COMP30027 Machine Learning Project 2 Report

TOR

1 Introduction

In modern era, the Internet is being widely used and the number of users on it is tremendous. The widespread use of Internet leads to the booming development of Internet applications. Blog, for instance, is one of the most popular application people would use to communicate with each other and it is also a way to express the personal feelings. Based on such excessive amount of users, the variety of ages of users using blog is unprecedented. Although most of the users provide true demographic information, a little amount of users would prefer not to diclose their real age to the website. This incurs the difficulty of recognising age groups which can be used commercially for advertisement services. We attempted to build a system to appropriately recognise the age group of a given blog author by extracting features and genres from their writtings.

This report described the study and machine learning approaches taken to build various distinct systems, including one stacked system, to classify the age group of blog authors on the Internet. In the following sections, we will introduce the overall assumptions made on the datasets we used as well as the preprocessing feature extraction from the datasets, following by two sections describing various techniques (including the method we abandoned with a brief description of the reason) we applied to build the system and the limitations of these methods on different datasets. Eventually, we will conclude our work and suggest what can be improved further.

2 Assumption

The most essential assumption we made to use the dataset is that all of the demographic information provided in the dataset is correct and unbiased (Schler et al., 2006). However, in real world application, a little portion of such information is probably spurious. For ease of building the systems, we will not preprocess the dataset to determine the correctness of information provided. The blog writer is assumed to use consistent genre throughout all blogs he/she wrote.

3 Datasets

The datasets (Schler et al., 2006) provided are divided into three types which are "train", "dev" (development) and "test". As inidicated by their names, "train" will be used to build the systems, "dev" (development) is used to do self-evaluation and "test" is the final test of the systems. Each of these datasets was further splitted into two different datasets which are "raw" and "top10". Therefore, we used two separated final systems to handle these different files.

3.1 Raw

Each of the raw datasets contains six columns, which are user IDs, genders, ages, occupations, star signs, dates and the raw texts respectively. However, since the test set has marked all genders, ages, occupations, star signs and dates as missing value, these columns are dilberately ignored when we build the system for raw dataset. A noticeable feature in user ID column in raw datasets is that there are many duplicated user ID for different instances and the corresponding ages are identical, which means they can be grouped together to create new datasets to be fitted into the same system and compare the performances. We will further discuss the effect of combining same user IDs in the following sections.

3.2 Top10

The top10 datasets contains words with top 30 frequencies count extracted and selected from the raw dataset. All non-alphabetic characters are removed and all of the words are converted to lower case. They are assigned with distinct instance IDs as the first columns, and the large columns are mapped into different age groups instead of using the actual ages. The age groups are "14-16", "24-26", "34-36", "44-46" and "?" (for none of the above) respectively. The expected output from our system is required to be in such representation as well. All of the methods and systems discussed in this report starts from an initial test on these top10 datasets.

3.3 Preprocessing

3.3.1 Raw

The user IDs was combined and all texts belongs to the same user were join together to form a complete vocabulary of such user. After combining the user

IDs, the number of instances decreases dramatically from 276415 to 1496. We use TF-IDF (Term Frequency Inverse Document Frequency) vectoriser [1] with sublinear term frequency (

$$\hat{tf} = 1 + log(tf) \quad (1)$$

as the scaling to update the term frequency) and smooth inverse document frequency (add 1 to each freqeuncy to prevent zero division) to preprocess the given blog texts in the training set. The vectoriser was set to extract unigrams only with no maximum features extraction limitation and it produced a full vocabulary with 381595 words. Another vectoriser, count vectoriser [2] , we considered intially was abandoned. The documents (specifically, blogs) provided have various length in both train and development set. Using a count vectoriser cannot generalise the real word circumstance which different people have different preferences of the length of the blogs they wrote. We also attempted to replace the exact ages in the raw dataset by the age groups.

3.3.2 Top10

The instance ID column in top10 dataset is replaced by extracting the user ID column from the raw dataset, then combine the user IDs as we did when preprocessed the raw dataset. The frequency of each word in the selected top 30 word list is averaged across different instances link to the same user ID. We also attempted another way to reduce instances and it will be discussed and compared to the original method of combination.

4 Systems for Top10

Different classification methods are used for initial test of performance on the datasets. We test the performance of these classifiers [3] on training data before and after combining the user IDs using the mechanism described in the preprocessing section.

4.1 Before Combining

The five different classifying methods which are used to built the systems are the k-nearest neighbours classifier, Gaussian naive bayes classifier, random decision forest classifier, multilayer perceptron classifier and adaptive boosting classifier with decision tree base (see Table 1 for accuracy scores of each classifiers). It is obvious that all of the classifiers are not performing well on the plain top10 dataset. The submitted output of test set (using multilayer perceptron classifier since it holds the highest score amongst all five classifiers), whose score on test set via Kaggle is approximately 0.4, also reflects the poor performance.

4.2 After Combining

From the result above, we have spotted that some of the instances are predicted to two age groups but they are linked to the same user ID. Alongside with the earlier observation with respect to the duplicated user IDs, a decision was made to combine these user ID to produce new datasets (as described in the preprocessing section) from the given datasets. The training and development set are both reproduced by mechanism described previously, and the predicted result will be converted back to be with the same shape as the original dataset using a user to instances mapping. In this run, we replaced the Gaussian naive bayes classifier and random decision forest classifier by gradient boosting classifier and logistic regression classifier. The reason is that Gaussian naive bayes has the worst performance (it is reasonable since there are an enormous number of zero counts in the top10 datasets) and random decision forest is worse than adaptive boosting (since they are both decision tree base). Unfortunately, the resulting accuracy scores are worse than before (see Table 2). These results are out of our expectations, since we predicted that the accuracy score would increase by this mechanism. From a printed table of predicted versus actual labels, we noticed that combining the user IDs will cause incorrect predictions on all instances linked to the same user if the original prediction to the processed dataset is incorrect. Therefore, making predictions as per instance is more appropriate to be used when the system has poor performance.

Classifier	Score
k-NN	0.4303
Gaussian Naive Bayes	0.4160
Random Decision Forest	0.4336
Multilayer Perceptron	0.4346
Adaptive Boosting	0.4337

Table 1: Accuracy score before combining

Classifier	Score
k-NN	0.4187
Gradient Boosting	0.4139
Multilayer Perceptron	0.4226
Adaptive Boosting	0.4119
Logistic Regression	0.4265

Table 2: Accuracy score after combining

4.3 Ensemble (Stacked)

As required, we build a stacked system by selecting three systems above. We run different combinations of three classifiers and one meta-classifier and chose the best result. Eventually, the adaptive boosting classifier is selected to be the meta-classifier and the three classifiers are k-nearest neighbours classifier, multilayer perceptron classifier and logistic regression classifier. We use an encoder [4] to encode all

[1] using sklearn.feature_extraction.text.TfidfVectorizer
[2] sklearn.feature_extraction.text.CountVectorizer
[3] models in different submodules from sklearn
[4] using sklearn.LabelEncoder

labels into numerical features to be fitted into the meta-classifier. The result is 0.4187 for the self-test set, and submission to Kaggle with this system has a score of approximately 0.41 for the test set, which is a minor improvement from the previous attempts but this is insufficient.

4.4 Extra

In this attempt to reduce the number of instances, all of the instances are grouped by the age groups regardless of the user IDs, and the counts of each of the top 30 words are summed and normalised to reflect that the probabilty of using such word for each age group. The accuracy score increases to 0.5219, which is an exceptional increase from all previous results.

4.5 Top10 Bottleneck

We used many different mechanism (which were removed from the notebook) to improve the performance of system applies to the top10 datasets. However, the highest possible accuracy we obtained is from the extra group by age groups mechanism in the previous section and most of the other methods were all scored approximately 0.42 in our self-test runs. We concluded that this is the hard limitation of using top10 datasets. The top 30 words selected are separated from the context of the blogs and many words are ignored during this process of feature extractions. Therefore, the raw datasets are used and the preprocessing mechanism is described previously.

5 System for Raw

We applied SGDClassifier [5] as the classifier for submission. The three more classifiers, random decision forest classifiers, logistic regression classifier and k-nearest neighbours are removed as they are scored comparatively lower and run slower than the SGDClassifier. Using SGDClassifier can also take the advantages of stochastic gradient descent learning so that the classifier can explore the optimal model for classification.

5.1 Before Replacement

We scored the accuracy for predicting before and after the reduction of instances by aggregation over user IDs, which are 0.4825 and 0.6191 respectively. This displayed the usefulness of combining different instances based on their user IDs. The vectoriser provided the term frequencies and inverse document frequencies as per user instead of an individual instance (speicifically, blog), which can further identified the characteristics (specifically, writing styles) of the corresponding blog writer. The score received from Kaggle is an accuracy score over 0.6, which is a tremendous improvement from all of the previous results. We further tuned the hyper parameters of SGDClassifier to obtain an optimal result. This is a reflection of our previous conclusion that the provided features extracted from the raw datasets limits the performance of classifiers.

[5] linear model with stochastic gradient descent learning, which is sklearn.SGDClassifier

5.2 After Replacement

Similarly, we obtained the accuracy scores of 0.5161 and 0.6413 respectively. It performs better on development set, but, in fact, the score from Kaggle is slightly lower than the previous one. In the previous system, it is fitted by the exact ages as its expected output. The final predicted output was wrapped into different age groups so that an error tolerance mechnism was provided and in the real world the actual (biological or psychological) ages are difficult to be distinguished defenitively. Incoporating a larger number of texts will increase the uncertainty (formally, entropy) of data, and it will lose the potential error tolerance system as in the previous mechanism. Nevertheless, the difference on performance is very small. We claim that it performs as good as the previous one.

6 Error Analysis (Extra)

We noticed that some of the ages (in raw dataset) or age groups (in development dataset) were marked as missing value and some of the ages are not included in the age ranges we need to discussed. All of these information is, unfortunally, excluded from the training set deliberately. Using a regressor (removed since it requires an extremely long time to converge) instead of classifier will incur a disastrous decrease of performance. Therefore, we considered this as an unavoidable error and we only concentrated on improving predictions over valid data.

7 Conclusions and Potential Improvements

Throughout the whole progress, various systems were built and tested and two of the systems were satisfactory (the two systems built based on raw datasets). However, the accuracy is still insufficient to be used in real world applications. There are many other mechanism we do not attempt to use. One possibility is to use neural network to build a model for prediction based on the vectoriser. The words can be converted back to its root word before vectorising the document as well. A more advanced mechanism to further improve the blog is to use recurrent neural network (Moon and Liu, 2016) with word embeddings which is advantageous to natural language processing.

References

Tim Moon and Eric Liu. 2016. Using feedforward and recurrent neural networks to predict a bloggers age.

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James Pennebaker. 2006. Effects of age and gender on blogging. *Proceedings of 2006 AAAI*

Spring Symposium on Computational Approaches for Analyzing Weblogs.