COMP30027 Machine Learning Project 2 Report

Anonymous

1 Introduction

In modern era, the Internet is being widely used and the number of users on it is tremendous. Tthe 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, then 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. 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.

3 Datasets

The datasets (**?**) provided are divided into three types which are "train", "dev" and "test". As inidicated by their names, "train" will be used to build the systems, "dev" is used to do self-test 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 distinguish final systems to handle these.

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, occupations, star signs and dates as missed value, these columns are dilberately ignored when we build the system for raw dataset. A noticeable feature in user ID column is that there are many duplicated user ID for different instances and the corresponding ages are identical. We will further discuss the effect of combining same user IDs in the following sections.

3.2 Top10

The top10 datasets contains top words with top 30 frequencies count extracted 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 in such representation as well.

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 TD-IDF vectoriser [1] with sublinear term frequency (that is, using

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

as the scaling to update the term frequency) and smooth inverse document frequency (that is, add 1 to each freqeuncy to avoid zero division) to preprocess the given training texts. The vectoriser was set

[1] using sklearn.feature_extraction.text.TfidfVectorizer

to extract unigrams only and it produced a full vocabulary with 381595 words. 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 which have 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 [2] on training data before and after combining the user IDs using the mechanism described in the preprocessing section.

4.1 Before Combining

The five classifying methods that is 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). The classifiers are not performing well. The submitted output of test set (using multilayer perceptron classifier), whose score on test set via Kaggle is approximately 0.4, also reflects this.

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

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. Therefore, a decision was made to combine these user ID to produce new fresh datasets (as described in the preprocessing section) from the given datasets. When the classifier was trained and predicted, the result will be converted back to the same shape as the given 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 and random decision forest is worse than adaptive boosting (since they are both decision tree base). 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. However, making predictions based on each instances will not have this problem.

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

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 meta-classifier is chosen to be an adaptive boosting classifier and the three classifiers are k-nearest neighbours classifier, multilayer perceptron classifier and logistic regression classifier. We use an encoder [3] to encode all labels into features to be fitted into the meta-classifier. The result is 0.4187 for the self-test set, and this second submission via Kaggle has a score of approximately 0.41 for the test set, which is a minor improvement from the previous attempts.

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.

5 System for Raw

We applied SGDClassifier [4] as the classifier for submission. The three more classifiers, random decision forest classifiers, logistic regression classifier and k-nearest neighbours are deleted from the Jupyter notebook as they are scored comparatively lower and run slower than the SGDClassifier.

5.1 Before Replacement

We scored the accuracy for predicting before and after the reduction of instances, which are 0.4825

[2] models in different submodules from sklearn

[3] using sklearn.LabelEncoder

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

and 0.62 respectively. This displayed the usefulness of combining different instances based on their user IDs. The vectoriser provided the term frequencies and inverse document frequencies based on the users instead of individual blog, which can further identified the characteristics (specifically, writing styles) of the corresponding blog writer.

6 Conclusions and Potential Improvements

Conclusion.