Twitter Sentiment Analysis

CS 583 – Data Mining & Text Mining

Harshad Hussain (651043985) and Viswanath Veerappan (671492285)

Abstract

The purpose of this research project is to develop a sentiment analyzers capable of analyzing tweets and extracting sentiments conveyed by them, using multiple supervised learning techniques and provide a comparative study of those techniques. The sentiment analyzer is a classification model built using supervised learning methodologies for classifying text into predefined classes. This corpus-based model uses a training data consisting of a collection of tweets about Obama and Romney during 2012 US Presidential elections. The tweets have been manually classified into three basic classes as positive, negative and neutral based on the sentiment conveyed in them. The classification model aims to classify a similar tweet as positive, negative or neutral based on statistical parameters learned from training data.

Introduction

Sentiment analysis or opinion mining is the process of computationally determining emotions conveyed by opinions, sentences and attitudes. One of the major applications of sentiment analysis is monitoring social media to understand public opinion on various topics. Twitter is one of the most widely used social media platforms through which general public consumes information and also expresses opinions on various topics ranging from politics, sports, entertainment, etc...

In this project, a sentiment analyzer is built using different supervised machine learning techniques. This sentiment analyzer analyzes tweets that talked about Obama and Romney during 2012 US Presidential elections. These tweets were collected for a specific period of time. The sentiment analyzer is a corpus-based system that requires the deep understanding of sentence structures and semantics. The task of sentiment analyzer is to classify a text, in this case a tweet, into different classes based on the sentiment conveyed in them. A statistical approach to text classification helps to simplify the problem by considering text as a set of vectors, rather than an entity with scope, meaning and structure. For this project, classification of tweets is done using some classic classification techniques like Multinomial Naïve Bayes, Logistic Regression, Support Vector Machines, and some ensemble techniques like Random Forest and ADA Boost. The effective of these techniques are evaluated by comparing parameters like – overall accuracy, precision, recall and F-Score – obtained for each of the techniques.

A training data consisting of over 7000 tweets that were made on Obama and Romney each, during a specific period of time have been classified as to be positive, negative or neutral based on the

sentiment conveyed in them. Two separate classifiers have been built to classify tweets relating to Obama and Romney. The classifiers are built using this training data using 10-fold validation technique before feeding testing data to be classified to the model.

Techniques

1. Data cleaning and preprocessing

The tweets collected for training and testing purposes contained noise in the form of HTML tags, URLs, irrelevant numbers, twitter handles (@ words), twitter keywords like RT, contracted words (aren't, can't), stop words (the, an), unnecessary white spaces, #tags.

Data cleaning and preprocessing techniques are applied to training tweets before using them to train the classifier, and to testing tweets before feeding them into the classifier to classify.

The following preprocessing filters are used to remove noise from raw tweets.

Numbers filter: Digits are not generally used to convey sentiments. Therefore, the incoming tweets are stripped off digits using regex.

HTML tags filter: An HTML tag remover built using beautifulsoup4 module for Python language strips the incoming tweet off HTML elements.

Twitter specific filters: A tweet generally contains twitter handles (words starting with @), hashtags, and keywords like RT. Filters written using regex are used to filter these out from tweets.

Case insensitivity: A tweet is made case insensitive by converting it into lowercase.

Expanding contractions: Tweets always contain contracted words – such as they're, aren't, wasn't, etc. These word contractions are expanded in input tweets. Therefore, the above-mentioned word contractions now appear as 'they are', 'are not', 'was not' etc.

URL filter: All the urls beginning with "www.", "https:\\", "http:\\" are removed from input tweet using regex. The above strings are replaced using a coming string 'URL'. 'URL' is then added to the list of stop-words.

Whitespaces filter: A tweet may contain unnecessary white spaces. A filter is used to limit

whitespaces to 1.

Repeating character filters: Tweets may contain intentionally misspelt words. For example, 'happy' could be tweeted as 'haaaapppppyyyyyyyy'. A filter written using regex is used to remove multiple continuous occurrences of same alphabets by two occurrences (' haaaapppppyyyyyyyy' will be 'haappyy').

Punctuation filter: A regex filter is used to remove punctuations from tweets.

Stop-words filter: Natural Language Tool Kit's (NLTK) sentiment analysis corpus provides a list of commonly used English language stop-words. Different techniques involving removing words (like 'no', 'not') that convey sentiments from this list, and words like 'retweet', 'atuser', 'url', 'obama', 'romney', 'biden' were added to stop-words list as these words do not affect the sentiment of a tweet, were tried. The combination that led to best results were retained.

Stemming: Snowball stemmer is a python module that is provided by NLTK. This has been used in the model for stemming.

Example

Raw tweet - @Barack<e>obama</e> <e>obama</e>'s <a>Female Debate Coach Complained About 'Hostile Workplace' at White House http://t.co/cmlFp0vI

After data preprocessing – femal debat coach complain hostil workplac white hous

2. Feature Extraction

For this research, the corpus was converted into a **TF-IDF vectors**. TF-IDF represents the relative frequency of a term in the text, normalized by the occurrence of the term in other passages and frequency of other terms in the same package. A TF-IDF matrix was constructed using the api 'TfidfVectorizer' provided by Python's sklearn feature extraction text module. All the machine learning algorithms were provided with the same features to enforce common grounds for comparison. Tfidf vectorizer was supplied the following parameters to extract features.

min_df (minimum document frequency) = 10
max_df (maximum document frequency) = 0.8
ngram_range = (1,3)
sublinear_tf = True
use_idf = True

3. Classification Models

Multinomial Naïve Bayes

Multinomial Naïve Bayes is a variant of Naïve Bayes algorithm used for text classification when data is distributed multinomially. A multinomial Naïve Bayes implementation considers the frequency of the word in the text to build the classification model. The frequency of a word in the text is calculated on the basis of a parameter named TF-IDF.

Linear Support Vector Machine

Support vector machine is a kernel based learning methodology. Given a set of labeled training data points, the algorithm aims to define an optimal hyperplane that can be used to classify new examples. For this research SVM using linear kernel. The classification model has been implemented using LinearSVC api from Python module sklearn.

Multinomial Logistic Regression

Logistic regression is a generalized linear model used for classification where there are more than two possible discrete outcomes. The probabilities describing the possible outcomes of a single trial are modeled using logistic functions. The classification model has been implemented using LogisticRegression api from Python module sklearn.

Random Forest Classifier

Random forest classifier is an ensemble learning method. The classifier constructs multiple decision trees during training. The final output class is the mode of classes determined by each decision tree. The classification model has been implemented using RandomForestClassifier api from Python module sklearn.

AdaBoost Classifier

This classifier is based on the boosting algorithm AdaBoost. This is also an ensemble classifier. The basic principle of AdaBoost is to fit a sequence of weak learners on repeatedly modified versions of the data. The classification model has been implemented using AdaBoostClassifier api from Python module sklearn. For this research number of estimators was set to 1000 to yield good results.

Evaluation of different Models

Each model based on different learning algorithm was developed separately for training and classifying tweets related to Obama and Romney. For building each classifier, the model was subjected to 10-fold cross validation mechanism. For each model precision, recall and fscore for both positive and negative sentiment classes where retrieved, along with the overall accuracy. Once the classifiers were trained, they were then used to classify the test data. The classifier that gave the best scores was a model built using Logistic Regression. The following tables give the results obtained for each class for both Obama and Romney – related tweets – both for 10-fold cross validation done on training data and for testing data.

Obama – Kfold Validation Training Results							
		Positive			Negative		
	Accuracy (%)	Precision (%)	Recall (%)	F-score (%)	Precision (%)	Recall (%)	F-score (%)
Multinomial NB	54.79	57.77	55.33	55.35	55.53	66.70	59.05
Linear SVC	54.57	54.11	54.09	54.09	58.15	59.71	57.84
Logistic Regression	56.13	58.22	54.62	55.44	58.18	63.53	59.47
Random Forest	54.51	61.75	67.16	62.86	57.34	61.93	58.08
AdaBoost	51.23	62.02	53.88	56.58	57.35	49.10	51.86

Romney – Kfold Validation Training Results							
		Positive			Negative		
	Accuracy (%)	Precision (%)	Recall (%)	F-score (%)	Precision (%)	Recall (%)	F-score (%)
Multinomial NB	55.79	61.74	20.32	29.66	57.53	89.43	68.71
Linear SVC	54.82	46.50	38.32	41.16	63.51	70.95	65.90
Logistic Regression	56.89	60.03	28.96	37.93	60.34	84.29	69.01
Random Forest	54.66	65.54	81.46	71.15	60.70	77.46	66.77
AdaBoost	52.14	60.91	83.90	69.10	56.37	80.02	65.17

Obama – Testing Data Results			
	Accuracy	Positive	Negative

	(%)	Precision (%)	Recall (%)	F-score (%)	Precision (%)	Recall (%)	F-score (%)
Multinomial NB	52.58	56.85	48.45	52.31	51.88	64.09	57.34
Linear SVC	52.99	55.00	48.10	51.32	56.21	57.84	57.02
Logistic Regression	54.38	57.05	51.81	52.10	57.14	61.04	59.18
Random Forest	51.51	50.08	51.03	50.55	54.03	55.52	54.76
AdaBoost	45.05	53.99	29.03	37.76	41.08	81.10	54.54

Romney – Testing Data Results							
		Positive			Negative		
	Accuracy (%)	Precision (%)	Recall (%)	F-score (%)	Precision (%)	Recall (%)	F-score (%)
Multinomial NB	59.31	64.00	29.09	40.00	59.55	89.58	70.54
Linear SVC	58.31	52.54	42.85	47.21	64.47	75.62	69.60
Logistic Regression	60.26	63.34	36.36	46.20	61.58	85.83	71.71
Random Forest	56.36	57.04	32.20	41.26	60.28	79.06	68.40
AdaBoost	54.52	54.19	21.81	31.11	54.99	92.29	69.92

Conclusion

The research project throws a light on how different learning algorithm gives different performance metrics for the same dataset. Based on the experiments performed on the five different learning techniques it was seen that Logistic Regression model performed better than the rest of the models. The model built using Logistic Regression performed consistently better than the models that were developed using other algorithms.

The experiment involved setting the right parameters to each classification algorithm to yield better results without over-fitting. As the data was very noisy, better and well-structure preprocessing steps yielded better accuracy and fscores.

References

- Natural Language Tool Kit (NLTK)
- Scikit Learn