News Group Identification using Semi-Supervised Classification

Aniruddh Kumar Shukla Govind Shukla Oishi Chatterjee Vatsal Pandey

Otto von-Guericke University Magdeburg

Motivation and Problem Statement

The motivation for the project is to classify documents based on the content of the document . We wanted to make a Machine learning model which could classify news based on the text content. This was the reason we went ahead and chose the 20 Newsgroup Dataset which contains around 18000 newsgroups posts on 20 topics . The question which we are trying to answer are:

- Can we classify news posts based on the content?
- Can we still classify the documents if we train on a small portion of pre-labelled examples?
- How can we extract meaningful information from plain text so that we can train a classifier ?
- How good our model performs based on other available tools?

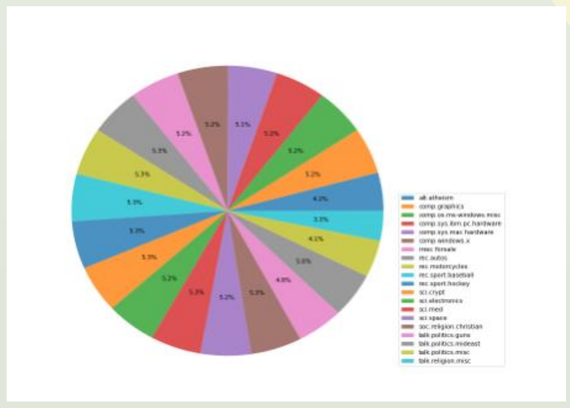
These are some of the important questions which we plan to answer in our project.

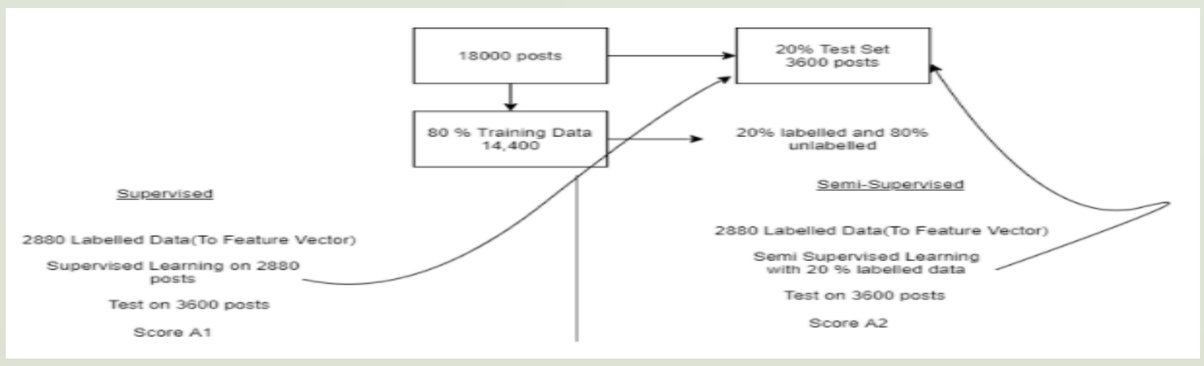
Data Set

The 20 Newsgroups data set is a collection of approximately 20,000 newsgroup documents, divided across 20 different newsgroups. Some of the newsgroups are very closely related to each other (e.g. comp.sys.ibm.pc.hardware / comp.sys.mac.hardware), while others are highly unrelated (e.g misc.forsale / soc.religion.christian). Below is a list of the 20 newsgroups, partitioned according to subject matter:
In Scikit-learn the dataset is divided into two subgroups namely train and test. The train set contains 11314 documents and the test set contains 7532 documents. Apart from the main content each file contains metadata like *'Subject'* and *'Organization'* which provides extra information. It is possible to distinguish groups by headers such as '*NNTP-Posting-Host*' and *'Distribution'* because they can appear more or less often. Another significant feature involves whether the sender is affiliated with a university, as indicated either by their headers or their signature.

The distribution of documents in different topics is displayed below. It can be determined that the distribution of documents is almost similar for all the categories.

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian





SSL Concept

The Semi-supervised learning(SSL) approach was selected so that we can make sure that we built such a system which can perform well when we have a small proportion of pre-labelled data. We plan to use an Inductive based SSL approach and one Transductive based SSL approach . The goal was to make sure we take all the factors into account and test our data on it. The SSL techniques which we have selected are:
Label Propagation & Label Spreading(Transductive based)
SemiSupervised EM Naive Bayes(Inductive based)
The plan to use transductive approach and Inductive approach for a comparison and choose the accurate model based on the results
Label Propagation & Label Spreading: Both of the methods are graph based and use a similarity matrix to label the unlabelled data .
SemiSupervised EM Naive Bayes: This is an extension of Multinomial Naive Bayes combined with an Expectation maximization approach. This we will compare with only Multinomial Naive Bayes and see how the algorithms compare.

The advantage of graph based methods is that it works on simple similarity measures and builds vertices which is an easy measure to calculate.
The disadvantage is that the method is transductive and only labels the unlabelled data in the train set. Another disadvantage is that graph based methods work well if we have an underlying network in the data . Ex: social network data .

The advantage of EM naive bayes is that it is built up on Multinomial Naive Bayes and that it is faster to compute as this is based on Naive Bayes . So this is faster to compute.
The disadvantage of EM based method is that it can be proven that EM improves the log likelihood log $p(D|\theta)$ at each iteration. However, EM only converges to a local optimum.

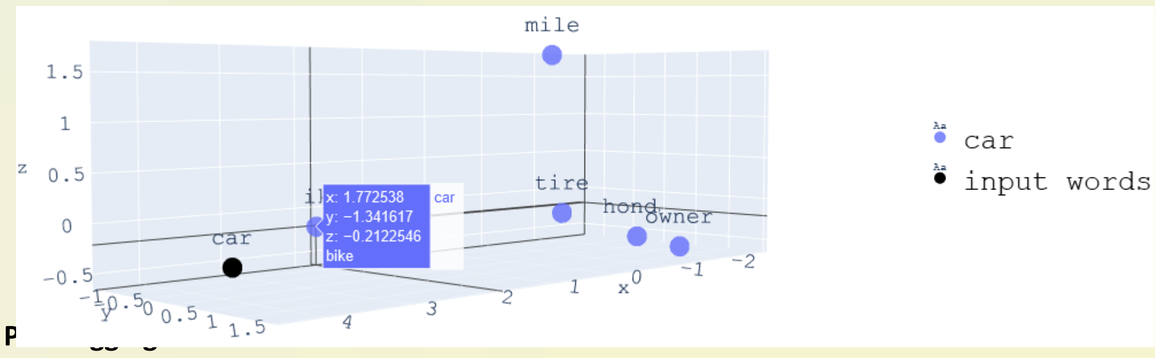
Features

Tf-Idf Vectoriser+Bag-of-Words(BOW)

After the preprocessing we will have a list of lemmatized tokens. Using this list we will create a bag of words vectors of each document using CountVectorizer(). Different versions of BOW will be created like bigrams and trigrams to check which one gives better results. Then scikit-learn tfidf vectorizer will be used to create tfidf matrix. This will result in a feature matrix.We will also try some feature selection methods to reduce the size of feature space. Maybe by using scikit-learn SelectKBest() with chi2.

Word Embeddings(Glove)

GloVe: We are using GloVe as a method of word embedding feature extraction for our data as it outperforms other models such as word analogy, word similarity, and named entity recognition. We are using SpaCy library to implement this method and sklearn.decomposition.TruncatedSVD to reduce the number of features to 50.



P

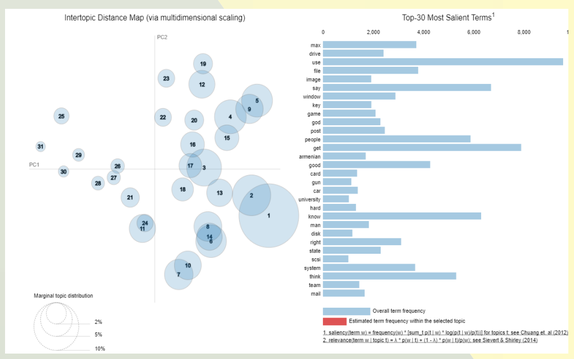
Part-of-speech (POS) tagging is a popular Natural Language Processing process which refers to categorizing words in a text (corpus) in correspondence with a particular part of speech, depending on the definition of the word and its context. We will calculate the Frequency distribution of Part of Speech Tags: noun, pronoun, verb, adverb, adjective and store them as features. We will use the TextBlob library to tokenize the text and find the POS for those individual tokens.

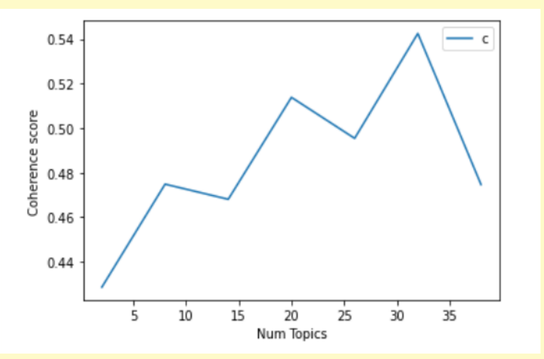
LDA

Topic modeling is a type of statistical modeling for discovering the abstract "topics" that occur in a collection of documents. Latent Dirichlet Allocation(LDA) is an example of a topic model and is used to classify text in a document to a particular topic. It builds a topic per document model and words per topic model, modeled as Dirichlet distributions.

Topics can also be generated using scikit-learn LatentDirichletAllocation. To generate corpus for model training CountVectorizer() will be used. Similar to gensim, we will generate topics and use them as features. To find optimal model grid search will be used. Performance of both the libraries will be compared and then the better model will be used in our final implementation.

Also, pyLDAvis will be used to plot the topics. This will help to find the spread of topics and also the words present in the topics.





Implementation

We used the gensim package in the first step to perform some preprocessing on the data.The data is taken from Sklearn datasets . The main steps are stripping tags,strip the numeric part in the text,removing punctuations,remove stopwords and also converted the text to lower case .This is done to make the data uniform to work with.

We have also used some sklearn functions to convert the text into vectors . The sklearn also helped us to split the data into different sets for training and testing . The pipeline used is also provided by sklearn . The classifiers used are also taken from sklearn and also used a custom library to work with EM based Naive bayes. The cross validation and feature selection is also implemented using sklearn . Furthermore we have used Sklearn metrics to perform test and selections .

How did you implement the concepts? Are there any special things that you need to take care of?

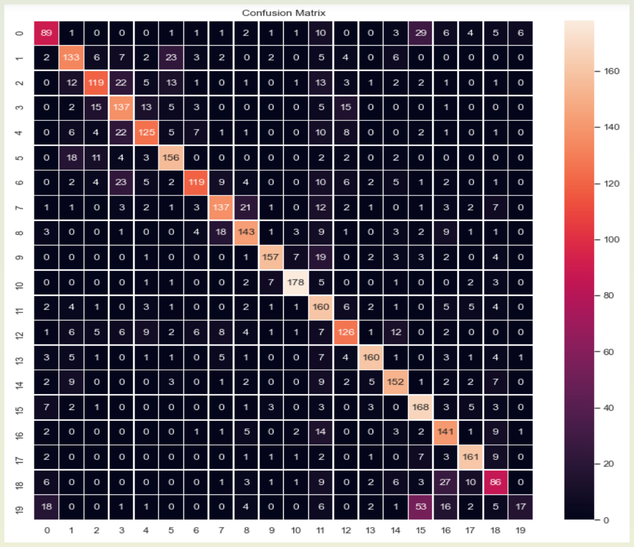
Answer TBD

There are several assumptions made . The first is that we cannot merge all the features together .The BOW and word embedding are as such that combining them would not have been meaningful . The LDA gives important topics that can also not be combined with others . Hence we have shown some features stand alone and not merged all the features together .This helped us give a good comparison among various feature set so that we can see how the features give results with SSL .

Evaluation

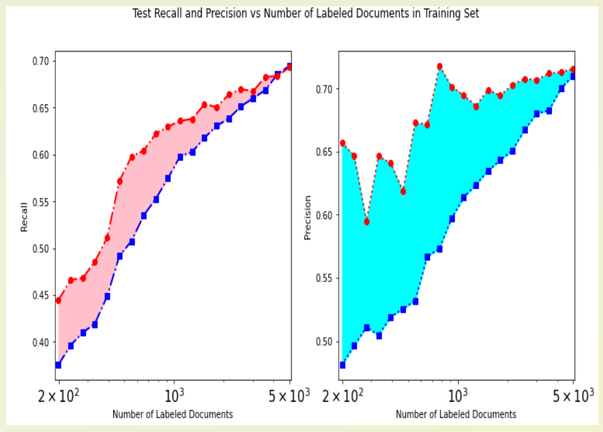
```
11314 documents
20 categories

Supervised SGDClassifier on 100% of the data:
Number of training samples: 8485
Unlabeled samples in training set: 0
Micro-averaged F1 score on test set: 0.876
----------

Supervised SGDClassifier on 20% of the training data:
Number of training samples: 1682
Unlabeled samples in training set: 0
Micro-averaged F1 score on test set: 0.777
----------

LabelSpreading on 20% of the data (rest is unlabeled):
Number of training samples: 8485
Unlabeled samples in training set: 6803
Micro-averaged F1 score on test set: 0.284
----------
```

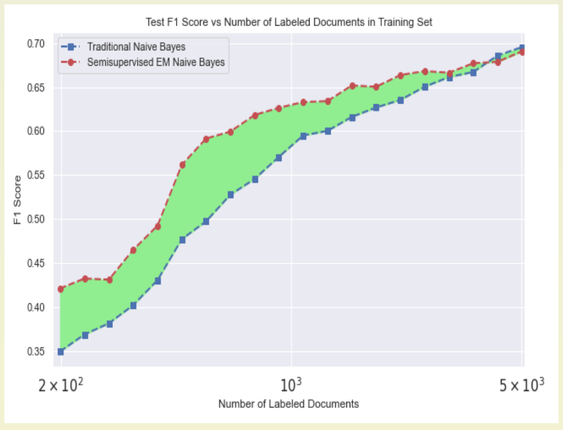
```
================================================================================
Validation:
MultinomialNB(alpha=0.01, class_prior=None, fit_prior=True)
Fold # 1
Fold # 2
Fold # 3
Fold # 4
Fold # 5
Validation time: 0.105 seconds
Average training accuracy: 0.686
Number of labeled documents:   4229
================================================================================
Validation:
MultinomialNB(alpha=0.01, class_prior=None, fit_prior=True)
Fold # 1
Fold # 2
Fold # 3
Fold # 4
Fold # 5
Validation time: 0.117 seconds
Average training accuracy: 0.699
Number of labeled documents:   5011
```

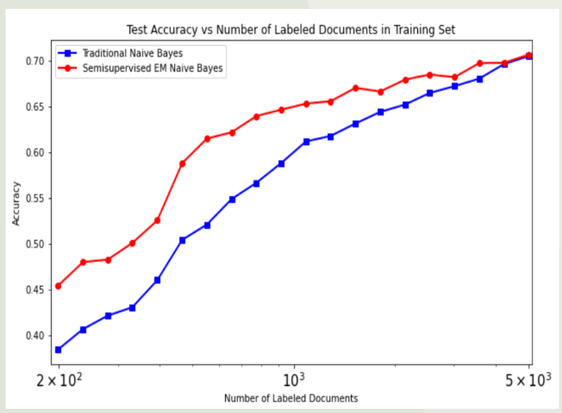


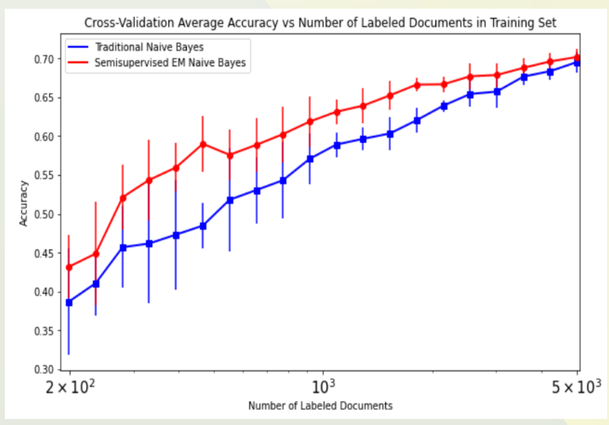
	precision	recall	f1-score	support
alt.atheism	0.64	0.56	0.60	160
comp.graphics	0.66	0.68	0.67	195
comp.os.ms-windows.misc	0.71	0.60	0.65	197
comp.sys.ibm.pc.hardware	0.61	0.70	0.65	196
comp.sys.mac.hardware	0.73	0.65	0.69	193
comp.windows.x	0.73	0.79	0.76	198
misc.forsale	0.80	0.61	0.69	195
rec.autos	0.74	0.69	0.72	198
rec.motorcycles	0.72	0.72	0.72	199
rec.sport.baseball	0.89	0.79	0.84	199
rec.sport.hockey	0.91	0.89	0.90	200
sci.crypt	0.50	0.81	0.62	198
sci.electronics	0.70	0.64	0.67	197
sci.med	0.88	0.81	0.84	198
sci.space	0.76	0.77	0.76	197
soc.religion.christian	0.61	0.84	0.71	199
talk.politics.guns	0.62	0.77	0.69	182
talk.politics.mideast	0.82	0.86	0.84	188
talk.politics.misc	0.57	0.55	0.56	155
talk.religion.misc	0.68	0.13	0.23	126
accuracy			0.71	3770
macro avg	0.72	0.69	0.69	3770
weighted avg	0.72	0.71	0.70	3770

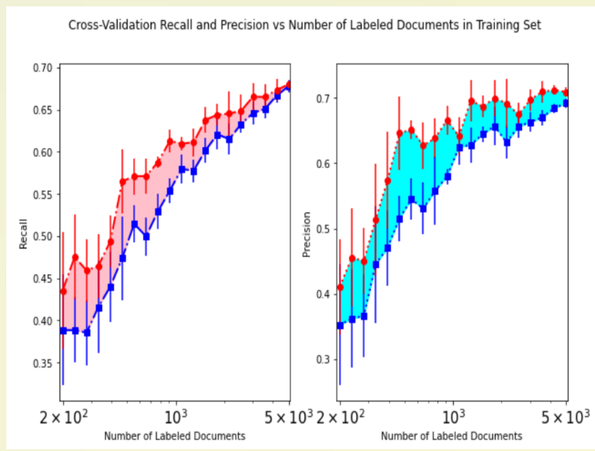
```
Supervised SGDClassifier on 100% of the data:
Number of training samples: 12626
Unlabeled samples in training set: 0
Micro-averaged F1 score on test set: 0.409
----------

Supervised SGDClassifier on 20% of the training data:
Number of training samples: 2522
Unlabeled samples in training set: 0
Micro-averaged F1 score on test set: 0.242
----------

SelfTrainingClassifier on 20% of the training data (rest is unlabeled):
Number of training samples: 12626
Unlabeled samples in training set: 10104
End of iteration 1, added 617 new labels.
End of iteration 2, added 532 new labels.
End of iteration 3, added 399 new labels.
End of iteration 4, added 33 new labels.
End of iteration 5, added 175 new labels.
End of iteration 6, added 221 new labels.
End of iteration 7, added 139 new labels.
End of iteration 8, added 86 new labels.
End of iteration 9, added 75 new labels.
End of iteration 10, added 93 new labels.
Micro-averaged F1 score on test set: 0.379
----------

LabelSpreading on 20% of the data (rest is unlabeled):
Number of training samples: 12626
Unlabeled samples in training set: 10104
Micro-averaged F1 score on test set: 0.363
----------
```

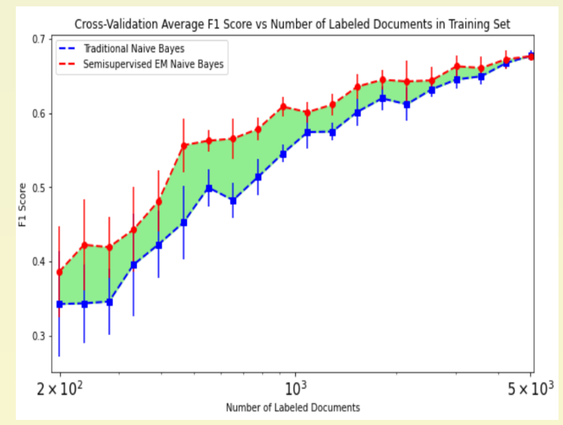






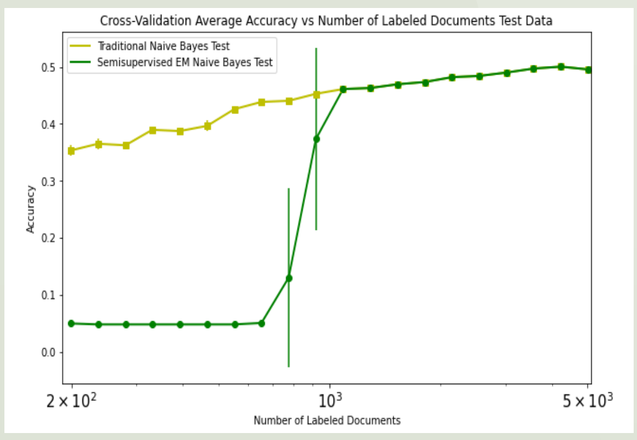


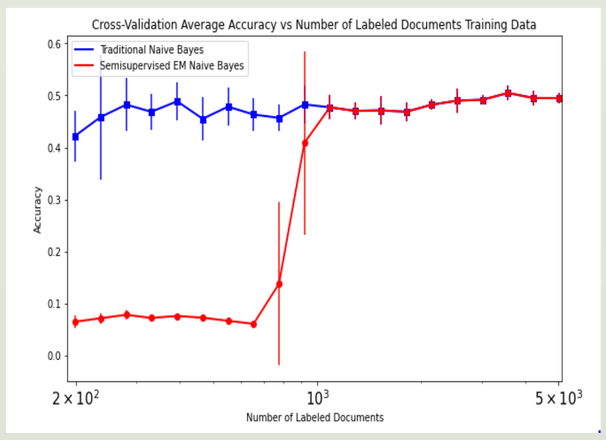




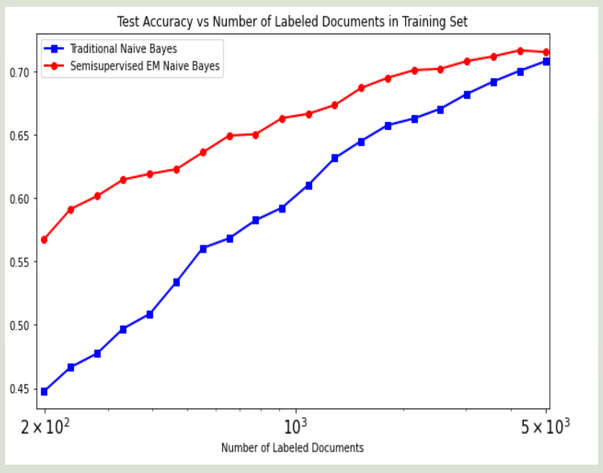
Other Generated resuts for all features

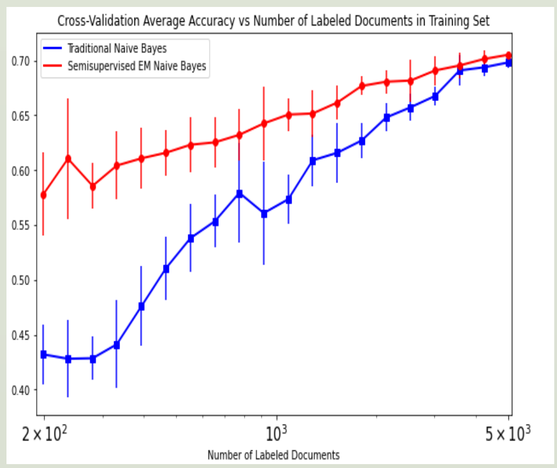
Count Vect +TF-IDF(Bag of Words)



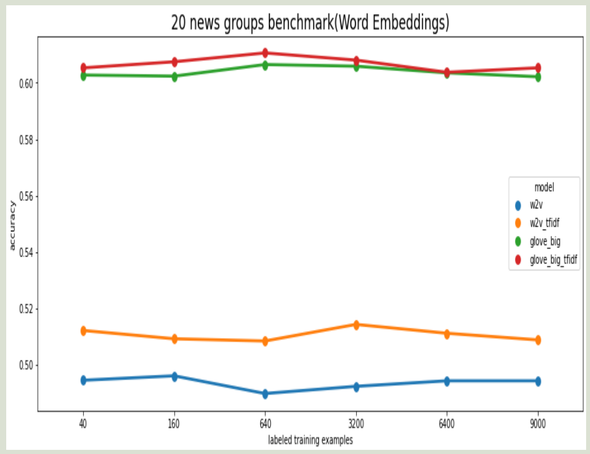


LDA





Count Vect + TF-IDF(Bag of Words) + POS Tagging



Word Embeddings

Conclusion

In our experiment we have tried to classify the 20NewsGroups data using both supervised and semi supervised classifiers. We have to extract meaningful information from the contents of 20NewsGroups data so that we can train a classifier on it. To achieve this we have extracted features using Bag of Words, TF-IDF, Parts of Speech tagging, Word Embedding (GloVe) and Topic Modelling(LDA).

Since Bag of Words sometimes fail to capture the syntactic relations between words we have used different versions of Bow like bigrams and trigrams. We have also used POS tagging to extract features based on parts of speech. We are using GloVe as a method of word embedding feature extraction for our data as it outperforms other models such as word analogy, word similarity, and named entity recognition. We were also able to extract unique topics from the data as a set of words using LDA.

We implemented Self learning with SGD classifier, Label propagation, Multinomial Naive Bayes and Naive Bayes with EM.

Comparing with SGD classifier and Naive Bayes, the SSL model had better accuracy score. But Naive Bayes with EM performed better than self training and label propagation. However, when TF-IDF and POS / TF-IDF and GloVe (features) were combined label spreading and label propagation we were getting very low accuracy scores.

We have learnt from this experiment that for text classification, even when we have less pre-labelled training data we can use semi supervised classifiers like Naive Bayes with EM and Label Propagation to classify an unlabelled instance with an average accuracy of 70%. A ML model is only as good as the features it learns. So the performance of our models could improve if we had domain expertise which could help us do a better feature extraction.

Feature Extraction	Supervised Accuracy	Semi-Supervised Accuracy
Count Vect + TF-IDF(Bag of Words) + POS Tagging	69.8%	70.5%
LDA	40.8%	40.8%
Count Vect +TF-IDF(Bag of Words)	70.5%	70.7%
Word Embedding	61%	