Twitter Sentiment Classification

CLASSIFY TWEETS INTO POSITIVE, NEUTRAL, NEGATIVE

SAKSHI PANDAY, SHOBHIT LAMBA

Contents

1. Abstract

The process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic, product, etc. is positive, negative, or neutral, is known as sentiment analysis. In this report, we aim to determine the sentiments of tweets towards Obama and Romney based on the given training dataset. We employ several machine and deep learning models to find out the sentiments conveyed by a tweet and finally see which model performs the best out of all those used. The proposed models include Naïve Bayes, Logistic Regression, Convolution Neural Networks (CNN) and finally Recurrent Neural Networks (RNN) with Long Short-Term Memory (LSTM).

Keywords- Sentiment Analysis, Naive Bayes, Logistic Regression, Convolution Neural Networks, Recurrent Neural Networks, Long Short-Term Memory.

2. Introduction

Sentiment analysis of text is well known Natural Language Processing problem. Several state-of-the-art techniques are available to analyse and extract sentiments from given text. But, in case of tweets, where the text is small, with slangs, emoticons, hyperlinks and unconventional grammatical structure, these techniques often fail. In this project, we try to achieve a good performance (≈57% accuracy) for multi-class sentiment analysis on twitter data. In Section 3, we first talk about the data pre-processing steps undertaken and the features used. Secondly, we talk about the various techniques we used. We discuss about the conventional techniques we explored, namely Naïve Bayes (Bernoulli and Multinomial) and Logistic Regression. Then we talk about the various Deep Learning models we tried. We discuss our implementation of convolution (CNN) and recurrent (RNN) neural networks with long short-term memory (LSTM). We discuss our first-hand experience with neural networks and how they fared for us in the given task. In Section 4, we finally conclude with all the results tabulated for classical as well as deep learning models.

3. Technique

3.1. Data pre-processing

The training data underwent numerous pre-processing steps before the feature extraction. The steps followed were as follows:

- Removal of tweets that were not labelled as positive, negative or neutral.
- Converting the entire text to lowercase.
- Removal of hashtags, URLs and characters that were not alphabets.
- Removal of stopwords.
- Stemming using PorterStemmer.

3.2. Features used

We did the following for feature selection:

- **Lemmatization**: splitting tweets into n-grams lemmas for example ngram_range = (1, 3). Our final approach has given the best result using *ngram_range of (1,5)*
- **Bag of words**: converting tweets into a list of lemmas
- **TF-IDF transformation**: computing how important a lemma is based on number of times it appears in the dataset

3.3. Classification methods tried

3.3.1. Naive Bayes

3.3.1.1. Bernoulli

The first iteration of the classifier was designed on Bag-of-Words of binary representation i.e. if the particular word is present in the tweet, we mark it 1, else 0, irrespective of the count of occurrences with the thought process that a tweet has less words reducing the chance of multiple occurrences. As binary features are best handled by Bernoulli, it was tried. In spite of trying n-grams later on, Bernoulli gave the lowest result (accuracy of ≈48%).

3.3.1.2. Multinomial

The second iteration used Multinomial method to classify. Although, dataset of Obama was balanced across classes, Romney was imbalanced heavier on negative class. Much more number of tweets classified as negative resulted in classifier binning most tweets into negative category. Tweaking class_prior parameter of Multinomial partially helped with that aspect giving an accuracy of ≈53%, which was an improvement on Bernoulli, but by not much.

3.3.2. Logistic Regression

As Naïve Bayes did not give a balanced result for both datasets, Logistic Regression was used along with TF-IDF as feature selection on n-grams. After trying several n-gram ranges, an accuracy of ≈57% was achieved using Logistic Regression which was the highest accuracy attained by any classifier tried.

3.3.3. Neural Networks

After classical models, we moved our focus to Neural Networks and tried to make use of different deep learning models. We focused on two models as discussed below.

3.3.3.1. Long short-term memory + Recurrent NN

Our first LSTM-RNN model had a structure consisting of:

- One embedding layer
- One LSTM layer with a dropout rate of 0.2
- One Dense Layer with Softmax activation

And was compiled with a Categorical Crossentropy loss and Adam optimizer. While the results on Obama dataset were normalized, the skewed nature of Romney dataset saw it underperform while classifying positive tweets. Numerous changes were made to the neural model to optimize the results of the train-test split. Hyper parameters were tuned. The structure was changed to include more LSTM layers. PReLU advanced activation layer was also added. GLoVe pretrained twitter embedding were also incorporated to boost the performance but surprisingly, the network still under-performed.

3.3.3.2. Convolutional

Next, we shifted our focus to CNNs as RNN-LSTM was not performing well and took a long time to train which was not feasible. Our basic structure had one Convolution-1D layer followed by a Maxpooling and dense layer with ReLU activation. But, similar to the previous NN, even after several changes to the structure, the CNN failed to outclass the results achieved by Logistic Regression.

4. Evaluation

4.1.1. Naive Bayes

Obama: Bern	Negative	Positive
F-Score	59.96	03.50
Precision	42.97	71.42
Recall	99.15	01.79
Accuracy	43.79	

Romney: Bern	Negative	Positive
F-Score	69.72	02.04
Precision	53.71	99.80
Recall	99.33	01.03
Accuracy	54.15	

Obama: Multi	Negative	Positive
F-Score	64.29	35.96
Precision	48.01	83.12
Recall	97.25	22.94
Accuracy	53.01	

Romney: Multi	Negative	Positive
F-Score	66.83	13.67
Precision	50.37	80.00
Recall	99.27	07.47
Accuracy	51.68	

4.1.2. Logistic Regression

Obama: LogReg	Negative	Positive
F-Score	67.21	57.72
Precision	54.99	65.45
Recall	86.44	51.61
Accuracy	58.51	

Romney: LogReg	Negative	Positive
F-Score	68.32	31.34
Precision	53.14	77.77
Recall	95.62	19.62
Accuracy	55.40	

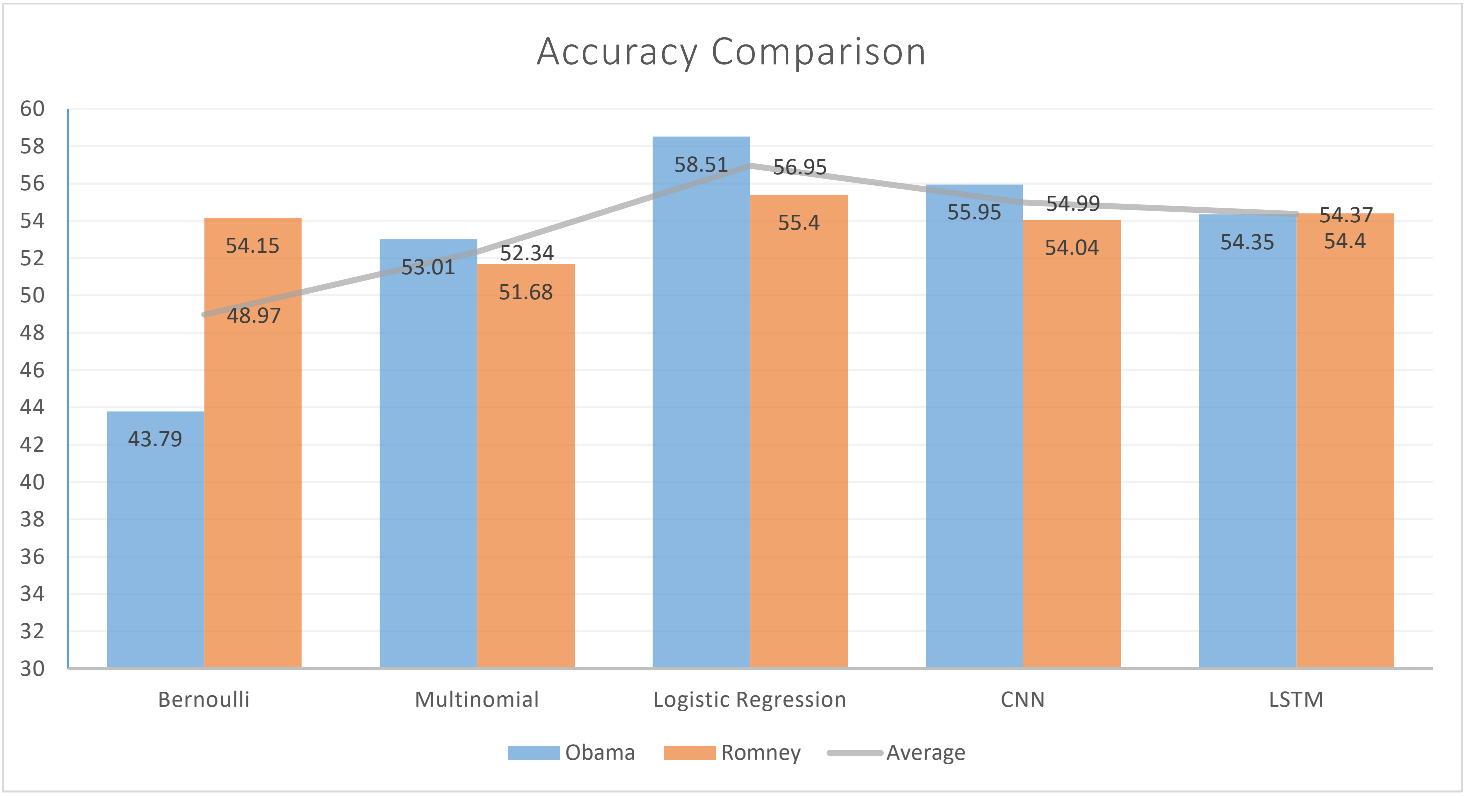
4.1.3. Neural Networks

Obama: CNN	Negative	Positive
F-Score	55.74	63.88
Precision	50.44	63.72
Recall	62.29	64.03
Accuracy	55.95	

Obama: LSTM	Negative	Positive
F-Score	54.78	58.58
Precision	53.36	60.10
Recall	56.28	57.14
Accuracy	54.35	

Romney: CNN	Negative	Positive
F-Score	66.44	36.63
Precision	61.70	42.04
Recall	71.98	32.45
Accuracy	54.04	

Romney: LSTM	Negative	Positive
F-Score	67.31	39.28
Precision	61.90	40.00
Recall	73.75	38.59
Accuracy	54.40	



5. References

[1] http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.BernoulliNB.html

[2] http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html

[3] http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[4] Peter Nagy, LSTM Sentiment Analysis | Keras, https://www.kaggle.com/ngyptr/lstm-sentiment-analysis-keras, 2017

[5] François Chollet, https://github.com/fchollet/keras/blob/master/examples/imdb_lstm.py

[6]Giuseppe Bonaccorso, https://gist.github.com/giuseppebonaccorso/061fca8d0dfc6873619efd8f364bfe89#file-twitter_training_output-txt

[7] Théo Szymkowiak, https://medium.com/@thoszymkowiak/how-to-implement-sentiment-analysis-using-word-embedding-and-convolutional-neural-networks-on-keras-163197aef623