COMP90049 assignment2 Report

Anonymous

1 Introduction

The purpose of this project is to predict geolocation of tweets by machine learning. In this paper, I will mention dataset and introduce some relevant literature firstly. Then, I will analyze feature engineering and explain the machine learning algorithms I used. Lastly, I will evaluate each method by Precision, Recall and F-Measure.

2 Dataset

There are several ARFF format datasets and row tweets datasets compiled by Eisenstein and Jacob (2010), Rahimi, Arshin, Trevor Cohn and Timothy Baldwin (2018) collectively.

1) "train/dev/test-most XX.arff"

These documents record instances with the most frequent appeared XX features.

2) "train/dev/test-best XX.arff"

These documents record instances with the top XX features filtering by mutual information.

3) "train-most/best XX.arff"

These documents are used as the training set, it has 96585 rows instances. For each instance, it has attributes including tweet-id, user-id, the number of XX features, and class.

4) "dev-most/best XX.arff"

These documents are used as the development set, it has 34028 rows instance. I use the datasets to evaluate the performance of the models.

5) "test-most/best XX.arff"

These documents are used as the test set, it has 32977 rows instance. These datasets do not have class, which should be predicted by ourselves.

3 Relevant literature

In the paper of Han and Cook (2014), they introduce "location indicative words" (LIWs), which can help classify the tweets. For example, the word "unimelb" are more likely to be appeared in the tweets from Melbourne because "unimelb" is the abbreviation of the University of Melbourne.

In addition, Sandve (2016) stated that "time zone" and "UTC-offset" can also be helpful. Different regions are in different time zones, and they may have different peak hours. For example, America and Australia are in different time zones, so the peak hours are also different, through which can help deduce the location of tweets.

4 Data preprocessing

There are several steps to preprocess data.

1) Firstly, I stored the data in csv format, which can be processed conveniently by DataFrame in python.
2) Then I found tweets which have the same user ids are all from the same locations. Therefore, I put together the feature messages from the same users on twitter, which means I do the work of predicting the geolocation of twitter users. I can have more useful feature information if I predict the geolocation of users than predict the geolocation of twitters.
3) Thirdly, I dropped the feature "tweet-id" and 'user-id' which are not used in model.
4) I found the frequency of stop words are high, so I deleted the features such as "is" and "are" …

5 Feature engineering

5.1 Pointwise Mutual Information

The pointwise mutual information is a measure of dependence of A and C. The formula is:

$$PMI(A,C) = \log\left(\frac{p(A,C)}{p(A)p(C)}\right)$$

If A and C are independent, we can get:

$$p(A,C) = p(A) * p(C)$$

It means $\log\left(\frac{p(A,C)}{p(A)p(C)}\right) = 0$

Therefore, we can get the conclusion that the greater the pointwise mutual information, the more relevant between A and C. In the paper, A means each feature, and C means one of the three classes. I can select the features with the highest pointwise mutual information.

5.2 Information Gain

Information gain is the difference of entropy between the phase (before splitting) and the phase (after splitting). The formula is:

$$IG(RA|R) = H(R) - \sum_{i=1}^{m} P(xi)H(xi)$$

From the formula, we can see if splitting a particular feature can get the biggest information gain, the feature can be considered it is more relevant with the class.

5.3 Chi-square

Chi-square is also a measure of dependence of A and C, which is similar with pointwise mutual information. The formula is:

$$\chi^2 = \sum_{i=1}^{r}\sum_{j=1}^{c}\frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}}$$

$O_{i,j}$ are observed values and $E_{i,j}$ are expected values, if all $O_{i,j} = E_{i,j}$, the $\chi 2$ is equal to 0, which means the two variables are less relevant, that is why I choose features with highest $\chi 2$.

6 Machine learning algorithms

The choice of algorithms is very important because different algorithms apply to different data. For example, SVM may not be suitable for this problem because the number of features and the number of instances is too big. If I used SVM as the model, it would run slowly. Furthermore, SVM is sensitive to outliers, which will lead to poor performance if there are a certain number of outliers.

6.1 Naïve Bayes

Naïve Bayes is a good model to process document classification and it is fast to run. Naïve Bayes will compare the probabilities that the instance belongs to different classes, and determine the instance belongs to the class with highest probability. However, when choosing the features, it needs to make sure that each feature is independent, which is the basis of the algorithm.

6.2 Decision tree

Decision tree is also a suitable algorithm for this problem. As mentioned early, I use information gain to select features, which is also a part of in decision tree algorithm. When building a decision tree, it will choose the feature with the biggest information gain for each new node. However, if simply use the decision tree to train the model, it would cause overfitting problem because the algorithm will record every attribute for every instance. To prevent this phenomenon, it can use pre-pruning or post-pruning technologies to delete some irrelevant trees.

6.3 Logistic Regression

Logistic regression is also a good algorithm because it applies to big data and is effective to train. Moreover, it tries to find the optimal solution. However, it is also easy to overfit, which can be reduced by regularization. It is worth noting that logistic regression applies to binary classification problem. For this problem, it should execute the algorithm three times (predict to New York vs predict to California, predict to New York vs predict to Georgia, and predict to Georgia vs predict to California).

Table 1: different feature engineering with Naïve Bayes

	PMI	IG	Chi-square
Precision	84.15%	81.39%	81.32%
Recall	75.78%	78.01%	68.75%
F-Measure	79.75%	79.66%	74.51%

Table 2: machine learning algorithms with PMI

	Naïve Bayes	Decision Tree	Logistic Regression
Precision	84.15%	64.36%	82.43%
Recall	75.78%	50.31%	71.65%
F-Measure	79.75%	56.47%	76.63%

7 Evaluation

In this paper, I choose Precision and Recall (based on multi-class confusion matrix) as evaluation metrics because they can evaluate model in different ways. Due to Precision and Recall are both important, I also introduce F-Measure to measure the performance comprehensively.

In order to compare the performance of different feature engineering methods, I select top 300 terms (top 100 terms of each class) for each method, and the result can be seen in table 1. With respect to F-Measure, PMI and IG have the similar performance, and they are better than Chi-square. For Precision, PMI does the best work at 84.15%, which is better than other two methods. I think the reason why PMI has the highest Precision is that when I select top 100 terms of three classes with PMI, there is no repeated terms (each class has completed different top 100 terms), which means for each feature, it has more relevant with one class and less relevant with others. However, for IG and $\chi 2$, there are some reduplicated terms.

In regard to compare different machine learning algorithms, I use the top 300 terms from PMI as the dataset. From the table 2, Naïve Bayes and Logistic regression are better than Decision tree. Naïve Bayes has the highest F-Measure, at 78.55%, while the F-Measure for Logistic Regression is marginally less than it. However, the performance of Decision tree is poor, maybe I should try random forest algorithm to instead or introduce tree prune technology.

I think the performances of the classifiers are not good enough. The reasons are as follows.

1) The data is sparse, for each instance, it only has a few features.
2) Bag of words (BOW) only consider the number of occurrences of words in the vocabulary but does not consider order. It ignores the relationship of context and cannot represent semantic information.

8 Conclusions

As a conclusion, I preprocess the data firstly. Then I use pointwise mutual information, $\chi 2$ and information gain to do feature engineering. Lastly, I use Naïve Bayes, Decision tree and Logistic regression to predict the label of test data. For the future improvement, I think N-Gram can be added to select features because it can express the information of word order. Furthermore, I think TF-IDF is also a good choice to find key words.

References

Eisenstein, Jacob, et al. A latent variable model for geographic lexical variation. Proceedings of the 2010 conference on empirical methods in natural language processing. Association for Computational Linguistics, 2010.

Han, B., Cook, P., & Baldwin, T. (2014). Text-based twitter user geolocation

prediction. Journal of Artificial Intelligence Research, 49, 451-500.

Rahimi, Afshin, Trevor Cohn, and Timothy Baldwin. Semi-supervised user geolocation via graph convolutional networks. arXiv preprint arXiv:1804.08049 (2018).

Sandve, S. (2016). A Metadata Approach to Predicting Twitter User Geolocation. NTNU, Trondheim, Norway.