An Analysis of Review Polarity by Text Classifying

Jiadao Zou, Houyi Liu and Xueling Fan
May 5, 2019

Abstract

In this project, we explored and performed sentiment analysis on reviews obtained from the Yelp Data Challenge. We were interested in finding the most efficient technique in predicting the sentiment polarity according to review context. During the experiment, we made a complete analysis by comparing the results from various feature extraction methods, learning algorithms, dataset. Our result gave out a clear performance comparison of many techniques which could be a fair reference for certain engineering problem.

Table of Contents

Introduction and problem description

Background

Yelp, founded in 2004, is an American multinational corporation headquartered in San Francisco, California. It helps people find great local businesses like dentists, hair stylists and restaurants. Yelp provides a platform for customers to give a star-rating for business and write public reviews. Surveys show that Yelp reviews affected customers' opinion and finally influence a business's first impression of new consumer and consuming decision.

Motivation

We believe that, compared to star ratings, the text-rich reviews are more indicative of restaurants' properties and user preferences. If we can extract the most important keywords from customers' reviews, we can not only identify hidden properties of businesses, but also accurately depict individual customer's profile. In other words, myriads of hidden information can be mined, and these invaluable data could be beneficial to business owners as well as customers, if we apply proper techniques. Nevertheless, how to extract the treasures behind enormous comments, has not conclude the most optimized solution for a long period.

So, by this project, we not only built models that can extract useful features from millions of reviews for processing these reviews using sentiment analysis, finding the most featured words(Koncz, Conference, & 2011, n.d.) and predicting the proper ratings solely based on review text, but also analyzed these models' performance under different scenarios. Our experiment could give out some useful evaluation metrics in order to decide which model is better for such problem.

Project target

We implemented a couple of feature selection algorithms with different classification algorithms and discovered some interesting differences between each model in this project. The model consists of various detail procedure, but, just like any other machine learning application, it contains learning part (applying some supervised learning algorithms to predict the author's sentiment polarity, e.g. positive, negative or neutral, towards the business based on user's comment alone) and predicting part (using the learnt model to predict an unseen review's rating on a given numerical scale from 1 to 5). We used all the multiple-classes classification algorithms include Multinomial Logistic Regression(Menard, 2002), Decision Tree(Safavian, systems, man, & 1991, n.d.) , Random Forest(Liaw, news, & 2002, n.d.) and Naïve Bayes(artificial & 2001, n.d.) . The evaluation metric based on f1, precision, recall and accuracy will be used as our criterion to analyze the effectiveness of our models.

Dataset Description

The dataset used for this project is readily available from Yelp Challenge at this link.
The dataset contains information about businesses as well as users in 12 metropolitan areas across four countries. Here, we focused on merely *review.json* file. The data we read contains following Information:

Attribute Name	Description
review_id	string, 22 character unique review id
user_id	string, 22 character unique user id
business_id	string, 22 character business id
stars	integer, star rating
date	string, date format YYYY-MM-DD
text	string, the review itself
useful	integer, number of useful votes received
funny	integer, number of funny votes received
cool	integer, number of cool votes received

Data preparation

Loading

After the dataset was loaded, we removed the null data. Afterwards, we split the dataset into 70% train set and 30% test set. In case of the one-time splitting is not general enough, repeated-holdout parameter was performed to randomly split data for several times to ensure the diversity of training and testing set.

Data pre-processing

Tokenization

Tokenizing refers to the act of separating a sentence into individual word tokens. It is the very first step towards processing individual words in a piece of text. We use the Tokenizer from *spark.ml library*, As an example:

Stanford University is located in California.

The result returned by the Tokenizer() is as follows:

["Stanford","University","is","located","in","California","."]

StopWordsRemover

After tokenization, we need to remove the common stop words in English. Stop words are words that appear extremely common but do not convey much meaning nor sentiment of bodies of text. We use the StopWordsRemover from *spark.ml library*, As an example:

["Stanford","University","is","located","in","California","."]

The result returned by the StopWordsRemover() is as follows:

["Stanford","University","located","California"]

FeatureExtractor

According to *spark.ml library*, they provide several FeatureExtractors: TF-IDF (based on Hashing TF), CountVectorizer, Word2Vec. Here, we make use of all of them to convert words into feature vectors.

Sentiment Polarity transform

Before we start looking into our data processing, we first have to come up with three terms that will be used repetitively in the whole process, and which are essential to our project. Throughout the whole project, we considered the reviews with star ratings higher than 3, to be the positive reviews, the with star ratings lower than 3, to be the negative reviews. Since the remaining reviews with 3 stars seem to be ambiguous. We just take them as neutral.

Pipeline model building, training and testing

Our project is a typical example of supervised multiple-classes classification task since a labelled dataset containing text documents and their labels is used to train a classifier while our goal is to make the predication as closer as to the validation set. To reduce redundant code, we use high-level API: *spark.ml.pipeline*. Thus, for a single model, there are following: First: pipeline declaration; Second: full model generation by declaring relevant paraGrid for cross validation to choose most optimized learning model parameter; Third: fitting the model; Fourth: Achieving metrics by taking advantage of *spark.ml.evaluation.MulticlassClassificationEvaluator*. In the following, we will express the machine learning part in detail.

Model Training

Through data pre-processing, a machine learning model could be trained on a labelled dataset's polarity. There are many different choices of machine learning models and as I have mentioned above we used all the multiple classification algorithms include Multinomial Logistic Regression, Decision Tree, Random Forest and Naïve Bayes.

ParameterGridBuilder was used for parameter tuning and CrossValidator was used to find the best model. Data was divided into k blocks for k-fold cross validation.

Last but not least, we use multiple holdout to ensure our result is not constrained by a single dataset splitting.

Model Testing and Evaluation

Apply the model created in the previous step and output classification evaluation metrics.

Classification Evaluation Metrics

	Predicted class		
Actual class		Class = positive	Class = negative
	Class = positive	TP	FN
	Class = negative	FP	TN

Referring the Spark ML document: (Spark ML doucment, 2019)
True Positive (TP): correctly predicted positive values
True Negative (TN): correctly predicted negative values
False Positive (FP): actual class is negative but predicted to be positive
False Negative (FN): actual class is positive but predicted to be negative

Metric	Definition
Precision	$\frac{1}{N}\sum_{i=0}^{N-1}\frac{\lvert P_i \cap L_i\rvert}{\lvert P_i\rvert}$
Recall	$\frac{1}{N}\sum_{i=0}^{N-1}\frac{\lvert L_i \cap P_i\rvert}{\lvert L_i\rvert}$
Accuracy	$\frac{1}{N}\sum_{i=0}^{N-1}\frac{\lvert L_i \cap P_i\rvert}{\lvert L_i\rvert+\lvert P_i\rvert-\lvert L_i \cap P_i\rvert}$
Precision by label	$PPV(\ell)=\frac{TP}{TP+FP}=\frac{\sum_{i=0}^{N-1} I_{P_i}(\ell)\cdot I_{L_i}(\ell)}{\sum_{i=0}^{N-1} I_{P_i}(\ell)}$
Recall by label	$TPR(\ell)=\frac{TP}{P}=\frac{\sum_{i=0}^{N-1} I_{P_i}(\ell)\cdot I_{L_i}(\ell)}{\sum_{i=0}^{N-1} I_{L_i}(\ell)}$
F1-measure by label	$F1(\ell)=2\cdot\left(\frac{PPV(\ell)\cdot TPR(\ell)}{PPV(\ell)+TPR(\ell)}\right)$
F1 Measure	$\frac{1}{N}\sum_{i=0}^{N-1} 2\frac{\lvert P_i \cap L_i\rvert}{\lvert P_i\rvert\cdot\lvert L_i\rvert}$

By taking our case into above formula table, we could get:

Accuracy = $\frac{TP+TN}{TP+FP+FN+TN}$, the ratio of correctly predicted observation to the total observations.

Precision = $\frac{TP}{TP+FP}$, the ratio of correctly predicted positive observations to the total predicted positive observations.

Recall = $\frac{TP}{TP+FN}$, also called sensitivity, is the ratio of correctly predicted positive observations to all observations in actual class.

F1 = $\frac{2\times(Recall\times Precision)}{Recall+Precision}$, the weighted average of precision and recall.(Carvalho & Cohen, n.d.)

Results and Analysis

As a summary, in this project, we use three different feature extractors, four different supervised classifications, three different size of dataset.(Carvalho & Cohen, n.d.)

Feature Extractors	hashingTF_IDF
	CountVectorizer
	Word2Vector
Multi-classes Classification	Multinomial Logistic Regression Classifier
	Decision Tree Classifier
	Random Forest Classifier
	Naïve Bayes
Dataset Size (# records)	1,000
	5,000
	50,000

What to be mentioned is that the combination of Word2Vec and Naïve Bayes is not able to come into the step of producing result due to the problem that *"Exception thrown in awaitResult:"* error when we were trying to use train dataset to fit the model. Therefore, for each size of dataset, we totally have 11 different combinations of feature extractors and classifiers to the same number of pipelines. With the *spark.ml.evaluator* package, we could easily get the evaluation metrics including precision, recall, f1 and accuracy to serve as the output for each experiment. Based on that, we then calculated the average result of the various experiments above and compared the effectiveness of different algorithms on the same dataset.

Table 1: Results of polarity prediction using 1,000 datasets

		F1	Precision	Recall	Accuracy
TFIDF	LR	0.59125	0.57916	0.61137	0.61137
	DT	0.5862	0.58493	0.62594	0.62594
	RF	0.55556	0.5743	0.65443	0.65443
	NB	0.60614	0.59276	0.63844	0.63844
W2V	LR	0.59355	0.5916	0.67724	0.67724
	DT	0.60119	0.58589	0.6508	0.6508
	RF	0.60795	0.59808	0.67683	0.67683
CV	LR	0.71164	0.70751	0.72659	0.72659
	DT	0.62853	0.62842	0.67683	0.67683
	RF	0.60473	0.62412	0.6884	0.6884
	NB	0.71467	0.71384	0.71861	0.71861

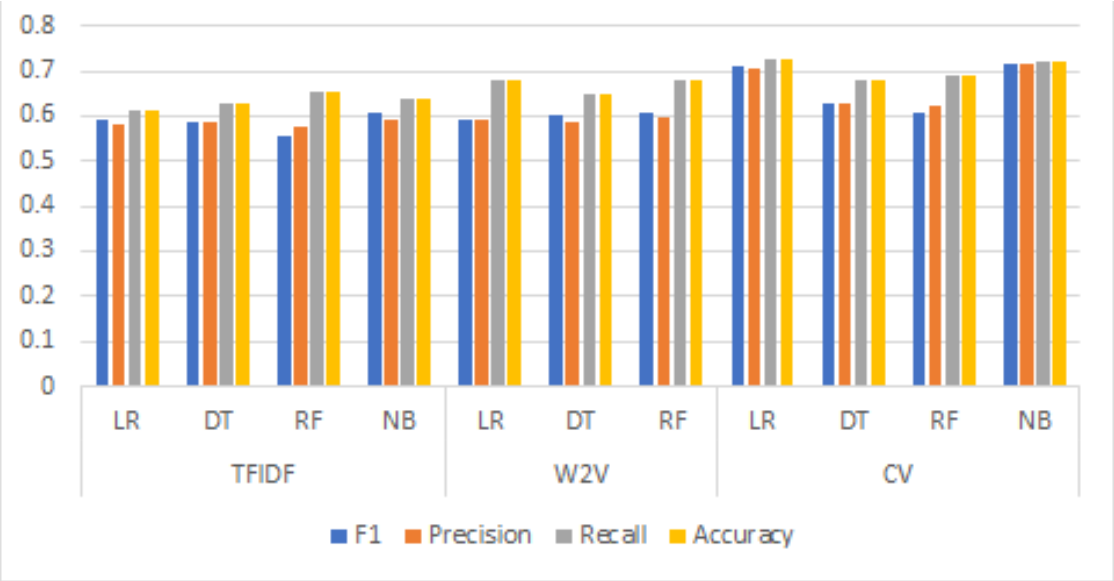

Figure 1: Results of polarity prediction using 1,000 dat

Table 2: Results of polarity prediction using 5,000 datasets

		F1	Precision	Recall	Accuracy
TFIDF	LR	0.632634	0.636243	0.693007	0.693007
	DT	0.591562	0.588727	0.667543	0.667543
	RF	0.559252	0.579444	0.680592	0.680592
	NB	0.64366	0.636581	0.686666	0.686666
W2V	LR	0.698642	0.695902	0.735358	0.735358
	DT	0.659872	0.643931	0.707602	0.707602
	RF	0.651299	0.67109	0.719111	0.719111
CV	LR	0.720406	0.717065	0.747464	0.747464
	DT	0.651772	0.640235	0.681634	0.681634
	RF	0.589862	0.618573	0.6932	0.6932
	NB	0.707809	0.704444	0.714294	0.714294

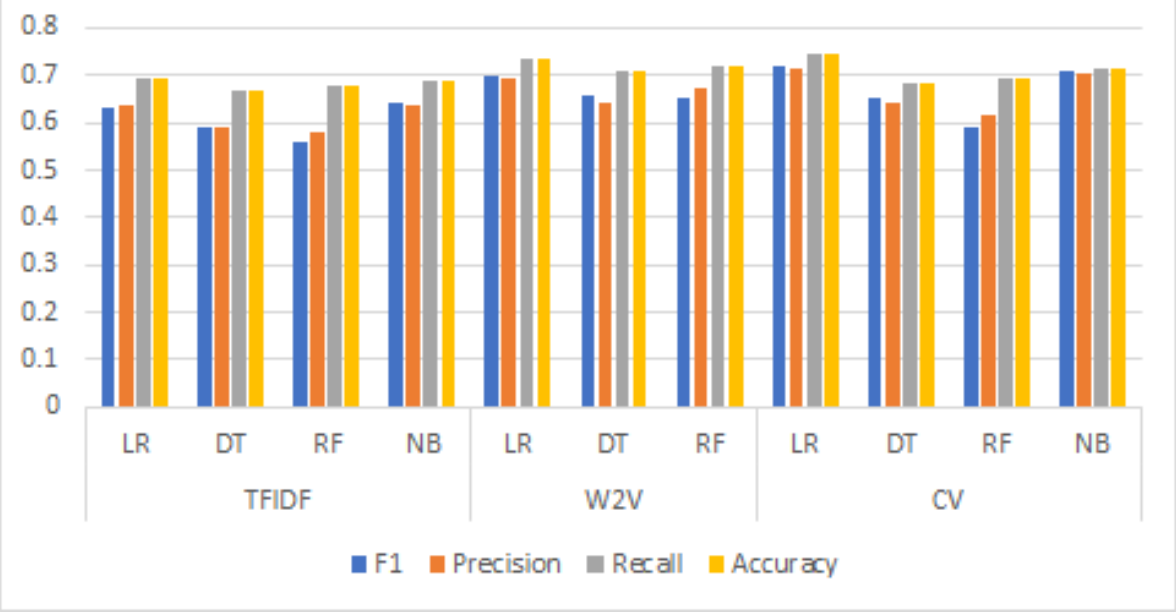

Figure 2: Results of polarity prediction using 5,000 datasets

Table 3: Results of polarity prediction using 50,000 datasets

		F1	Precision	Recall	Accuracy
TFIDF	LR	0.618862	0.638193	0.68791	0.68791
	DT	0.568909	0.575826	0.670288	0.670288
	RF	0.534607	0.593629	0.66267	0.66267
	NB	0.629327	0.627086	0.683604	0.683604
W2V	LR	0.762789	0.756629	0.789732	0.789732
	DT	0.672271	0.635582	0.723948	0.723948
	RF	0.654381	0.647032	0.721696	0.721696
CV	LR	0.7096	0.710892	0.747797	0.747797
	DT	0.615113	0.598243	0.690361	0.690361
	RF	0.584551	0.616567	0.683008	0.683008
	NB	0.6935	0.688888	0.699172	0.699172

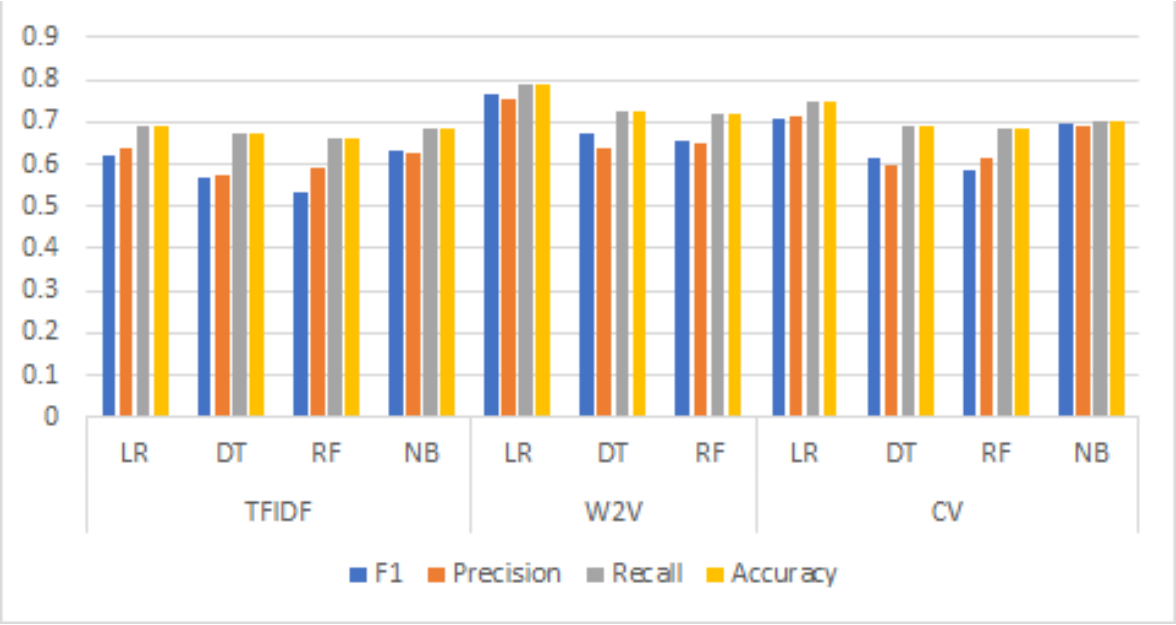

Figure 3: Results of polarity prediction using 50,000 datasets

Table 4: Results of stars prediction using 1,000 datasets

		F1	Precision	Recall	Accuracy
TFIDF	LR	0.363684	0.360918	0.39584	0.39584
	DT	0.32044	0.300226	0.38156	0.38156
	RF	0.280783	0.269487	0.406907	0.406907
	NB	0.365374	0.354389	0.406856	0.406856
W2V	LR	0.331238	0.318959	0.412565	0.412565
	DT	0.34489	0.328849	0.397996	0.397996
	RF	0.34049	0.343992	0.438997	0.438997
CV	LR	0.44054	0.429345	0.470792	0.470792
	DT	0.35642	0.35001	0.414648	0.414648
	RF	0.324883	0.326588	0.423453	0.423453
	NB	0.444203	0.439829	0.458785	0.458785

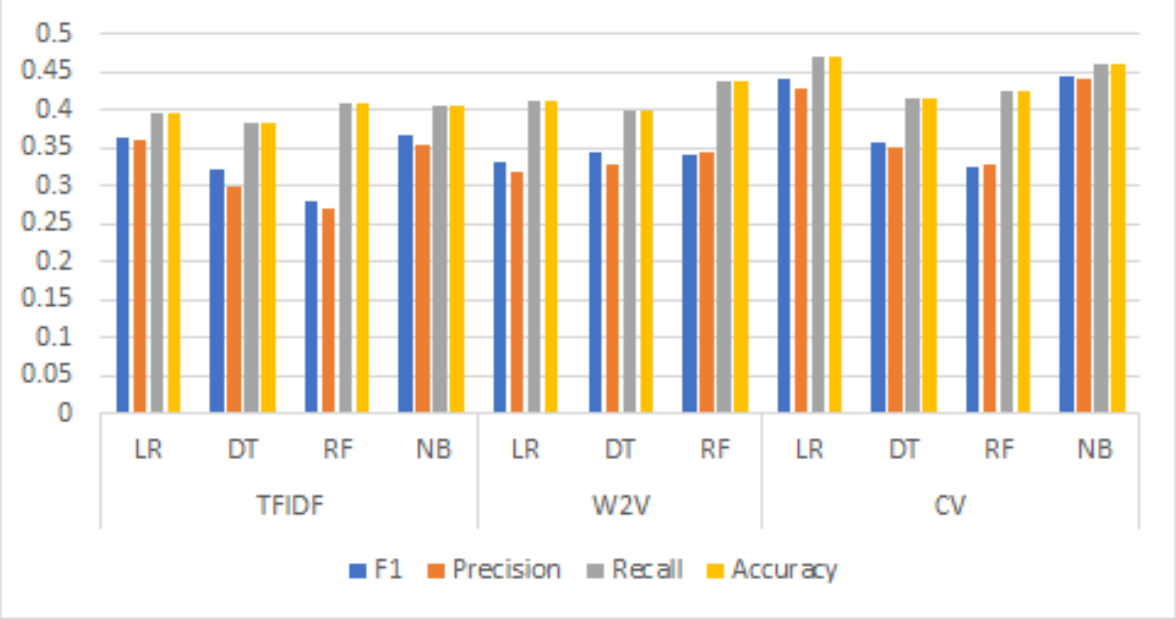

Figure 4: Results of stars prediction using 1,000 datasets

Table 5: Results of stars prediction using 5,000 datasets

		F1	Precision	Recall	Accuracy
TFIDF	LR	0.415862	0.424319	0.48513	0.48513
	DT	0.345065	0.325744	0.446686	0.446686
	RF	0.317068	0.352929	0.456805	0.456805
	NB	0.428198	0.424745	0.48067	0.48067
W2V	LR	0.479465	0.477898	0.531666	0.531666
	DT	0.420996	0.387369	0.470034	0.470034
	RF	0.387505	0.388614	0.495616	0.495616
CV	LR	0.48169	0.479801	0.536757	0.536757
	DT	0.371552	0.36176	0.4552	0.4552
	RF	0.346067	0.390424	0.473826	0.473826
	NB	0.50369	0.496449	0.517759	0.517759

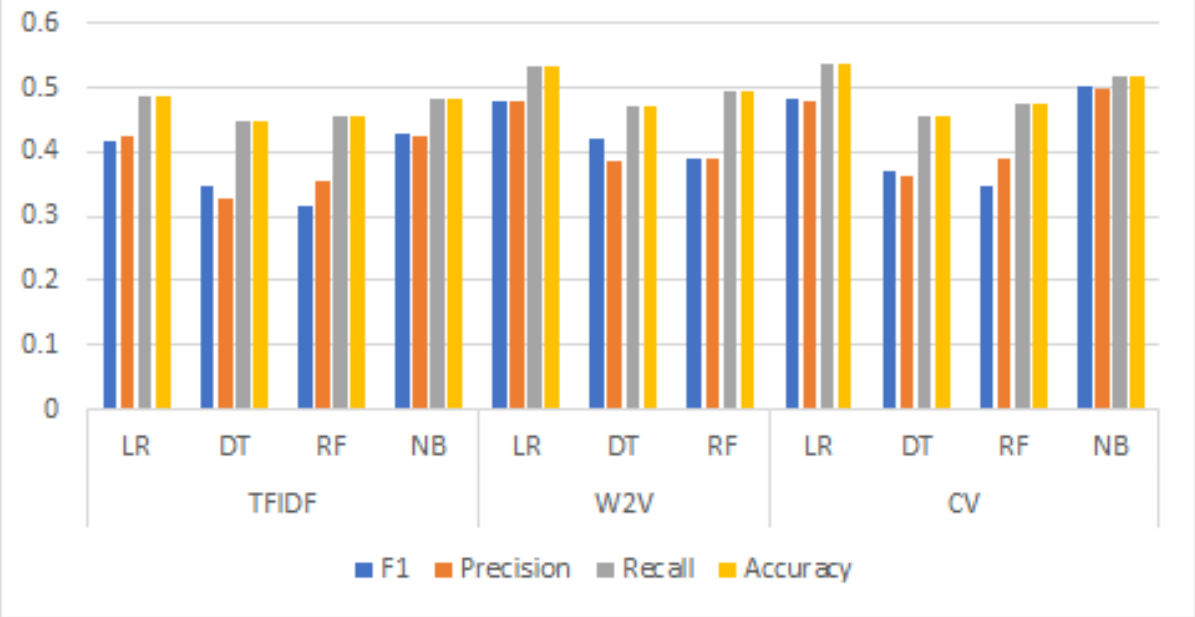

Figure 5: Results of rating prediction using 5,000 datasets

Table 6: Results of stars prediction using 50,000 datasets

		F1	Precision	Recall	Accuracy
TFIDF	LR	0.405147	0.425294	0.483503	0.483503
	DT	0.334943	0.342901	0.450774	0.450774
	RF	0.294468	0.30305	0.443727	0.443727
	NB	0.420955	0.425085	0.48245	0.48245
W2V	LR	0.569838	0.561575	0.597366	0.597366
	DT	0.47712	0.45543	0.510635	0.510635
	RF	0.416294	0.39197	0.519131	0.519131
CV	LR	0.492951	0.496773	0.545143	0.545143
	DT	0.412359	0.407217	0.470267	0.470267
	RF	0.332715	0.369503	0.467962	0.467962
	NB	0.509737	0.501885	0.523411	0.523411

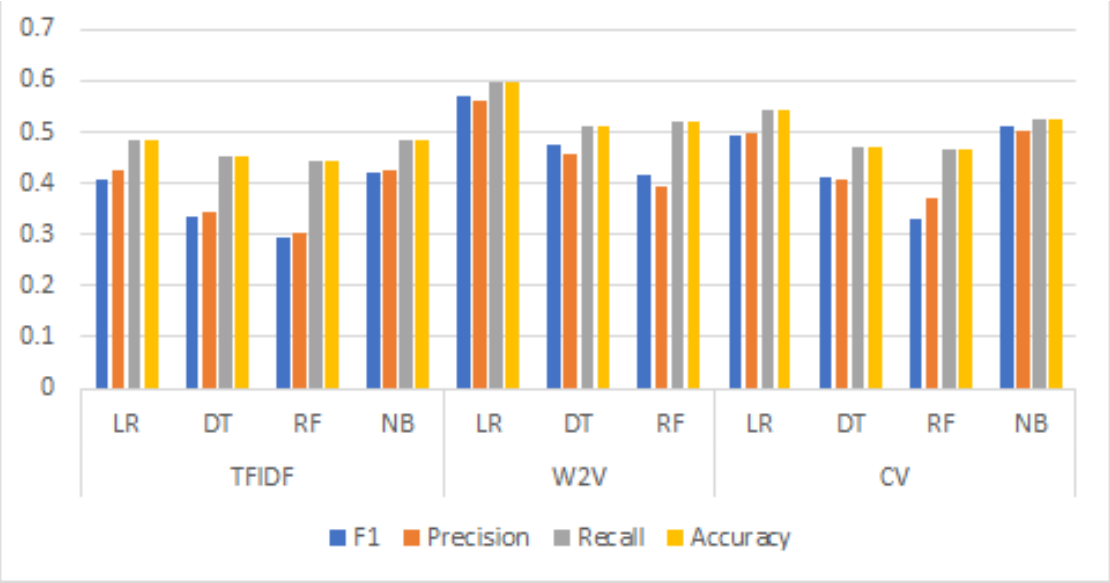

Figure 6: Results of stars prediction using 50,000 datasets

From above tables, we see that among all the three classifiers, logistic regression comes out the best performance whereas decision tree has the worst result produced. Logistic regression wins the match may due to parameters tuning on regularization parameter and elastic net. And the overall performance analysis of three different feature extractors shows TF-IDF is the worst one.

When we are comparing the results of sentiment prediction using 1,000 records dataset, 5,000 records dataset and 50,000 records datasets, we found that a larger dataset will cause the difference in performance to be more significant. For example, the difference of the best combination CountVector + Logistic Regression and the worst combination CountVector + DecisionTree is much larger than 1,000 records datasets when using 50,000 datasets. Also, when using a small dataset, difference in the performance of those two models decrease. So, we could conclude that larger dataset could enlarge the difference between better model and worse model.

Moreover, by comparing the results of sentiment task and rating task, the results show that sentiment prediction has a higher accuracy, which is not surprising since for the number of labels for sentiment prediction is three (positive, negative, neutral) whereas number of possible output of rating prediction is five (1-5 stars). As the number of labels goes up, it asks for more computing and becomes harder to correctly predict the text.

Another thing we can conclude from the results is that larger dataset yields higher accuracy. Overall, we could get the best model: Word2Vector has the accuracy of 78.9% of 50,000 records dataset in sentiment analyzing, while the same model has an accuracy about 67.7% of 1,000 records dataset. What an obviously comparison. This phenomenon also shows up in numeric starts reviewing. In summary, expect for the predicting capability of model, the size of dataset also takes a big part in final result. So, we should take as large dataset as possible under current available computation resource.

Conclusion

In this project, we have applied different combinations of feature extractors and machine learning models to predict the user's sentiment and rating based on Yelp reviews dataset. The results reveal that our evaluation metrics: f1, precision, recall and accuracy vary a lot using different algorithms or on different size of dataset.

In all cases, logistic regression has the best performance whereas decision tree has the worst performance. While the dataset goes larger, the difference in performance of different models becomes more significant, and larger dataset also yields higher accuracy which is reasonable cause larger dataset give the algorithms more training information to learning and self-correcting space. Due to the sentiment prediction only ask for three-choose-one prediction, it's result definitely would beat five-choose-one rating predicting task.

Future work

We are going to apply streaming and visualizing techniques like Kibana and Kafka, just as what we have done in assignment 3, to dynamically visualizing streaming data. Also, this project could be extended to other topics such as business reviews, and other kind of sentiment analyzing fields. Therefore, in such cases, we expected different result from current one, cause the weight of specific feature extracting techniques may vary when it is applied on different topics of data.

Contribution claim

As a team, we all actively take part in each of the steps of the whole project.

References

artificial, I. R.-I. 2001 workshop on empirical methods in, & 2001, undefined. (n.d.). An empirical study of the naive Bayes classifier. *Cc.Gatech.Edu*.

Carvalho, V. R., & Cohen, W. W. (n.d.). *Learning to extract signature and reply lines from email*.

Koncz, P., Conference, J. P.-2011 15th I. I., & 2011, undefined. (n.d.). An approach to feature selection for sentiment analysis. *Ieeexplore.Ieee.Org*.

Liaw, A., news, M. W.-R., & 2002, undefined. (n.d.). Classification and regression by randomForest. *Researchgate.Net*.

Menard, S. (2002). *Applied logistic regression analysis*.

Safavian, S., systems, D. L.-I. transactions on, man, undefined, & 1991, undefined. (n.d.). A survey of decision tree classifier methodology. *Ieeexplore.Ieee.Org*.