COMP90049 Project 2: Geolocation of Tweets with Machine Learning

San Kho Lin (829463) sanl1@student.unimelb.edu.au

1 Introduction

In this project, I am going to experiment machine learning classification on real world Twitter data. The objective of the project is to build a Geo-location classifier for Twitter users based on their Tweets.

2 Related Works

In my previous assignment, I experimented on identifying the misspelled location in Twitter user's tweets by using formally defined location names in a gazetteer. The approach was using different approximate string matching methods and it is mainly a data-driven text processing experiment. In this project, I apply what I have gained knowledge from previous project. Additionally there are model driven approach on the processed text data - namely supervised machine learning.

To begin with, I study on the prior research articles on the given problem. Geo-locating user solely based on their tweets is also known as Inferring Locations of Twitter User based on what they posted[1] or Content-based approach Geolocating Twitter User[2]. In these papers, the authors states on not only heuristic and empirical approach but also machine learning model driven approach to perform location estimation on the Twitter user.

In Cheng at el. paper [2], it particularly highlights on the important concept of identifying ***Local Word*** in Tweets and the authors characterize it as a decision problem. The key hypothesis is such that *a subset of words which have a more concrete geographical scope compared to other words in the dataset.* For instance, the word "rockets" has more affinity to people's tweets near Houston which is home of NASA and NBA basketball team Rockets. The paper continue discuss details on how to determine best on a local word from large noisy tweets corpus. The profound approach is such that a local word has high local focus and a fast dispersion. The authors states that relaying on formally defined location names in a gazetter is neither scalable nor provides sufficient coverage.

However, in Mahmud at el. paper[1] a combination of different heuristic approaches including consideration of a local word concept. The authors on this paper makes use of USGS gazetteer for place name identifier and a local word by finding probabilistic measure on number of users to each word occurrences.

Then, both paper discuss on machine learning model to select the generalize subset of words or terms to infer the Twitter user location for future unseen data. By studying these paper, I learn the idea on how to incorporate them into this project experiment – together with lectures learnt from the subject.

3 Project Work Details

In this project, it has provided the sample features that can be used with Weka machine learning tool as well as the raw tweets corpus. The raw tweets data is partitioned by hold out strategy for evaluation – 60% training, 20% each for development and testing. This is also to prevent the over fitting the data.

The concept of engineering a feature in machine learning is that how best the attributes generalize better into stages from training to unseen test data. It is an empirical observation to select the attributes that see fit to correctly classify the unseen data.

3.1 Preprocessing Raw Data

To carry out on engineering a feature, I start with pre-processing tweets raw corpus by removing URLs and @username patterns. It is the hypothesis that these patterns do not describe best to location estimation process. In all cases, tweets are tokenized, case is folded and the standard 319 stop words are filtered.

3.2 Determining Features

I use 3 approaches for determining the classifier for Geo-locating users from their tweets. First approach is using Part-of-Speech (POS) tagger to determine nouns and extract them from user tweets. Intuitively, most location names are in

nouns group. I use OpenNLP library as POS tagger and remove adjectives, verbs, prepositions, etc. because they are often generic and may not discriminate among locations[1].

In addition to POS tagger, I filter the English word stemming using Apache Lucene library. This will collapse the words into their original form in hope that it reduces the noise in corpus. The last approach is the place name classifier that make use of what I learnt from previous project. In addition to POS tagger, the term tokens are approximate string matching to USGS gazetteer dictionary using Levenshtein distance algorithm. Any match terms are collected for potential attributes for target classes. The key hypothesis for processing in addition to POS Noun tagger is that the locations are **name entity recognition** by nature – i.e. usually nouns.

I also observe that the raw tweet data are *previously identified* by locations Boston(B), Houston(H), San Diego(SD), Seattle(Se), and Washington DC(W). This could has been derived from the location coordinate from the original tweets data – accessible by Twitter API. As such, the collected terms are further analysis in Excel and term tokens are frequency counted during processing. In addition to term frequency, the number of user count on each terms are also collected. It is then sorted by user count followed by term frequency. Doing so in hope that it can infer the local words – what user talk most – for each city.

3.3 Determining Classifier

Initially, I experiment the supervised machine learning algorithms with Weka tool on the provided feature selection. Basically, the machine learning model steps that I try are as follow.

- Determine the baseline using Zero-R and One-R
- Then using *NaiveBayes*, *NaiveBayesMultinomial*, *k-NN* (Lazy IBk), *J48 Decision Tree* to further refine the model
- Use *10-Fold Cross Validation* to test the variance and bias of the classifier
- Then using *Ablation approach* to revisit the attributes, train and evaluate the model

It is a iterative work as it will go in cycles of select attributes, train and evaluate the output result until it comes to (at your) best classifier for the Geo-locating user from their tweets.

I observe that in terms of performance time complexity, the `k-NN` and `J48` Decision Tree are the longest to build the model. Most of the time, the Weka program crash and this highlights that these models require a better processing power and memory in order to have optimal operation. On the contrary, both `NaiveBayes` and `NaiveBayesMultinomial` are fast in building the model and optimally fit for this project training data size i.e. 214,880 lines of tweets.

3.4 Evaluations

For the evaluations, I use all the steps as aforementioned except k-NN and J48 due to computing limitation[1]. From feature engineering, each approach I collect top 100 terms of user frequency to term frequency order, except for Place Name classifier which is only sorted by term frequency due to partition run on long operation[2]. These 100 terms are further analyse in Excel to narrow down top 50 terms in **uniqueness by city order** and **highest term frequency order**.

For example Figure 1 shows the '*san diego*' is unique to SD class with 2600 counts while the below figure shows '*day*' as spread across all classes though having high frequency count. I also cross compare with the project given features selection over analysis. These top 50 terms for each feature classifiers (POS, Stem, Place) run with Weka. For example, Figure 2 shows the evaluation matrix with NaiveBayes Multinomial algorithm for top 50 classifiers for each approach. As in table, I can conclude that stemming approach is the lowest performer, while POS classifier with unique by city terms are the best outcome.

I then iterate a couple of cycle by using Ablation approach in couple with 10-Fold Cross Validation test to empirically pick the top performing attributes from POS Classifier Unique and Place Name Classifier Unique feature set to come up with the final best 40 attributes. Figure 3 shows the evaluation output with different

[1]Weka crash often and take long time. Max try over nights come up with memory error.

[2]Each token will compare against dictionary entries for Levenshtein distance. Further technical limitation details in README.txt

machine learning algorithms.

4 Conclusions

In conclusion, I observe that the key element for a better classifier model outcome is how good the feature generalize from training data to test data or future unseen data. The supervised machine learning algorithms contribute a little or none for the output classifier. The model will be as good as it is for a good features set. The algorithms are there to help as a mechanism for predicting future unseen data in most none-bias and undiscriminating way possible.

From the Kaggled competition, I can infer that my approach still has room to improve. My feature engineering approach still might be required to represent the problem better as well as a more in depth learning on machine learning algorithms might be still required. Even though my understanding is right, there might also be bug in program code in such that it produces the other way.

By experimenting this project, it give me the realization of the profound thinking and approach to get deep understanding of the problem domain and, how best to represent the problem feature is the great challenge. Critical analysis on the data, transform it into the information, which then using machine learning technique to discover unseen facts or make future prediction is the key knowledge task lesson learnt in studying this subject.

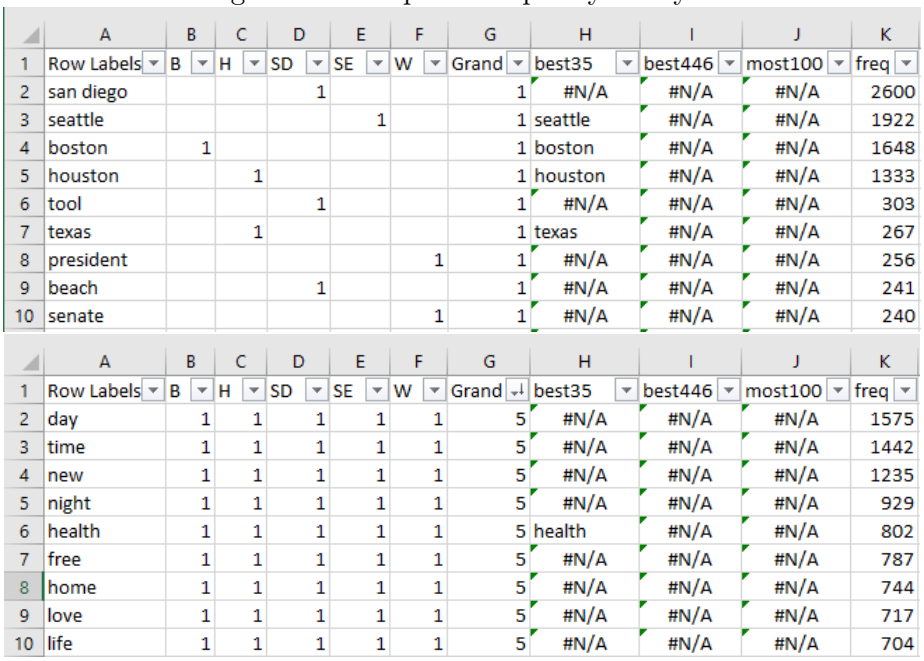
5 References

[1] Jalal Mahmud et al. "Where Is This Tweet From? Inferring Home Locations of Twitter Users". In: *In Proceedings of the 6th International Conference on Weblogs and Social Media (ICWSM'12)* (2012).

[2] Zhiyuan Cheng et al. "You Are Where You Tweet: A Content Based Approach to Geo locating Twitter Us ers". In: *In Proceedings of the 19th ACM international conference on Information and knowledge management (CIKM '10)* (2010).

6 Appendix

Figure 1: Excerpt of Frequency Analysis



Row Labels	B	H	SD	SE	W	Grand	best35	best446	most100	freq
san diego			1			1	#N/A	#N/A	#N/A	2600
seattle				1		1	seattle	#N/A	#N/A	1922
boston	1					1	boston	#N/A	#N/A	1648
houston		1				1	houston	#N/A	#N/A	1333
tool			1			1	#N/A	#N/A	#N/A	303
texas		1				1	texas	#N/A	#N/A	267
president					1	1	#N/A	#N/A	#N/A	256
beach			1			1	#N/A	#N/A	#N/A	241
senate					1	1	#N/A	#N/A	#N/A	240

Row Labels	B	H	SD	SE	W	Grand	best35	best446	most100	freq
day	1	1	1	1	1	5	#N/A	#N/A	#N/A	1575
time	1	1	1	1	1	5	#N/A	#N/A	#N/A	1442
new	1	1	1	1	1	5	#N/A	#N/A	#N/A	1235
night	1	1	1	1	1	5	#N/A	#N/A	#N/A	929
health	1	1	1	1	1	5	health	#N/A	#N/A	802
free	1	1	1	1	1	5	#N/A	#N/A	#N/A	787
home	1	1	1	1	1	5	#N/A	#N/A	#N/A	744
love	1	1	1	1	1	5	#N/A	#N/A	#N/A	717
life	1	1	1	1	1	5	#N/A	#N/A	#N/A	704

Figure 2: Top 50 - NaiveBayes Multinomial Analysis with Different Classifier

Top 50 - NaiveBayes Multinomial					Weighted Avg		
	Correctly Classified Instances		Incorrectly Classified Instances		Precision	Recall	F-Measure
Place Name Classifier	19742	28.81%	48790	71.19%	0.452	0.288	0.189
Place Name Classifier Unique	20147	29.40%	48385	70.60%	0.506	0.294	0.187
POS Classifier	19248	28.09%	49284	71.91%	0.36	0.281	0.225
POS Classifier Unique	20594	30.05%	47938	69.95%	0.436	0.301	0.21
Stem Classifier	19869	28.99%	48663	71.01%	0.367	0.29	0.199
Stem Classifier Unique	19544	28.52%	48988	71.48%	0.348	0.285	0.192

Figure 3: Top 40 - Classifier with Different Machine Learning Algorithms

Top 40 Classifier					Weighted Avg		
	Correctly Classified Instances		Incorrectly Classified Instances		Precision	Recall	F-Measure
NaiveBayes Multinomial	21005	30.65%	47527	69.35%	0.605	0.306	0.206
J48	19888	29.02%	48644	70.98%	0.285	0.29	0.276
NaiveBayes	19638	28.66%	48894	71.34%	0.556	0.287	0.164
OneR	15772	23.01%	52760	76.99%	0.225	0.23	0.225
ZeroR	17929	26.16%	50603	73.84%	0.068	0.262	0.108