Abstract

This project addresses the problem of sentiment analysis in twitter; that is classifying tweets according to the sentiment expressed in them: positive, negative or neutral. Twitter is an online social-networking platform which allows users to write short status updates of maximum length 140 characters. Analyzing the public sentiment is important for many applications such as firms trying to find out the response of their products in the market, predicting political elections. The aim of this project is to develop a classifier for accurate sentiment classification of tweets.

1. Introduction

Twitter sentiment analysis comes under the domain of "text Classification". Twitter based features are more informal and relate with how people express themselves and compress their sentiments in the limited space of 140 characters offered by twitter. They include twitter hashtags, word capitalization, question marks, presence of URL in tweets, retweets, exclamation marks, internet emoticons and internet slangs.

Supervised classification technique is used to classify tweets. Supervised approach is used when we have pre-labeled data samples available and we use them to train our classifier. Training the classifier means to use the pre-labeled to extract features that best model the patterns, and then classifying an unlabeled data sample according to whichever pattern best describes it.

We also performed 10-fold validation on the training data set to avoid overfitting. During the training phase, various vectorizers are used to convert each input data to a feature set. Then, these features along with corresponding labels are input into the machine learning algorithm to train a model. During the prediction phase, the same vectorizer is used on the test data, and the converted feature sets are input into the model to generate the predicted labels.

There are several metrics proposed for computing and comparing the results of our experiments. Some of the most popular metrics include Accuracy, Precision, Recall, F-Score-measure.

2.Techniques used:

2.1 Data processing steps:

In order to prevent the feature space from oversizing, all the duplicate information present in the feature space is removed, similar words are grouped together as a single feature and finally only meaningful words are kept in the feature space. Therefore, before extracting the features, preprocessing is done on the tweets as described below:

- Considered only the tweets and the label fields from the given data.
- Considered only the tweets with labels -1, 0 and 1.
- Removed all the tags like <a>, <e>.
- All the punctuation marks embedded within a tweet is removed.
- Every tweet is converted to lowercase so that duplication due to case mismatch is eliminated.
- Hash tags can give us some useful information, so all the words in a tweet that begin with '#' are replaced with the exact same word without the hash. For example: #friendsforever is replaced with friendsforever

- Data obtained from tweets contains many embedded links with ‘www.’ or ‘http(s)’ such links are replaced with the word URL
- All the words containing @user is replaced with AT_USER to treat all the users the same.
- Replaced all words with more than two characters repeating with words in which characters are repeated exactly twice. (Example: exxxtenddddd to exxtendd)
- Acronyms such as lol, are expanded to laugh out loud using a list of acronym expansions.
- Stemming is done on each word of the tweet to reduce inflectional forms and derivationally related forms of a word to a common base form. Used the below NLTK libraries for stemming.
    - Porter Stemmer.
    - Snowball Stemmer.
- All the stop words provided by NLTK English are removed from the tweets to focus on more important words and to remove low value words.

2.2 Features used

Feature space is very crucial in classifying tweet sentiments because feature vector is used to build a model which the classifier learns from training data. This model is now used to new data.

- Considered only the tweets and labels (-1,0 1) from the given data.
- Unigrams, Bigrams and trigrams were used as well to construct the features.

2.3 Classification Algorithm and feature extraction methods

	TF- IDF	**Count Vectorizer**	**Hashing Vectorizer**
Multinomial Naïve Bayesian	**Yes**	**Yes**	**-**
Linear SVC	**Yes**	**Yes**	**Yes**
Logistic Regression	**Yes**	**Yes**	**Yes**
Decision Tree Classifier	**Yes**	**Yes**	**Yes**
Random Forest Classifier	**Yes**	**Yes**	**Yes**
K Neighbors Classifier (k = 5)	**Yes**	**Yes**	**Yes**
XGBoost	**Yes**	**Yes**	**Yes**
AdaBoost Classifier	**Yes**	**-**	**-**

3. Evaluation:

3.1 10-Fold Results:

Below are the results we obtained after performing 10 -fold cross validation on the testing data. Different feature extraction methods like TF_IDF, count vectorizer and Hashing vectorizer were used. Each of these extraction methods were tried with oversampling and under sampling. Under sampling results are not included in the results because under sampling did not improve the results. Over sampling improved the performance of the model. Date presented in red color are the results obtained through oversampling. Below are the specific effects of different techniques used.

***Note**: Oversampled results are in red.*

NaiveBayesin	**TF-IDF**				**Count vector**			
	OBAMA		ROMNEY		OBAMA		ROMNEY	
Accuracy	**58.5**	**60.4**	**54.86**	**57.1**	**57**	**58**	**57**	**56.1**
Precision (+)	66.6	60.4	81.2	48.8	55.9	55.4	51.1	43.1
Precision (-)	55.7	60.5	53.9	68.8	58.9	60.1	61.2	68.6
Recall (+)	57.9	71.5	10.6	62.9	64	69.6	35.4	57.8
Recall (-)	70.4	65.9	97.9	65.0	66.4	64.3	81.4	66
F-Score (+)	62	65.5	18.8	54.9	59.6	61.7	41.7	49.3
F-Score (-)	62.2	63.1	69.6	66.8	62.4	62.1	69.9	67.2

Linear SVC	**TF-IDF**				**Hashing vector**		**Count vector**			
	OBAMA		ROMNEY		OBAMA	ROMNEY	OBAMA		ROMNEY	
Accuracy	**59.5**	**59.2**	**56.3**	**54**	**60.1**	**60.2**	**53.8**	**54.3**	**54.7**	**53.4**
Precision (+)	63.7	62.2	52.8	47	63.8	61.7	52.5	52.8	45.5	41.9
Precision (-)	61.2	61.6	62.7	65.4	59.6	62.1	58.9	59.1	65.5	66.7
Recall (+)	61.9	62.6	38.5	47.1	61.1	34.9	62.9	64.9	42.8	49.2
Recall (-)	61.8	60.3	73.2	63.9	64.4	84.8	53.6	53.4	66.6	61.6
FScore (+)	62.8	62.4	44.5	47.1	62.3	44.4	57.2	58.1	43.9	45.1
FScore (-)	61.5	61	67.5	64.6	61.8	71.6	55.9	56	66	63.9

LogisticRegressin	**TF-IDF**			**Hashing vector**		**Count vector**			
	OBAMA	ROMNEY		OBAMA	ROMNEY	OBAMA		ROMNEY	
Accuracy	**58.0**	**57.7**	**56.1**	**57.9**	**56.8**	**57.9**	**57.9**	**57.8**	**55.8**
Precision (+)	63.6	71.1	47.7	63.1	68.4	58.2	56.8	53.6	45.5
Precision (-)	58.1	58.1	69.1	56.1	56.9	60.5	62.4	63.6	68.7
Recall (+)	58.5	26.2	55.1	56.5	16.3	63.9	66.6	37.5	53.8
Recall (-)	63.4	88.0	63.9	66.9	92.7	59.2	58.1	77.1	63.1
FScore (+)	60.9	38.3	51.2	59.5	26.3	60.8	61.3	44	49.2
FScore (-)	60.6	70.0	66.4	60.9	70.5	59.8	60.1	69.6	65.8

DecisionTree	**TF-IDF**				**Hashing vector**		**Count vector**			
	OBAMA		ROMNEY		OBAMA	ROMNEY	OBAMA		ROMNEY	
Accuracy	**44.9**	**46.1**	**50.8**	**47**	**50.79**	**52.3**	**49.9**	**50.3**	**49.5**	**49.4**
Precision (+)	45.9	45.9	37.8	37.7	53.1	42	46.8	46.8	34.3	34.2
Precision (-)	48.1	50.1	59.4	60	51.9	59.9	54.6	56.1	61.9	65.2
Recall (+)	51.7	49.8	30.3	38.2	52.7	29.7	59.5	61.4	42.3	51.6
Recall (-)	41.4	43.9	65.8	55.8	50.1	69.8	48.5	47.8	60.1	54.2
F-Score (+)	48.7	47.7	33.6	37.9	52.8	34.6	52.2	53.1	37.8	41.1
F-Score (-)	44.5	46.8	62.4	57.8	50.8	64.4	51.3	51.5	60.8	59.1

	kNN(TF-IDF)				**Random Forest (TF-IDF)**				**XGBoost(TF-IDF)**			
	OBAMA		ROMNEY		OBAMA		OBAMA		OBAMA		OBAMA	
Accuracy	**51.6**	**47.5**	**54.5**	**43.9**	**55**	**55.5**	**55**	**55.5**	**52.7**	**52.3**	**52.7**	**52.3**
Precision (+)	53.7	50.3	59.5	59.7	58.2	56.6	58.2	56.6	54.3	52.1	54.3	52.1
Precision (-)	59.3	54.4	60.3	76	58.9	59.9	58.9	59.9	51.5	53.8	51.5	53.8
Recall (+)	53.9	50.9	18.1	22.2	59.7	61.8	59.7	61.8	60.2	63.4	60.2	63.4
Recall (-)	42.4	38.7	69.4	32.9	52.3	52.2	52.3	52.2	54.3	56.2	54.3	56.2
F-Score (+)	53.9	54.4	27.7	29.8	58.9	59.1	58.9	59.1	57.1	57.1	57.1	57.1
F-Score (-)	49.5	45.2	64.5	40.6	55.4	55.7	55.4	55.7	52.8	55.1	52.8	55.1

Note: For kNN, Random Forest and XGBoost count vectorizer and hashing vectorizer results are not included as they did not perform better than TF-IDF

Effects of techniques tried:

- **Classification methods:** Trying different classification algorithms did not improve the performance. In general, Naïve Bayesian, Linear SVC and Logistic Regression performed the best.
- **Under sampling:** Under sampling reduced the sample size and reduced the overall performance of all the models.
- **Over sampling:**
  - o F-Score: As Romney data set was very imbalanced, average (positive and negative) F-score for Romney data increased for all the models.
  - o Accuracy: For Obama and Romney data, accuracy improved for some of the models.
- **Acronym expansion:**
  - o Accuracy: Improved for both Obama and Romney by 1%.

Best results obtained on training data after 10-fold cross validation:

	Model	*Accuracy*	*Precision (+)*	*Recall (+)*	*F-Score (+)*	*Precision (-)*	*Recall (-)*	*F-Score (-)*
OBAMA	Naïve Bayesian	**60.4**	60.4	71.5	**65.5**	60.5	65.9	**63.1**
ROMNEY	Linear SVC	**60.2**	61.7	34.9	**44.4**	62.1	84.8	**71.6**

3.2 Test data results

Best results obtained on testing data with Hashing Vectorizer:

	Model	*Accuracy*	*Precision (+)*	*Recall (+)*	*F-Score (+)*	*Precision (-)*	*Recall (-)*	*F-Score (-)*
OBAMA	**Linear SVC**	**57.09**	61.25	51.89	56.18	57.46	63.22	60.20
ROMNEY	**Linear SVC**	**62.15**	69.78	42.59	52.90	62.46	87.18	72.78

4. Conclusion

- Oversampling improves accuracy for skewed data, in this case for Romney data which has more negative class tweets.
- 10-Fold testing reduced the possibility of overfitting by randomizing the tweets and then splitting into test (10%) and train sets (90%)
- Naive Bayesian and Linear SVC classifier performed better than any other classifiers tried
- Linear SVC classifier with hashing vectorizer performed best for test data with accuracy of 57.09% for Obama data and 62.15% for Romney data.

5. References

1. http://scikit-learn.org/stable/modules/classes.htm
2. http://stackoverflow.com/questions/19431063/python-re-sub-to-replace-3-or-more-of-same-character-with-1
3. https://raghu4690.bitbucket.io/dataScience/Twitter_Sentiment_Classification_using_Apache_Spark/