Classification of Textual Data by Applying Five Machine Learning Models

Donghan Liu, Dingtao Hu, Jifeng Wang
McGill University COMP551 Mini project2

Abstract— **In this mini project, we were tasked to train multiple classification models(Logistic Regression, Decision Tree Classifier, Support Vector Machine, Ada boost, Random Forest and two additional models Naive Bayes, K-Nearest Neighbors) from the scikit-learn package to perform sentiment analysis on the embedded training subset in sklearn.datasets and IMDB movie reviews dataset. Additionally, we studied the performance of the above models under the effect of different hyperparameters for each data set. In the field of further exploration, we applied different CountVectorizer() parameters on the best performed model obtained from the aforementioned steps to achieve better performance. We extracted the occurrences of words, and transformed text word counts to TF*IDF (Term Frequency Inverse Document Frequency).It can be found that our best performing model is SVM that tuned with various hyperparameters, using Grid Search Cross Validation technique and Count Occurrence + TF*IDF feature extraction as the standard.The best performed model can achieve a classification accuracy of roughly 69.9196% on the test 20 newsgroup dataset and 89.736% on IMDB data set which has exceeds TA's benchmark!**

I. Introduction

The project focuses on the following tasks: implementing the required 5 classifiers and 2 additional models imported from SciKit-Learn on two data sets, applying Grid Search Cross Validation to select the best parameter for each model, trying different CountVectorizer() parameters and NLP tools from nltk to improve the performance of the best model selected from the above steps.

The Twenty Newsgroups data set is a built-in default dataset from scikit-learn which contains the news from 20 newsgroup in a format of natural language and the other data set consists of English movie reviews from IMDB in text format in which the training and test sets each consist of 25000 data points. We cleaned the data sets by normalization or standardization.The NLP preprocessing process (stemmer/lemmatizer tokenizer, frequent words removal, rare words removal etc.) helps to improve the accuracy by 2-3 points. But what is surprisingly found is that the standardization from standscaler() decreases the accuracy for most models by approximately 4-5%.

The crucial findings are SVM provides more accurate classifications than other models in Twenty Newsgroups data set with an accuracy of 69.9196% in IMDB data set with 89.736% . While DT performs worst among all required models in both data sets and fall behind other models with around 3-4points and solely achieves 31.837% in 20news. KNN obtains the similar results as DT. It can be noted the Grid Search cross validation technique took much computation time on training and selecting, hundreds of seconds normally, but allowed for a better prediction accuracy and relatively optimal parameters on the validation set.

When testing our SVM model with only a word count feature and normalizer, we had an accuracy of 87% on the validation set for 20news. In the optimization process, the combination of removal, lower text, wordtokeniz and WordNetLemmatizer in preprocessing procedure improves the accuracy of SVM by 2 points for two data sets.

II. Related Work

Many machine learning techniques have been successfully used in for text classification. Some examples include: Logistic regression [1], Decision tree [2], Supporting vector machines [3] Ada boost [4] and Random forest [1]. Applications of text classification are often related to scientific or technical domains where the number of instances is too large for human classification and there is a need simplifying search.

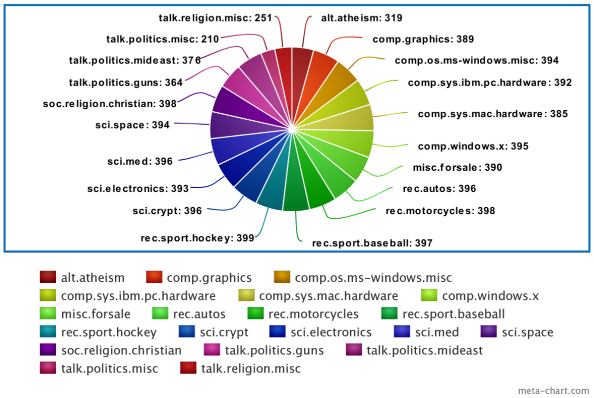
Multi-class classification problem is a field by applying data mining techniques that assigns categories to a collection of data in order to aid in more accurate predictions and analysis. Classification is a data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data.The task begins with a data set in which the class assignments are known. The aim of the classification algorithm is to discover how that set of attributes reaches its conclusion .

The findings from the reference [1] indicate that the Logistic Regression multi-class classification method for product-reviews has achieved the highest (min 0.3243, max 0.5850) classification accuracy in comparison with Naive bayes, Random forest, Decision tree, and Support vector machines classification methods. On the contrary, Decision tree has got the lowest average accuracy values.

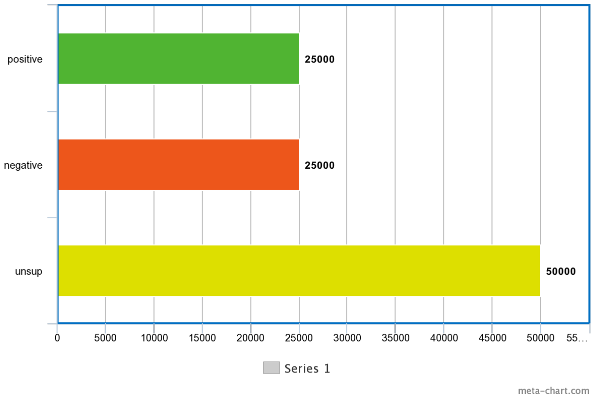
III. Dataset and Setup

A. Dataset Description

1) 20 news group dataset: The 20 news group dataset is a collection of approximately 19,000 newsgroup documents that partitioned into 20 distinctive newsgroups, each corresponding to a different topic under one of six subjects, roughly evenly as presented as a pie chart below.



2) IMDB reviews: The IMDB review datadet is a binary classification set with 100,000 reviews, 25,000 reviews for training and 25,000 reviews for testing, where exactly half of the reviews are positive and the other half is negative, and 50,000 unlabeled reviews neither positive nor negative as shown below



B. Dataset Preprocessing

For cleaning of both dataset, we remove all non-words from the dataset, such as ”////”, transform the review in lower case for better classification, remove all stop words, and perform stemming/lemmating for subsequent optimization on the model. In addition, we check and correct the spelling in the 20 newsgroup data for better learning performance.

IV. Proposed approach

A. Models Description

In the project, the models Logistic Regression, Decision Tree Classifier,Support Vector Machine, Ada boost, Random Forest and two additional models Naive Bayes, k-nearest neighbors are provided from SciKit-Learn packages [5].

1) Logistic Regression: Logistic Regression is a Machine Learning algorithm which is used for the classification problems, it is a predictive analysis algorithm and based on the concept of probability.The cost function for LR can be defined as the Sigmoid function ($g(z) = \frac{1}{1+exp(-z)}$) which is for mapping predicted values to probabilities.The hypothesis of logistic regression ($h_\beta(x) = g(\beta^T x)$) tends it to limit the cost function between 0 and 1 where the cost function is $cost(h(\theta)X, Y) = -log(h(\theta)X)$ when Y=1 or $-log(1 - h(\theta)X)$ when Y=0. It has been reported by several practitioners [6] that Logistic Regression works best with fairly large dimensions, which is the case for text data.

2) Decision Tree: Decision Tree is a tree-like model. Leaves are classes, and inner nodes are binary classification rules. The learning process partitions the training data into separate classes using the features provided. The motivations to use it are that it is easily interpretable, and the mathematics are simple. However, it has the lowest accuracy score when compared to the logistic regression and linear SVM and is sensitive to the training data.

3) Support Vector Machine algorithm: Support vector machine algorithm aims to find a hyper-plane in an N-dimensional space(N the number of features) that distinctly classifies the data points and the separating hyper-plane should equally split the maximum gaps between different classes in the data set. The margin distance is determined by its support vectors. The model attempts to maximize the margin to have maximum confidence in separating points close to the hyper-plane. The loss function that helps maximize the margin is hinge loss:

$$\text{c(x,y,f(x))=} \begin{cases} 0, & \text{if } y * f(x) \geq 1 \\ 1 - y * f(x), & \text{otherwise} \end{cases}$$

Support vector machine is highly preferred by many as it produces significant accuracy with less computational cost.

4) Ada Boost: Ada-boost is a kind of ensemble classifier which combines multiple classifiers with selection of training set at every iteration and assigning right amount of weight in final voting. It retrains the algorithm iteratively by choosing the training set based on accuracy of previous training and the weight-age of each trained classifier at any iteration depends on the accuracy achieved.

5) Random Forest: Random forest consists of a large number of individual decision trees that operate as an ensemble.Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes the models prediction. The application of RF is motivated by the fact that a large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.

6) Multinomial Naive Bayes: Multinomial Naive Bayes implements the naive Bayes algorithm for multinomially distributed data, and is one of the two classic naive Bayes variants used in text classification (where the data are typically represented as word vector counts, although tf-idf vectors are also known to work well in practice). The distribution is parametrized by vectors $\theta_y = (\theta_{y1}, ..., \theta_{yn})$ for each class , where is the number of features (in text classification, the size of the vocabulary) and θ_{yi} is the probability $P(x_i|\ y)$ of feature i appearing in a sample belonging to class y.The parameters θ_{yi} is estimated by a smoothed version of maximum likelihood, i.e. relative frequency counting: $\theta_{yi} = \frac{N_{yi}+\alpha}{N_y+n\alpha}$. [7]

7) K-Nearest Neighbor: K-nearest neighbor method was used as a baseline method to test classification. The distance metric was defined as the euclidean distance. The key issue with KNN is choosing the number of neighbors. Cross validation was implemented and used on the training dataset to choose the number of neighbors among the set 1, 5, 10, 15,20 which produces the smallest misclassification error. For each run, the test data was classified based on its the distance to the k-closest instances in the remaining training data.

B. Hyper-parameters Tuning

Hyper-parameters are parameters that are indirectly learnt within estimators. In scikit-learn they are passed as arguments to the constructor of the estimator classes. In this project, we searched the hyper-parameter space for the highest grid search cross validation score. The grid search provided by GridSearchCV exhaustively generates candidates from a grid of parameter values specified with the param_grid parameter which contributes to find the best parameter for each model. The GridSearchCV can evaluate all combinations we define. For example, we defined the param_grid as 'rf_n_estimators':[1000,5000],'rf_max_features':[10, 50], 'rf_max_depth':[300,1000],'rf_n_jobs':[11], there are 2*2*2*1*5=40 combinations of settings. We then fitted the model, displayed the best hyper-parameters, and evaluated performance. Instead of assigning each paramters for different parameters, we built a models() function to automatically call the parameters and pipelines for each model.We used l2 regularization strategy by default. Furthermore, the Grid Search CV handles the splitting for train/validation.

C. Feature Extraction

We used SciKit-Learns CountVectorizer for converting a collection of text documents to a matrix of token counts and then transformed the count matrix to a normalized Term Frequency Inverse Document Frequency (tf-idf) or tf representationits by the attached TfidfTransformer. for transforming the extracted ngrams into a TF*IDF feature. The IDF takes into account the words which appear more commonly in all abstracts (such as a,and, the) and can reduce their impact in the vector produced.

Before processing the abstract into the TFIDF table, we first removed all non-words like the numbers, punctuation and stopwords (i.e. a, as, the, to etc.) from them transformed reviews to lowercase. Additionally, Stemming and Lemmatization procedure can pick up stems and remove all inflectional endings/suffix from the words to return the words to the base form, e.g.: studies, studied and studying will be stemmed/lemmatized to study. After the above preprocessing steps, using the scikit-learn TfIdfVectorizer package, we vectorized the entire training and test set separately using the TFIDF metric. The vectorizer is tweaked to extract one gram, bi-gram and tri-gram words, which extracts features based on one word, two consecutive words and three consecutive word combinations.The n-grams approach in vectorization is found to work considerably better, because combinations of consecutive words provide contextual information pertaining to the subject of the abstract. Each model automatically calls the feature extraction pipelines by the models() function.

D. Algorithm Selection and Implementation

All 7 models are implemented under the above procedure while SVM archives the best performance in both two data sets among hundreds models by hyper-parameter tuning. SVM is selected to be applied for the following optimization process by adjusting CountVectorizer() parameters. After another grid search,for both IMDB and 20 Newsgroup data, LinearSVC() reaches the highest test accuracy across all models.

E. Model Optimization

Stemming is the process of producing morphological variants of a root/base word. Stemming programs are commonly referred to as stemming algorithms or stemmers. A stemming algorithm reduces the words chocolates, chocolatey, choco to the root word, chocolate. Stemming is desirable as it may reduce redundancy as most of the time the word stem and their inflected/derived words mean the same [8]. We attached PorterStemmer from NLTK after all preprocessing steps in model SVM.

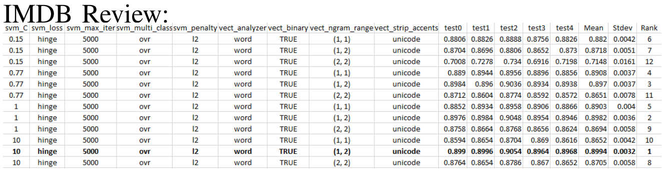
Lemmatization is the process of converting a word to its base form. The difference between stemming and lemmatization is, lemmatization considers the context and converts the word to its meaningful base form, whereas stemming just removes the last few characters, often leading to incorrect meanings and spelling errors.Wordnet is an large, freely and publicly available lexical database for the English language aiming to establish structured semantic relationships between words. It offers lemmatization capabilities as well and is one of the earliest and most commonly used lemmatizers [9].We replaced PorterStemmer by WordNetLemmatizer from NLTK in model SVM and compared the two results.

V. Results

A. Best Parameters using Grid Search CV

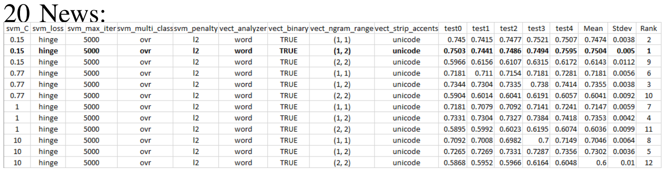
This section is implemented by applying Grid Search to select the best parameter which obtains the highest mean test score and relatively lowest std test score. We attached the graph concentrating on showing the parameter selection of model SVM on both data sets. The results for other 6 models are in the attached txt files.

IMDB Review:



svm_C	svm_loss	svm_max_iter	svm_multi_class	svm_penalty	vect_analyzer	vect_binary	vect_ngram_range	vect_strip_accents	test0	test1	test2	test3	test4	Mean	Stdev	Rank
0.15	hinge	5000	ovr	l2	word	TRUE	(1, 1)	unicode	0.8806	0.8826	0.8888	0.8756	0.8826	0.882	0.0042	6
0.15	hinge	5000	ovr	l2	word	TRUE	(1, 2)	unicode	0.8704	0.8696	0.8806	0.8652	0.873	0.8718	0.0051	7
0.15	hinge	5000	ovr	l2	word	TRUE	(2, 2)	unicode	0.7008	0.7278	0.734	0.6916	0.7198	0.7148	0.0161	12
0.77	hinge	5000	ovr	l2	word	TRUE	(1, 1)	unicode	0.889	0.8944	0.8956	0.8896	0.8856	0.8908	0.0037	4
0.77	hinge	5000	ovr	l2	word	TRUE	(1, 2)	unicode	0.8984	0.896	0.9036	0.8934	0.8938	0.897	0.0037	3
0.77	hinge	5000	ovr	l2	word	TRUE	(2, 2)	unicode	0.8712	0.8604	0.8774	0.8592	0.8572	0.8651	0.0078	11
1	hinge	5000	ovr	l2	word	TRUE	(1, 1)	unicode	0.8852	0.8934	0.8958	0.8906	0.8866	0.8903	0.004	5
1	hinge	5000	ovr	l2	word	TRUE	(1, 2)	unicode	0.8976	0.8984	0.9048	0.8954	0.8946	0.8982	0.0036	2
1	hinge	5000	ovr	l2	word	TRUE	(2, 2)	unicode	0.8758	0.8664	0.8768	0.8656	0.8624	0.8694	0.0058	9
10	hinge	5000	ovr	l2	word	TRUE	(1, 1)	unicode	0.8594	0.8654	0.8704	0.869	0.8616	0.8652	0.0042	10
10	**hinge**	**5000**	**ovr**	**l2**	**word**	**TRUE**	**(1, 2)**	**unicode**	**0.899**	**0.8996**	**0.9054**	**0.8964**	**0.8968**	**0.8994**	**0.0032**	**1**
10	hinge	5000	ovr	l2	word	TRUE	(2, 2)	unicode	0.8764	0.8654	0.8786	0.867	0.8652	0.8705	0.0058	8

20 News:

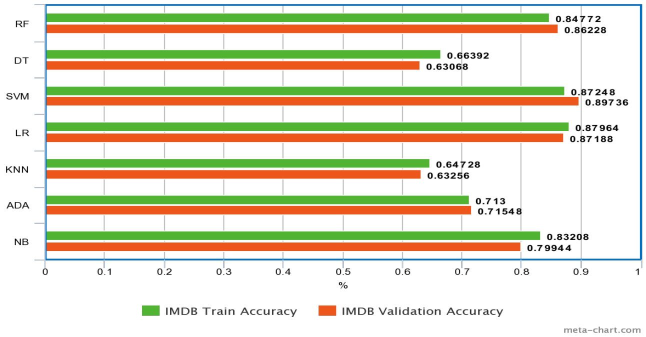


svm_C	svm_loss	svm_max_iter	svm_multi_class	svm_penalty	vect_analyzer	vect_binary	vect_ngram_range	vect_strip_accents	test0	test1	test2	test3	test4	Mean	Stdev	Rank
0.15	hinge	5000	ovr	l2	word	TRUE	(1, 1)	unicode	0.745	0.7415	0.7477	0.7521	0.7507	0.7474	0.0038	2
0.15	**hinge**	**5000**	**ovr**	**l2**	**word**	**TRUE**	**(1, 2)**	**unicode**	**0.7503**	**0.7441**	**0.7486**	**0.7494**	**0.7595**	**0.7504**	**0.005**	**1**
0.15	hinge	5000	ovr	l2	word	TRUE	(2, 2)	unicode	0.5966	0.6156	0.6107	0.6315	0.6172	0.6143	0.0112	9
0.77	hinge	5000	ovr	l2	word	TRUE	(1, 1)	unicode	0.7181	0.711	0.7154	0.7181	0.7281	0.7181	0.0056	6
0.77	hinge	5000	ovr	l2	word	TRUE	(1, 2)	unicode	0.7344	0.7304	0.7335	0.738	0.7414	0.7355	0.0038	3
0.77	hinge	5000	ovr	l2	word	TRUE	(2, 2)	unicode	0.5904	0.6014	0.6041	0.6191	0.6057	0.6041	0.0092	10
1	hinge	5000	ovr	l2	word	TRUE	(1, 1)	unicode	0.7181	0.7079	0.7092	0.7141	0.7241	0.7147	0.0059	7
1	hinge	5000	ovr	l2	word	TRUE	(1, 2)	unicode	0.7331	0.7304	0.7327	0.7384	0.7418	0.7353	0.0042	4
1	hinge	5000	ovr	l2	word	TRUE	(2, 2)	unicode	0.5895	0.5992	0.6023	0.6195	0.6074	0.6036	0.0099	11
10	hinge	5000	ovr	l2	word	TRUE	(1, 1)	unicode	0.7092	0.7008	0.6982	0.7	0.7149	0.7046	0.0064	8
10	hinge	5000	ovr	l2	word	TRUE	(1, 2)	unicode	0.7265	0.7269	0.7331	0.7287	0.7356	0.7302	0.0036	5
10	hinge	5000	ovr	l2	word	TRUE	(2, 2)	unicode	0.5868	0.5952	0.5966	0.6164	0.6048	0.6	0.01	12

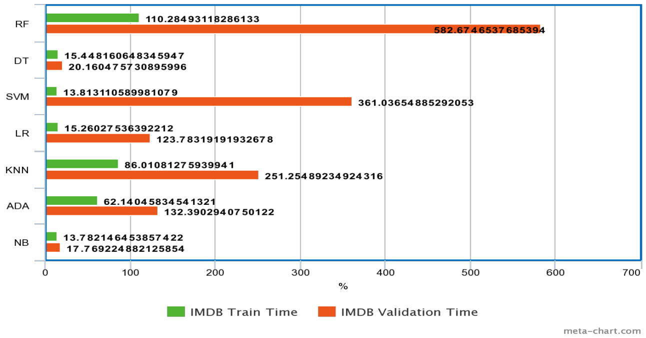
The row where the best parameter located is in bold. For example, in IMDB, the best parameter is 'svm_C': 10, 'svm_loss': 'hinge', 'svm_max_iter': 5000, 'svm_multi_class': 'ovr', 'svm_penalty': 'l2' It can be seen that there are 12 different parameters compared with each other and they are tested for five times for each group. The parameter with the relatively largest mean and relatively smallest std is regarded as the best. Additionally, the rank of all 12 parameters is shown in the last column.

B. Comparison between Training Accuracy and Test Accuracy

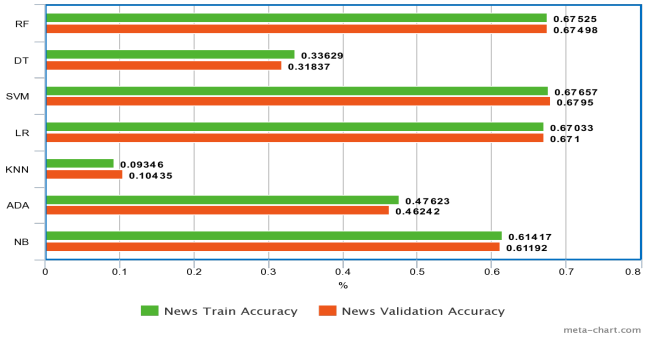
The data load of train set is 5 times larger than that of test set so we compared train and validation for both data sets and the following is the result.
IMDB Review(accuracy):



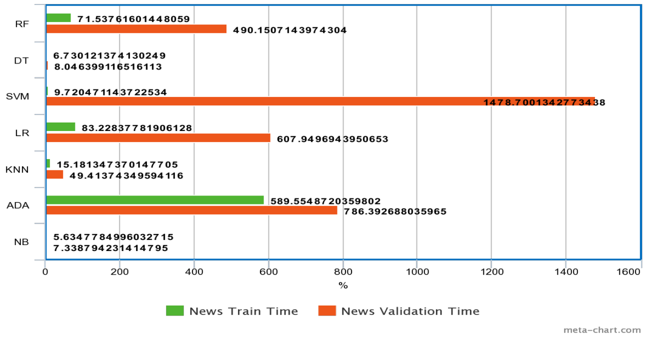
IMDB Review(time):



20 News(accuracy):



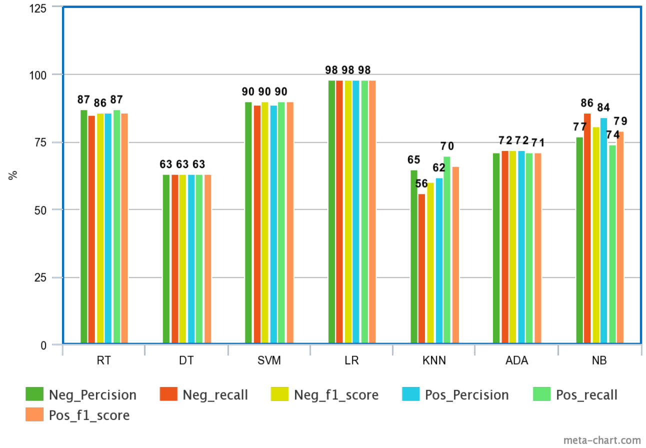
20 News(time):



From the histogram graph of accuracy we can find the highly similarity between the accuracy of different models on two data sets. For 20news data set, all models except KNN archives slightly higher training accuracy than validation which refers to the fact that the models have not yet over-learned the training dataset, showing comparable skill on both datasets. For IMDB data set, RT and SVM's validation obtains a slightly higher accuracy than train set within the range of 0.5%. The phenomenon stands for the fact that they are overfitting at some extent. The fact that only l2 regularization applied on SVM may result in this phenomenon. The reducing overfitting method (combining l1) should be focused in the further exploration.

C. Performance of Different Model

We performed seven training models on the dataset. The comparison of performance of different model including test accuracy and running time is listed below: IMDB Review:

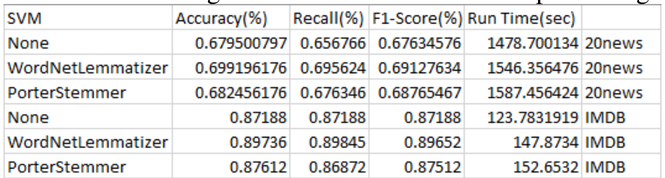


20 News: see appendix
SVM attains the highest accuracy among all 7 models for both two data sets before stemming and lemmatization. LR archives almost the same high accuracy as SVM. Model DT and KNN performs worst in both data sets which is less than 0.65 in IMDB and KNN even reaches down to 0.104 in 20news. Random Forest and Ada-Boost, both derived from DT, obtains 10%-20% higher than their parent. However, DT and KNN are extremely time-saving than other models while they are finished within 6-50 seconds, but other models consumes hundreds of seconds due to Grid Search CV mainly.
It is also noticed that LR performs better in the field of precision, recall and f1 score than the best model SVM.

With high precision and low accuracy, each value will be off by a similar amount. With high accuracy and low precision, each value is closer to the true or expected value, but repeatability suffers[10].Due to this, it can be concluded that LR lost to SVM by larger gap to the expected value which is pursued in this project.

D. *Best Model and Different Tokenizers*

SVM is selected as the best model on two data sets for further improvement on performance. We applied two NLP tools Stemming and and lemmatization from NLTK for this section. The goal of both stemming and lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. The following form shows the effect of the processing.



SVM	Accuracy(%)	Recall(%)	F1-Score(%)	Run Time(sec)	
None	0.679500797	0.656766	0.67634576	1478.700134	20news
WordNetLemmatizer	0.699196176	0.695624	0.69127634	1546.356476	20news
PorterStemmer	0.682456176	0.676346	0.68765467	1587.456424	20news
None	0.87188	0.87188	0.87188	123.7831919	IMDB
WordNetLemmatizer	0.89736	0.89845	0.89652	147.8734	IMDB
PorterStemmer	0.87612	0.86872	0.87512	152.6532	IMDB

SVM+OccurenceCount+TF*IDF is labeled as ’None’ in the form and ’WordNetLemmatizer’ refers to SVM+OccurenceCount+TF*IDF+WordNetLemmatizer and etc. At a quick glance, WordNetLemmatizer archives a relatively significant increase on accuracy with 2 points on both data sets comparing to PorterStemmer solely improves 0.4-0.6 points. We also notice that there isnt any significant difference in the accuracy, recall and f1-score , therefore we can conclude that our classifier isnt biased towards making either false positives or false negatives.

By far, we used the pipeline of NLP preprocesing techniques, machine learning classification model, grid search the best hyperparameters, and model selection to help our tuning and fitting. Both datasets attain the highest accuracy on SVM model, though their parameters are slightly different.

VI. Discussion and Conclusion

To conclude, SVM outperforms the all other learning algorithms such as Decision Tree, Logistic Regression, Adaboost, Random Forest, KNN and Multinomial Naive Bayes in the field of accuracy with 69.9196% on the test 20news data set and 89.736% on IMDB data set which has exceeds TA’s benchmark! While Multinomial Naive Bayes (the model we added) archives the shortest running time for both two datasets (7.3387s for 20news and 17.7692 for IMDB). The improvement of 2% on SVM in both two datasets is obtained by adding WordNetLemmatizer in the preprocessing combination– remove non-words, lower text, word_tokenize, WordNetLemmatizer, remove most freq words, remove least freq words which has a better effect than PorterStemmer.

The further research direction will concentrate on trying different feature extraction methods or the combinations of them to pursue a better performance in accuracy. What we used for this project is CountVectorizer()+TF*IDF, it is expected to explore other methods like OpinionLexiconCount, BinaryOccurence and their integration with textblob tokenizer, stemming tokenizer and nltk.word tokenize etc. It is also noted that Standardization applied for cleaning the data results in a decrease on accuracy. The significance to this fact worths more research. Additionally, the methods to reduce overfitting such as constraining model complexity and other regularization measures are considered to be explored.

VII. Statement of Contribution

Donghan Liu: Worked on all of the code and most of code design Dingtao Hu: Worked on most of the report and part of code design Jifeng Wang: Worked on data analysis and part of the report

References

[1] PRANCKEVIIUS.T and MARCINKEVIIUS.V (2017), Comparison of NaveBayes, Random Forest, Decision Tree, Support Vector Machines,and Logistic Regression Classifiers forText ReviewsClassification, Baltic J. Modern Computing, Vol. 5(2017), No. 2, 221-232

[2] Noormanshah.W and Nohuddin.P (2018), Document Categorization Using Decision Tree: Preliminary Study, International Journal of Engineering and Technology 7(4.34):437-440

[3] V. K. Bhalla and N. Kumar, An efficient scheme for automatic web pages categorization using the support vector machine, (in English), New Review of Hypermedia and Multimedia, Article vol. 22, no. 3, pp. 223-242, Sep 2016.

[4] Tu. C and Xu. B (2018), The Application of the AdaBoost Algorithm in the Text Classification.

[5] F. Pedregosa and G. Varoquaux, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research, vol. 12, pp. 28252830, 2011.

[6] Classifying text with bag-of-words: a tutorial. http://fastml.com/classifying-text-with-bag-of-words-a-tutorial/

[7] https://scikit-learn.org/stable/modules/naivebayes.html

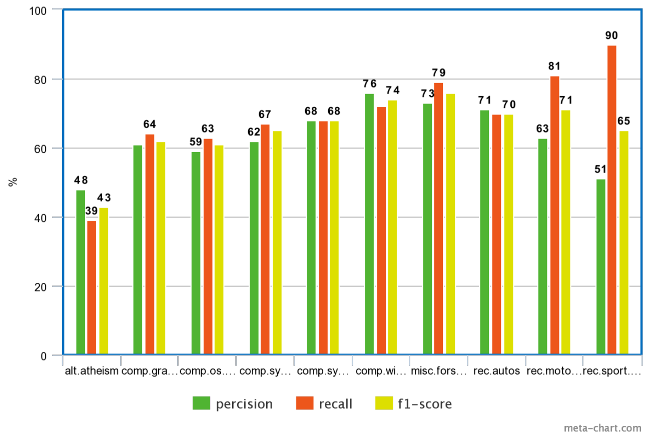
[8] https://tartarus.org/martin/PorterStemmer/

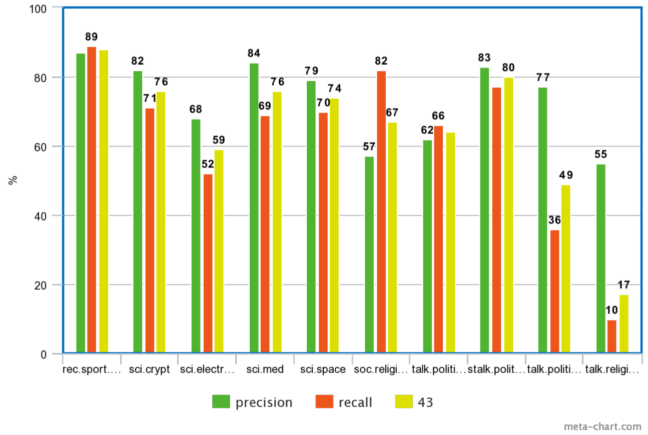
[9] Prabhakaran. S (2018), Lemmatization Approaches with Examples in Python

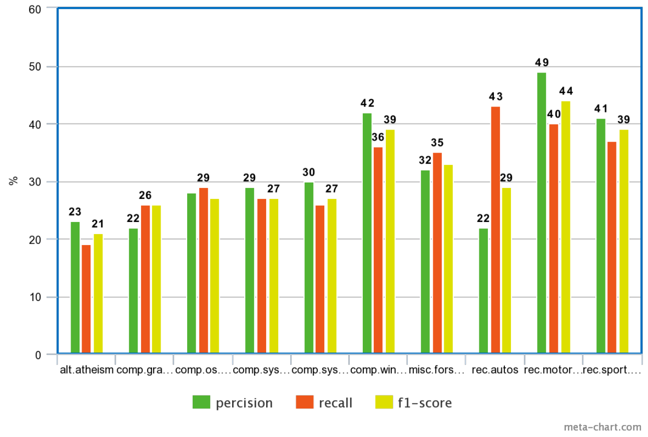
[10] The Difference Between Precision And Accuracy. https://toolguyd.com/the-difference-between-accuracy-and-precision/

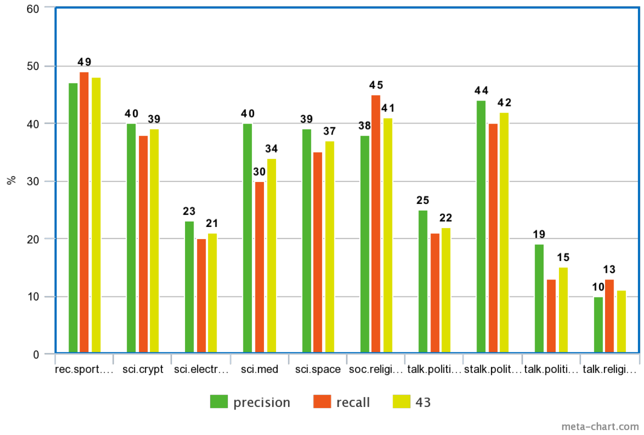
VIII. Appendix

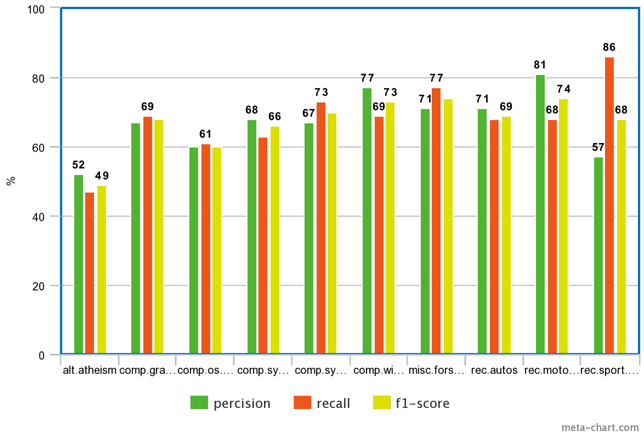
Histograms of percision, recall and f1-score to 20 divided groups in 20news data set.

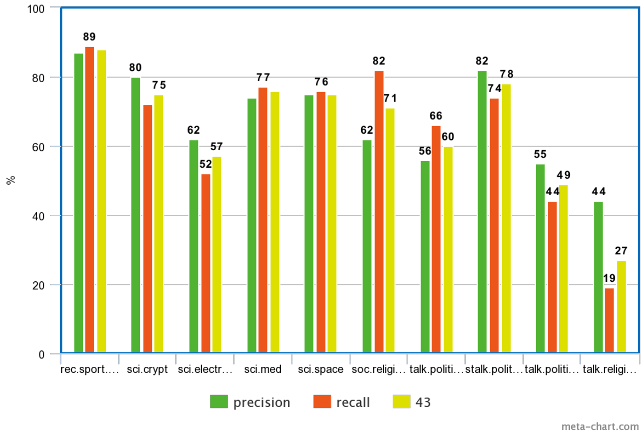


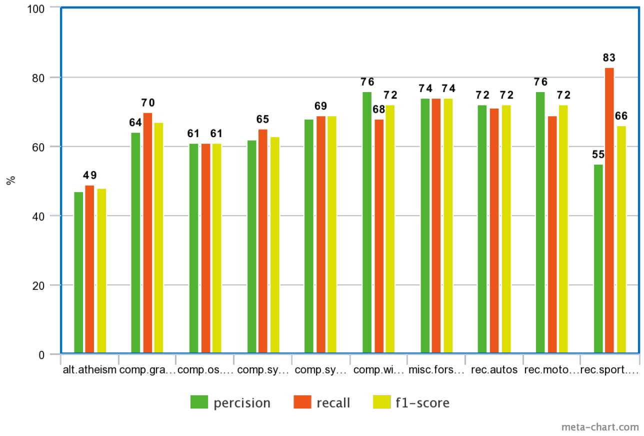


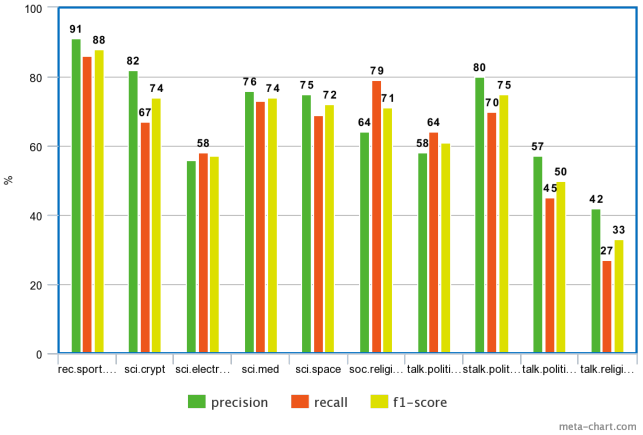


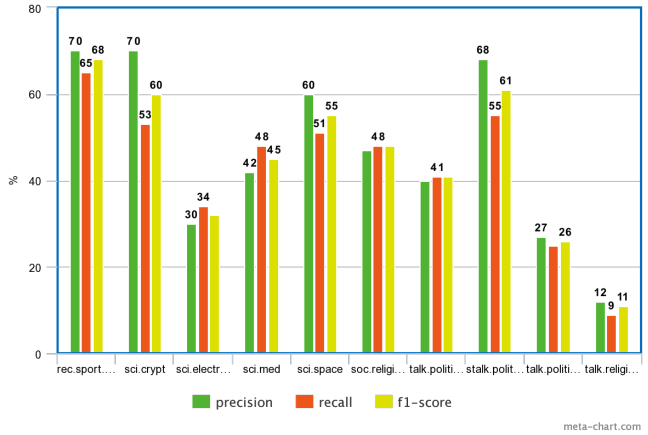


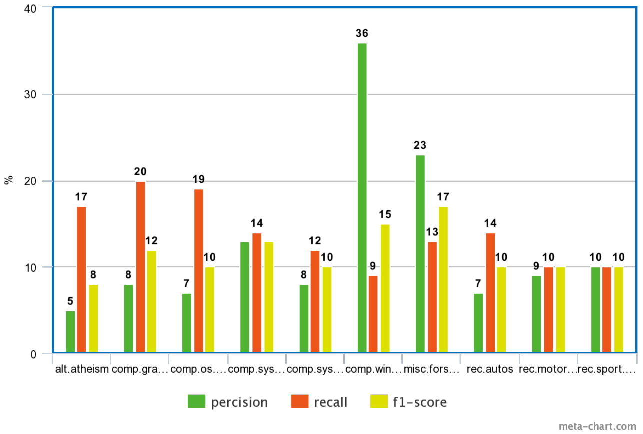


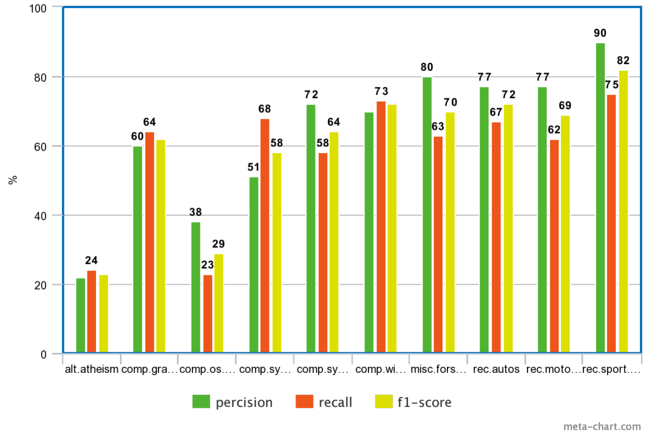


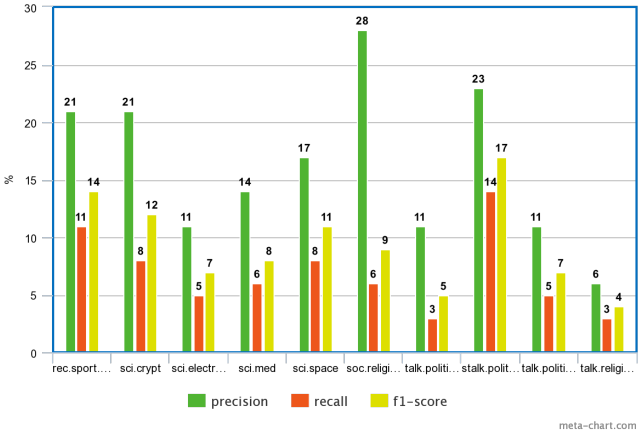


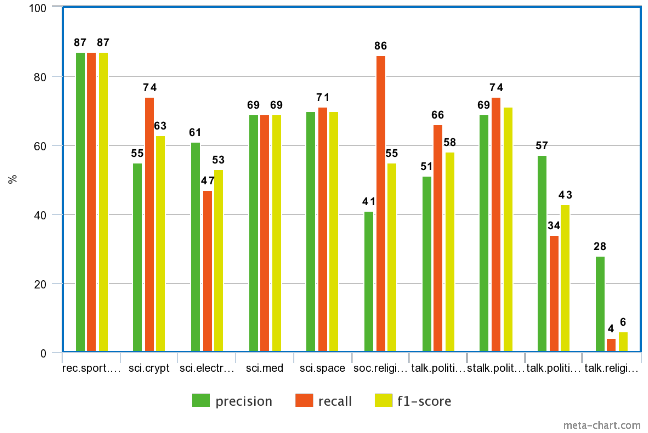

100
80
60
40
20
0
%
91
88
82
67
74
58
76
74
75
72
64
79
71
58
64
80
70
75
57
45
50
42
27
33
rec.sport.…
sci.crypt
sci.electr…
sci.med
sci.space
soc.religi…
talk.politi…
stalk.polit…
talk.politi…
talk.religi…
precision
recall
f1-score
meta-chart.com

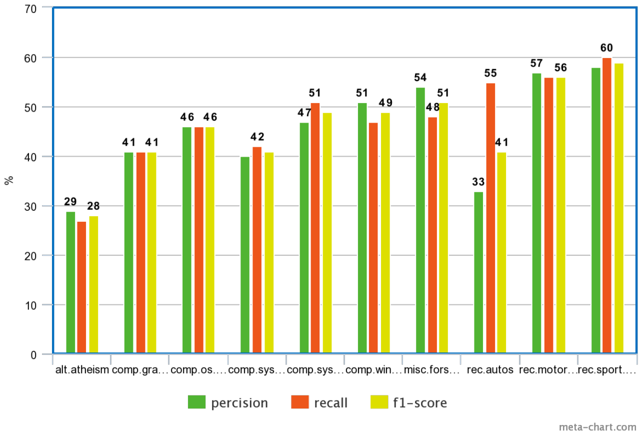

80
60
40
20
0
%
70
65
68
70
53
60
30
34
42
48
45
60
51
55
48
41
68
55
61
27
26
12
9
11
rec.sport.…
sci.crypt
sci.electr…
sci.med
sci.space
soc.religi…
talk.politi…
stalk.polit…
talk.politi…
talk.religi…
precision
recall
f1-score
meta-chart.com


40
30
20
10
0
%
17
5
8
20
8
12
19
7
10
14
12
8
10
36
9
15
23
13
17
14
7
10
9
10
10
10
alt.atheism
comp.gra…
comp.os.…
comp.sys…
comp.sys…
comp.win…
misc.fors…
rec.autos
rec.motor…
rec.sport.…
percision
recall
f1-score
meta-chart.com


100
80
60
40
20
0
%
24
60
64
38
23
29
68
51
58
72
58
64
73
80
63
70
77
67
72
77
62
69
90
75
82
alt.atheism
comp.gra…
comp.os.…
comp.sy…
comp.sy…
comp.wi…
misc.fors…
rec.autos
rec.moto…
rec.sport.…
percision
recall
f1-score
meta-chart.com


30
25
20
15
10
5
0
%
21
11
14
21
8
12
11
5
7
14
6
8
17
8
11
28
6
9
11
3
5
23
14
17
11
5
7
6
3
4
rec.sport.…
sci.crypt
sci.electr…
sci.med
sci.space
soc.religi…
talk.politi…
stalk.polit…
talk.politi…
talk.religi…
precision
recall
f1-score
meta-chart.com


100
80
60
40
20
0
%
87
87
74
55
63
61
47
53
69
69
71
86
41
55
66
51
58
69
74
57
34
43
28
4
6
rec.sport.…
sci.crypt
sci.electr…
sci.med
sci.space
soc.religi…
talk.politi…
stalk.polit…
talk.politi…
talk.religi…
precision
recall
f1-score
meta-chart.com


70
60
50
40
30
20
10
0
%
29
28
41
41
46
46
42
47
51
51
49
54
48
51
55
33
41
57
56
60
alt.atheism
comp.gra…
comp.os.…
comp.sys…
comp.sys…
comp.win…
misc.fors…
rec.autos
rec.motor…
rec.sport.…
percision
recall
f1-score
meta-chart.com