COMP30027 Project2 Report

1. Introduction

The task of this Project is to automatically identify the location from which a textual message was sent, as one of four Australian cities. Certain features are extracted from the training sets given and are put into different machine learning classifiers. Development datasets are used as test sets to find the best classifier by getting and comparing the accuracy. In this case, Decision Tree is chosen as the most accurate and effective classifier.

2. Related Literature

There have been several approaches in the past that have focused on various levels of geo-location estimation. For example, J. Luo, D. Joshi, J. Yu, and A. Gallagher did detailed surveys of geotagging social media content and discussed a variety of geotagging approaches in [1]. Zamir and Shah, in [2], propose a novel variant of the Nearest Neighbor algorithm using Generalized Minimum Clique Graphs (GMCPs) to exceed state-of-the-art results in geolocalization. This initiative built upon their previous work of matching using SIFT features and trees [3], to perform fine-grained localization within cities.

In this project, the raw data has been pre-processed by Pointwise Mutual Information(PMI) to give the train data and development data. Pointwise mutual information (PMI, 5) is a measure of how much the actual probability of a particular co-occurrence of events p(x, y) differs from what we would expect it to be on the basis of the probabilities of the individual events and the assumption of independence p(x)p(y). Gerlof Bouma concluded that PMI performs relatively well as an association measure in those cases where bare occurrence frequency does not in [4].

3. Technical Approach

3.1 Learners

Three learners are tested in this project: Naïve Bayes; Decision Trees; and Support Vector Machines(SVM).

With the use of Naïve Bayes, a probabilistic model of the training data is built, and then use that to predict the class labels of the test data. In this case, raw data has been pre-processed by using PMI and token and hence the test instances can be compared and predicted directly.

Decision Trees are used to classify novel instances by traversing down the tree and classifying according to the label at the deepest reachable point in the tree structure (leaf). They are fast to train and even faster to classify.

Support Vector Machines(SVM) is a non-probabilistic binary linear classifier. A single binary, max-margin SVM is built for each class that determines whether a given tweet belongs to that class or not. During testing, we then see which SVM outputs the greatest score for that example, and choose the corresponding label as our prediction.

3.2 Features

The features used in this project are mainly in two aspects:

One aspect is the data sets given. The raw data sets are all pre-processed by PMI and token to give the top 10, top 50, and top 100 data sets for training, developing and testing.

The other aspect is that I wrote a featureEng() function to generate two features dictionary, which are username dictionary and location dictionary. I used regular expression to find all the location string (format @location) and user name string (format @username). It will remove some low frequency features according to the number given in the parameter. Another function is wrote to compute the number of the features in each single sample. It will return an array that record the count of each features in all the samples (the sample's ID must in the IDlist). Since the amount of name tags and location tags are too much in the training sets, I only chose the name tags that appear for more than 20 times (about 500 tags) and location tags that appear equal or more than 2 times (about 300 tags).

4. Evaluation

4.1 Data

I separated the given data into two parts: The first part contains the data those values for all the attributes are 0s and the second part contains those values are not all 0s.

For the non-all-0 data. Since the pre-processed vectors is already very useful to predict them. The accuracy of the prediction of this part of data is about 50%-53%. I just added a small amount of the location features in the existing data to make a high correction rate. After the engineering features, the accuracy of this part reached 55%.

For the all-0 data. Since the pre-processed vectors are all 0, the accuracy of the prediction is fixed at 25% no matter what kind of classifier is used. Since the all-0 data has very few meaningful data, thus I used extra features to predict them. These features include: location features and @username features. After the engineering features, the accuracy of this part of the data reached 26.5%.

4.2 Classifiers

I tried three classifiers to predict the results: Naïve Bayes, Decision Trees, and Support Vector Machines(SVM).

Naïve Bayes gives an accuracy at about 30.5% but the results are not very stable and the difference between the results are quite large (+- 1%). Another drawback of this classifier is that it takes up too much memory and this always leads to memory error.

```
0.5285171102661597

/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:475:
int64 was converted to float64 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)

0.261985844017094
0.3145835566513024
```

Figure 1 [5]

Decision tree gives an accuracy at about 32%. The time taken is not too long and it does not take up too much memory. It is chosen as the final classifier to predict the result.

```
0.5444052145573058
/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:475:
int64 was converted to float64 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
0.26178552350427353
0.31755815199914245
```

Figure 2 [5]

Support Vector Machines(SVM) gives an accuracy at about 29%-30%. The reason for this might be linear svc does not work well with data that are not so clear in their signature. The other problem with this classifier is that the complexity is too large and it cannot run properly if we use all the data given. I can only choose the first 10thousands lines of data from the 100thousands of data to train and it will lead to inaccurate results.

Based on the results above, Decision Tree is the most effective classifier to predict the location based on a textual message.

5. Error analysis

When using the linear svc classifier, only 10000 lines of data is used to train since it takes too much time and memory to run all the data given. A large amount of data is not trained therefore the prediction is just based on the small amount of data. This will cause a quite large difference in the result.

6. Conclusion

The final accuracy I got for the prediction is about 31.5% with the Decision Tree classifier.

During this project, I found out some points that I did not noticed in the past. For example, for sparse data, pre-processing is necessary to remove some data with very low value. Otherwise it will take too much time if you just put them into the system. Another point is that linear svc is not suitable for data whose characteristics are not obvious and whose amount is large.

Word count: 1129

Bibliography

[1] J. Luo, D. Joshi, J. Yu, and A. Gallagher, “Geotagging in multimedia and computer visiona suevey”, Multimedia Tools and Applications, vol.51, no.1, pp. 187-211, 2011

[2] A. R. Zamir and M. Shah. Image Geo-localization based on Multiple Nearest Neighbor Feature Matching using Generalized Graphs. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2014.

[3] A. R. Zamir and M. Shah. Accurate Image Localization Based on Google Maps Street View. Proceedings of the 11th European conference on Computer vision, 2014.

[4] Gerlof Bouma. Normalized (pointwise) mutual information in collocation extraction**.** Proceedings of GSCL, 2009

[5] For Figure 1 and Figure 2, the first line is the accuracy for the non-all 0 data, the second line is the accuracy for the all 0 data, the third line is the accuracy in total.