Identifying ADR in Social Media: What Matters and Why?

1. Introduction

ADR (adverse drug reaction) refers to negative side effect caused by drug consumption. It is a major contributor of unnatural death worldwide. Both pharmaceutical companies and medication administration authorities are constantly searching for methods that can identify ADR accurately and efficiently. This report will focus on ADR identification through social media (twitter) and evaluate the performance of several ML algorithms as well as the proposed feature set.

2. Task description

This text classification problem tries to classify a given tweet into 1 of 2 categories:

- the tweet content is associated with ADR and classified as **Y**

- the tweet content does not contain ADR and classified as **N**

Below are some summary statistics of the dataset (which is a subset of the dataset used by the DIEGO Lab) [1]:

	Total	N	Y
train.txt	3166	2793 (88.219%)	373 (11.781%)
dev.txt	1076	962 (89.405%)	114 (10.595%)
Total	**4242**	**3755 (88.520%)**	**487 (11.480%)**
test.txt	1087	/	/

The above observation shows that the class distribution is severely **imbalanced**, Y/N ratio is almost 1/9 which is to say that by always predicting a given instance N, we will be correct 9 out of 10 times without much effort.

The baseline performance used in the report will be the dummy Zero-R classifier and we train on ***train.txt***, evaluate on ***dev.txt***. This is a good practice to prevent **data leakage** as in reality we do not have access to the test instances during training. We will also apply various rebalancing techniques to the training data to adjust the instance weights or class distributions and see how that will affect our result.

3. Related work

There is numerous amount of work dedicated to text classification. Some of them point out that pre-processing, includes stemming, lemmatizing, stop words removal and so on plays a key role in feature engineering. [2] Also, some previous work on mining ADR in tweets suggests that the task resembles but is not equivalent to sentiment analysis due to twitter's noisy, informal nature and some domain-specific properties of medicine and pharmacy. [3]

4. Methodology

4.1 Bag-of-words model

We present the evaluation results generated by **WEKA** [4] and the original 94 word-unigram feature set using the full training dataset and the rebalanced training set:

			N	**Y**
	ACC	**ROC**	**F-Measure**	
Full Training Set				
Zero-R	89.405	0.500	0.944	0
Naïve Bayes	82.156	0.758	0.896	0.364
SVM	89.591	0.555	0.944	0.200
Decision Tree	89.312	0.570	0.943	0.196
WEKA Class Balancer				
Naïve Bayes	71.468	0.758	0.818	0.340
SVM	82.249	0.707	0.896	0.401
Decision Tree	80.762	0.560	0.890	0.247
Random under-sampling of N				
Naïve Bayes	72.305	0.738	0.826	0.323
SVM	77.231	0.656	0.863	0.321
Decision Tree	71.933	0.664	0.825	0.284
Synthetic Minority Over-sampling				
Naïve Bayes	85.966	0.622	0.922	0.284
SVM	83.178	0.697	0.902	0.399
Decision Tree	88.011	0.645	0.935	0.168

The result shows that only SVM 'beats' the Zero-R classifier by a narrow margin in terms of accuracy when using the full training data. By balancing the dataset, we observe more correct Y predictions as well as misclassifications of N to Y. SVM and Decision Tree's F-score for Y is improved and in return the overall accuracy is worsen.

One of the advantages of SVM and Decision Tree is that they take the interactions between features into account and hence increase the model complexity and training time. Naïve Bayes, however, simple and fast because it assumes features are independent. This is invalid in our case since there are features that frequently occur jointly such as ***I am, gain weight,*** etc.

Another observation is that over-sampling tends to produce better results than under-sampling especially when dataset is small because when we under-sample we are throwing away real data and effectively trained on a smaller dataset.

We only performed resampling and reweighting once in this case, however, it is critical to do this repeatedly and try different weights or sample spread every time so that the optimal parameter setting can be found.

It is also a clever idea to use **cost-sensitive classifier** without changing the original data. In this case, the penalization of misclassifications of different classes can be adjusted by customizing the cost matrix.

Both NB and Decision Tree are easily interpretable, however they are of different natures. NB is **generative** and learns the class distribution through training while SVM and Decision Tree are **discriminative** and try to capture the "best" boundary between classes geometrically rather than probabilistically. Discriminative models tend to outperform generative ones in terms of accuracy. We can possibly train using a basket of learning algorithms rather than one by adopting more advanced methods such as Boosting, Bagging, Stacking, etc.

The evaluation metric also matters in this case and is application-specific. We could use accuracy but the class is not evenly distributed hence the measure is skewed. Or we could use for example recall on Y if we only care about identifying all ADR cases.

Bag-of-words model is often the first setup in text classification, however, the scalability of bag-of-words model is limited without feature selection. Also, stemming and lemmatization is a decision to make and often critically determines the size of the feature set. Furthermore, the order of words is not preserved [5] which could potentially be informative and non-alphanumeric characters are all discarded.

4.2 Feature engineering

What makes a tweet ADR present? We will consider the following aspects:

- **Basic**: tweets length, contain how many digits, punctuations (emoticons, exclamation marks), unigram, bigram, trigram of word tokens.
- **Subjectivity**:
  stating a fact: e.g. "*everything is a Cymbalta commercial.*"
  expressing an opinion: e.g. "*I've decided I'm really not happy with the pain lamotrigine is causing or the poor sleep.*"
- **Polarity**:
  extreme tone: (strong positive or negative, **intensifiers** like *very, really,* **emoticons**, **elongated words**) e.g. "*I feel siiiiiiiiiiiiiiick. Damn venlafaxine.*"
  or neutral
- **Part-of-Speech Tagging**: bigram and trigram of POS tags

We consider all the above by leveraging several Python packages to fulfil the task:

- **NLTK**, **scikit-learn, NumPy** for generating basic features and vector space model of n-grams of tokens
- Polarity and Subjectivity score returned by **Text Blob**, a python sentiment analysis package
- Compound polarity score returned by **Vader** [6] which is a sentiment analysis tool that smartly handles sentiment-laden slang words and acronyms, emoticons, etc.
- Bigrams and trigrams of POS tags produced by the **CMU ARK Twitter Part-of-Speech Tagger** [7]

word-unigram + word-bigram + basic				
			N	**Y**
	ACC	**ROC**	**F-Measure**	
Naïve Bayes	81.970	0.748	0.896	0.326
SVM	90.427	0.583	0.949	0.280
Decision Tree	89.591	0.555	0.944	0.200

POS tags + sentiment scores				
			N	**Y**
	ACC	**ROC**	**F-Measure**	
Naïve Bayes	82.435	0.786	0.898	0.359
SVM	89.312	0.561	0.943	0.218
Decision Tree	89.126	0.567	0.941	0.235

Due to the 3000+ feature size and the long training time, the feature set is split into 2 and we evaluate separately on subsets of features.

By adding bigrams and some basic features that are mentioned previously we manage to improve the performances of SVM and Decision Tree slightly. However, this is not the case for 2nd set, as Naïve Bayes is improved but not the other 2.

Notice that NB is more sensitive to new features as the probability measure may change due to new information, however, SVM and Decision Tree are more robust to new features since they automatically filter out uninformative features during model training.

After running some attribute selection algorithms including Gain Ratio, chi-Squared, classifier based, etc. We found out that some features clearly have predictive power e.g. Vader compound score, Text Blob polarity score, tweet length, number of punctuations in tweet, "*feel like*", "*make/made me/you*", "*Vyvanse make*", "*back pain*", etc.

Some terms are subject and verb combinations showing subjectivity and tendency of express personal feelings. Other attributes include ***pain, drowsy, headaches, crippled, etc.*** These terms are adjectives and nouns that are negative drug reactions themselves.

However, rather disappointing, majority of the bigrams and trigrams of the word tokens and POS tags are not adding any predictive value and the word-unigrams remain dominant in the feature set in terms of predictive power.

5. Summary

Here is a list of the observations and reflections over the experiments and some suggested improvements for future experimentation:

- Text classification task is challenging and especially when dealing with noisy text such as twitter
- Adding more features is tempting however the computation overhead is considerable
- Feature selection is critical especially when there are hundreds or thousands of features and you want to narrow down the range
- Sentiment analysis toolkits are indeed adding valuable features. However, majority of these classifiers are trained using for example movie reviews, etc. Ideally, we need large corpus from medical domain to train these classifiers to yield better results
- POS tags are adding very little value and are wasteful in terms of feature size

And finally, we attempt to answer the question that is listed in the title: what matters in ADR identification through twitter and why?

ML algorithm	**Generative** (NB, Markov) vs **Discriminative** (SVM, Tree-based, CRF)
Evaluation Metric	e.g. overall Accuracy, per class F-score, etc
Training Method and Parameter Tuning	e.g. resampling, reweighting, etc
Feature Engineering	e.g. pre-processing or not, frequency-based numeric or Bernoulli or Nominal attribute
Feature Selection	reduce the training time and noises

References

[1] Abeed Sarker and Graciela Gonzalez. (2015) *Portable automatic text classification for adverse drug reaction detection via multi-corpus training*. Journal of Biomedical Informatics, 53: 196-207.

[2] Asghar M. Z., Khan A., Ahmad S.and Kundi F. M., 2014. *A Review of Feature Extraction in Sentiment Analysis*. J. Basic. Appl. Sci. Res., 4(3): 181-186.

[3] Pastel, B. and Villanueva, B., *Discovering Adverse Drug Reactions via Natural Language Processing of Twitter Posts*.

[4] Eibe Frank, Mark A. Hall, and Ian H. Witten (2016). The WEKA Workbench. Online Appendix for "*Data Mining: Practical Machine Learning Tools and Techniques*", Morgan Kaufmann, Fourth Edition, 2016.

[5] Salton and McGill: G. Salton and M. J. McGill. Introduction to Modern Information Retrieval. McGraw-Hill, New York, NY, USA, 1983.

[6] Hutto, C.J. & Gilbert, E.E. (2014). *VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text*. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014.

[7] Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. *In Proceedings of the Annual Meeting of the Association for Computational Linguistics*, companion volume, Portland, OR, June 2011.