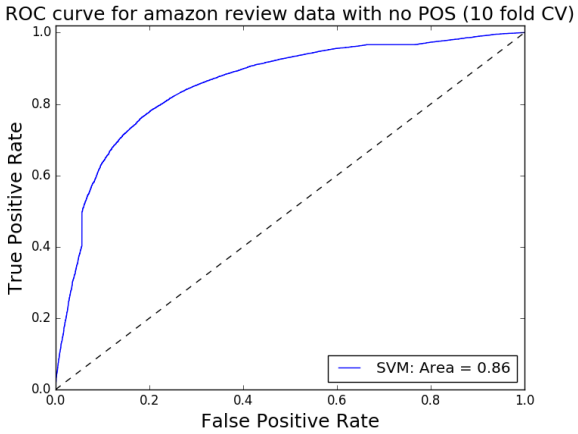


Fig 1. ROC Curve for amazon review data with SVM and without POS.

Fig 2. ROC Curve for amazon review data with Naive Bayes and without POS.

TABLE IV. CONFUSION MATRIX FOR LOGISTIC REGRESSION

	Negative	Positive
Negative	5632	3446
Positive	2691	9231

TABLE V. CONFUSION MATRIX FOR NAIVE BAYES

	Negative	Positive
Negative	6173	2905
Positive	4029	7893

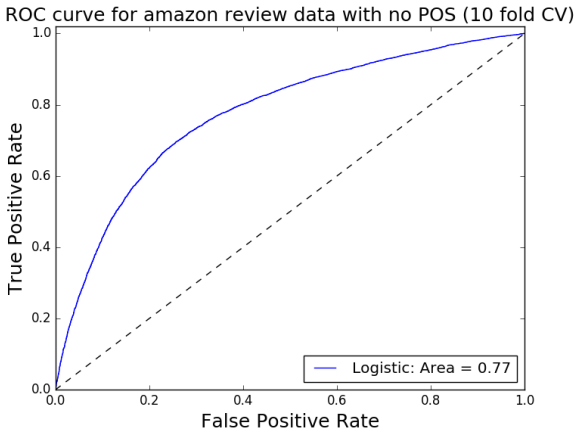


Fig 3. ROC Curve for amazon review data with Logistic Regression and without POS.

Inference: From the above tables (II, III, IV, V) and graph we could say that SVM is best algorithm that works well on the data followed by Logistic regression and at the last naive Bayes classifier.

B. Results for various classifiers tested on amazon review dataset with POS

TABLE VI. RESULTS FOR VARIOUS ALGORITHM ON AMAZON REVIEW DATASET WITH 10 FOLD CV

Algorithm	Accuracy	Precision	Recall	F-1
SVM	74.77	0.75	0.75	0.75
Logistic	74.17	0.74	0.74	0.74
Naive Bayes	66.53	0.67	0.67	0.67

TABLE VII. CONFUSION MATRIX FOR LOGISTIC REGRESSION

	Negative	Positive
Negative	6487	2591
Positive	2833	9089

TABLE VIII. CONFUSION MATRIX FOR SVM

	Negative	Positive
Negative	6977	2101
Positive	3196	8726

TABLE IX. CONFUSION MATRIX FOR NAIVE BAYES

	Negative	Positive
Negative	6122	2956
Positive	4071	7851

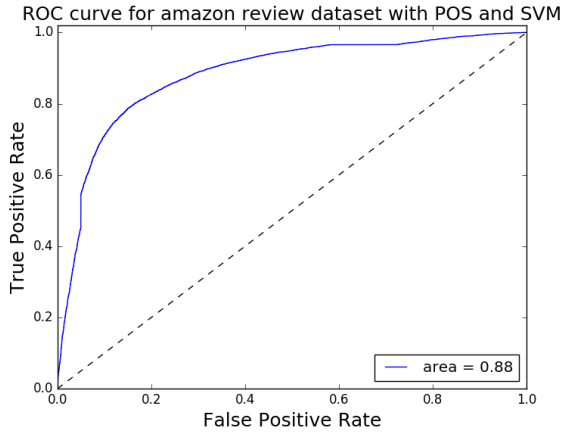


Fig 4. ROC Curve for amazon review data with SVM.

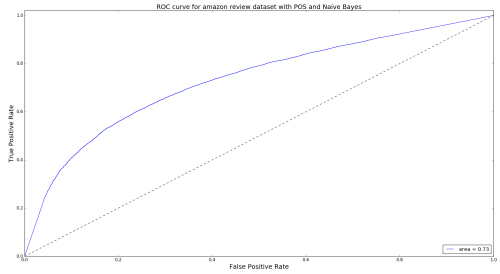


Fig 5. ROC Curve for amazon review data with Naive Bayes.

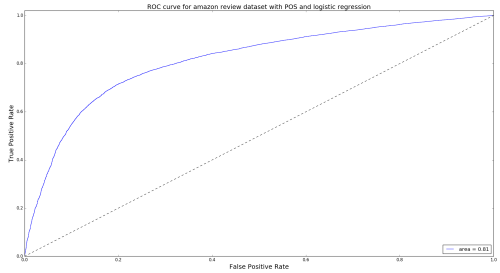


Fig 6. ROC Curve for amazon review data with Logistic Regression

Inference: From the above tables(VI,VII,VIII,IX) and graph we could say that SVM is best algorithm that works well on the data followed by Logistic regression and at the last naive Bayes classifier.

Also we would see a greater increase in efficiency when we

add Part of Speech tagger to the review data for training.

C. Results for various classifier for the twitter dataset

TABLE X. RESULTS FOR VARIOUS ALGORITHM ON TWITTER DATASET WITH 10 FOLD CV

Algorithm	Accuracy	Precision	Recall	F-1
SVM	57.75	0.58	0.58	0.58
Logistic	58.88	0.59	0.59	0.59
Naive Bayes	54.03	0.55	0.54	0.53

TABLE XI. CONFUSION MATRIX FOR LOGISTIC REGRESSION

	Negative	Positive
Negative	2454	1547
Positive	1742	2255

TABLE XII. CONFUSION MATRIX FOR SVM

	Negative	Positive
Negative	2478	1523
Positive	1855	2142

TABLE XIII. CONFUSION MATRIX FOR NAIVE BAYES

	Negative	Positive
Negative	2834	1167
Positive	2509	1488

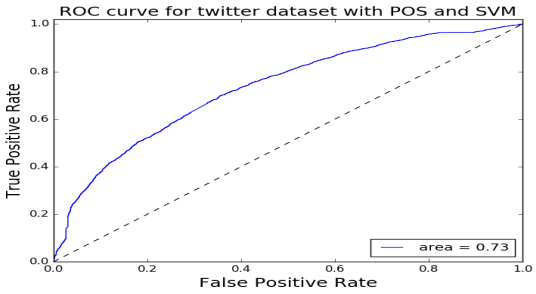


Fig 7. ROC Curve for twitter data with SVM.

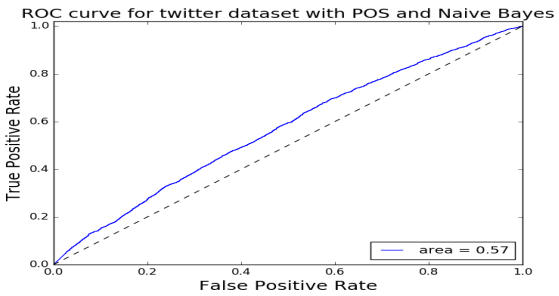


Fig 8. ROC Curve for twitter data with Naive Bayes.

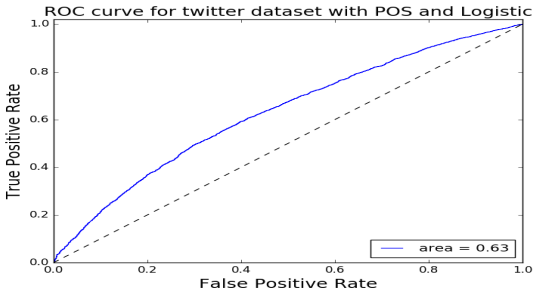


Fig 9. ROC Curve for twitter data with Logistic regression.

Inference: From the results (table - X,XI,XII,XIII) we could see that the classifier doesn't work with the data, but of all the algorithms SVM works well compared to the other algorithms such as logistic regression and Naive Bayes.

D. Results for various classifiers when trained on amazon review dataset and tested on twitter dataset

TABLE XIV. CLASSIFICATION REPORT

Algorithm	Accuracy	Precision	Recall	F-1
SVM	48.69	0.49	0.49	0.48
Naive Bayes	47.77	0.47	0.48	0.46
Logistic	47.32	0.47	0.47	0.47

TABLE XV. CONFUSION MATRIX FOR LOGISTIC REGRESSION

	Negative	Positive
Negative	1688	2313
Positive	1900	2097

TABLE XVI. CONFUSION MATRIX FOR SVM

	Negative	Positive
Negative	1358	2643
Positive	1460	2537

TABLE XVII. CONFUSION MATRIX FOR NAIVE BAYES

	Negative	Positive
Negative	1113	2888
Positive	1289	2708

Fig 10. ROC Curve for when amazon trained data tested on twitter.

Inference: From the above tables(XIV,XV,XVI,XVII) and graph we could say that the model trained on amazon doesn't

work well on the twitter data compared to the actual classifier developed on the twitter data.

So we perform domain adaptation to increase the accuracy of the data.

E. Results for various classifiers when developed with domain adaptability data and tested on twitter data

TABLE XVIII. CLASSIFICATION REPORT

Algorithm	Accuracy	Precision	Recall	F-1
SVM	53.71	0.54	0.54	0.53
Naive Bayes	52.78	0.53	0.53	0.51
Logistic	52.06	0.52	0.52	0.52

TABLE XIX. CONFUSION MATRIX FOR SVM

	Negative	Positive
Negative	2475	1526
Positive	2176	1821

TABLE XX. CONFUSION MATRIX FOR LOGISTIC REGRESSION

	Negative	Positive
Negative	2834	1167
Positive	2609	1388

TABLE XXI. CONFUSION MATRIX FOR NAIVE BAYES

	Negative	Positive
Negative	2110	1891
Positive	1943	2054

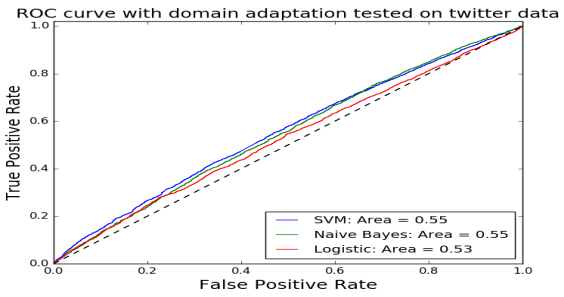


Fig 11. ROC Curve with domain adaptation tested on twitter data.

Inference: From the above tables(XVIII,XIX,XX,XXI) and graphs we can see that the new model developed by taking samples from both amazon and twitter data has shown some great improvement when tested on twitter data when compared to the result of amazon trained model on the twitter data.

TABLE XXII. CLASSIFICATION REPORT

Algorithm	Accuracy	Precision	Recall	F-1
SVM	56.51	0.57	0.57	0.57
Naive Bayes	63.71	0.63	0.64	0.63
Logistic	56.7	0.57	0.57	0.57

TABLE XXIII. CONFUSION MATRIX FOR SVM

	Negative	Positive
Negative	4906	4172
Positive	4960	6962

F. Results for various classifiers when developed with domain adaptability data and tested on amazon data

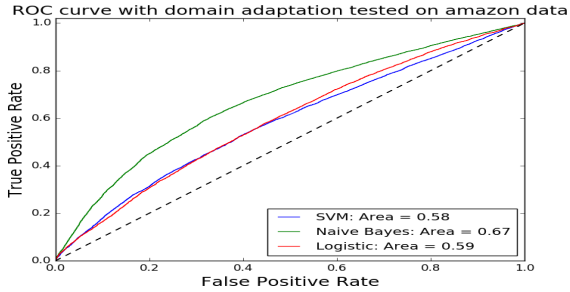


Fig 12. ROC Curve with domain adaptation tested on amazon review data.

Inference: From the above tables(XXII,XXIII,XXIV,XXV) and graphs we can see that the new model developed by taking samples from both amazon and twitter data has shown some decent result when tested on amazon review dataset. We could see a decrease in efficiency of the classifier since the model developed is to adopt both twiiter test data and amazon test data.

For all, the Naive Bayes seems to work well compared to the other classifier in this case.

IX. DISCUSSION AND CONCLUSIONS

A. Results summarization

1. From the results that we have obtained we could see that Part of Speech tags really helps to improve the accuracy of the data.

2. Of the three classifiers SVM, Logistic Regression and Naive Bayes, SVM seems to perform the best. This might be beacuse SVM handles sparse data better.

3. The generalized model seems to work on an average scale on both the tested data.

B. Limitations

1. The domain adaptation portion could yield better results when implemented with iterative method and dynamic adaptation method. This helps to achieve better results.

TABLE XXIV. CONFUSION MATRIX FOR LOGISTIC REGRESSION

	Negative	Positive
Negative	4705	4373
Positive	3250	8672

TABLE XXV. CONFUSION MATRIX FOR NAIVE BAYES

	Negative	Positive
Negative	5038	4040
Positive	5053	6869

This could not been implemented in this paper because of its high complexity involved in that.

2. The marginalized stacked denoising autoencoders (mSDA) was implemented, but results does not show any improvement with respect to this domain data.

C. Future Work

1. The neural network algorithm will be implemented and the results to be compared with the other algorithm results.

2. The domain adaptation can be tested on many other domains in order to know the model consistency. Dataset such as the twenty

3. Various other domain adaptation algorithms such as "Iterative approach", "Dynamic adaptation" can be implemented and compared.

4. Implementing model with multiclass data.

X. CONCLUSION

The final conclusion that we can make out of this project are as follows.
1. SVM is the best algorithm for text classification.
2. The challenging domain adaptation can be seen as a great opportunity in text analysis area.

ACKNOWLEDGMENT

The authors would like to thank Professor Muhammad Abdul-Mageed for his constant support and feedback throughout this paper. He has been very kind and helpful to give us valuable tips that helped us complete this paper successfully. We deeply express our appreciation to all those who have helped us in this process.

REFERENCES

[1] Mikolov, Tomas; Sutskever, Ilya; Chen, Kai; Corrado, Greg S.; Dean, Jeff (2013). Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems.

[2] Mikolov, Tomas; et al. "Efficient Estimation of Word Representations in Vector Space" (PDF). Retrieved 2015-08-14.

[3] Goldberg, Yoav; Levy, Omer. "word2vec Explained: Deriving Mikolov et al.s Negative-Sampling Word-Embedding Method" (PDF).

[4] Rehurek, Radim. Word2vec and friends (Youtube video).

[5] https://en.wikipedia.org/wiki/Word2vec

[6] http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

[7] http://levyomer.wordpress.com/2014/04/25/word2vec-explained-deriving-mikolov-et-al-s-negative-sampling-word-embedding-method/

[8] http://levyomer.wordpress.com/2014/09/10/neural-word-embeddings-as-implicit-matrix-factorization/

[9] Sentiment Analysis Using Support Vector Machine, Ms. Gaurangi Patil, Ms. Varsha Galande, Mr. Vedant Kekan3, Ms. Kalpana Dange4

[10] A Survey on Sentiment Analysis of (Product) Reviews

[11] Opinion mining and sentiment analysis by Bo Pang1 and Lillian Lee2

[12] Feature Selection and Classification Approach, Gautami Tripathi and Naganna S.

[13] http://www.statmt.org/OSMOSES/FeatureEx.pdf

[14] Bridle, John S.; Cox, Stephen J (1990). "RecNorm: Simultaneous normalisation and classification applied to speech recognition". Conference on Neural Information Processing Systems (NIPS): 234240.

[15] Ben-David, Shai; Blitzer, John; Crammer, Koby; Kulesza, Alex; Pereira, Fernando; Wortman Vaughan, Jennifer (2010). "A theory of learning from different domains". Machine Learning Journal 79 (1-2).

[16] Crammer, Koby; Kearns, Michael; Wortman, Jeniifer (2008). "Learning from Multiple Sources". Journal of Machine Learning Research 9: 17571774

[17] http://nlp.stanford.edu/software/tagger.shtml

[18] https://en.wikipedia.org/wiki/

XI. WORK BREAK DOWN

1. Conception of the project/idea: Anirudh K M with Professor's input.
2. Data Collection: Anirudh K M along with data pre processing steps.
3. Experiments: Scripts developed by Anirudh K M for data cleaning, classifier and domain adaptation.
4. Literature review reading and collection: Contributed by all.
5. Paper Writing: Done by Srijhari, Anirudh and Mohit.
6. Group Communication and arrangements: Done by Srijhari, Anirudh and Mohit.

XII. MEMBER BLURB

1. Anirudh, Kamalapuram Muralidhar
I am doing my Masters in Data Science at Indiana University Bloomington.
E-mail: anikamal@iu.edu
Github: https://github.com/anirudhkm/
Twitter: @anirudhkm (Photo in here)
Interests: Data Science, Neural Network, Data mining.

2. Srijhari Katragadda
Pursuing Data Science at Indiana University Bloomington.
E-mail: skatraga@indiana.edu
Twitter: @k_srijhari (Photo in here)
Interests: Anything that can make an impact on the world.

3. Venkata Koppu
Currently pursuing Masters in Computer Science at Indiana University Bloomington.

E-mail: vkoppu@iu.edu
Twitter: @VenkataKoppu (Photo in here)
Interests: Algorithm Design and Data Mining.

4. Mohit Galvankar
M.S. Data Science at Indiana University Bloomington.
E-mail: mgalvank@iu.edu