COMP30027 Machine Learning Report

1. Introduction

The Internet has vastly reshaped our way of communication. While all internet users do have the option to remain anonymous about whatever they post, it might still be possible to identify some information about someone based on their writings, which in this case is ‘age group’. The author explores several machine learning techniques and aims to find the viability of author age group identification based on an author’s online posts.

2. Datasets

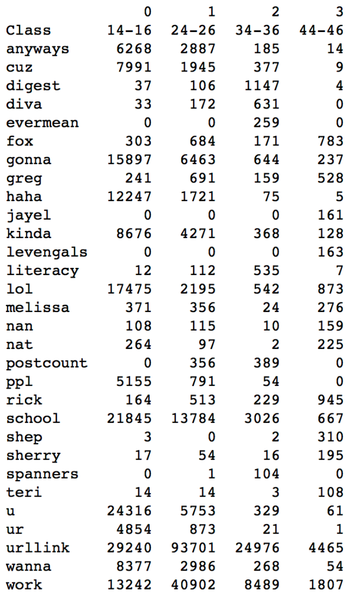
This study is conducted based on the dataset provided in ‘Effects of Age and Gender on Blogging’ (Schler et al (2006)). The training dataset contains nearly 270k entries, each includes User ID, Gender, Age, Occupation, Star Sign, Date and Text. Features are in form of unigram words, selected from the all text available using mutual information (MI). Only the top 30 features with the highest MI are kept and serve as the basis of our classifier.

2.1. Basic analysis of the dataset

A basic summary of the training dataset is shown on the right.

Each class has a total number of instances 98454, 141104, 30347, 6510 respectively. The distribution of classes is very imbalanced. There are more instance of twenties and very few instances of forties.

The development dataset has a number of 13100, 17298, 2584, 551, 11799, for classes of ‘14-16’, ‘24-26’, ‘34-36’, ‘44-46’ and ‘?’ respectively. There are 45332 instances in the dataset. The distribution of known classes is similar to the training data.



```
                0      1      2      3
Class       14-16  24-26  34-36  44-46
anyways      6268   2887    185     14
cuz          7991   1945    377      9
digest         37    106   1147      4
diva           33    172    631      0
evermean        0      0    259      0
fox           303    684    171    783
gonna       15897   6463    644    237
greg          241    691    159    528
haha        12247   1721     75      5
jayel           0      0      0    161
kinda        8676   4271    368    128
levengals       0      0      0    163
literacy       12    112    535      7
lol         17475   2195    542    873
melissa       371    356     24    276
nan           108    115     10    159
nat           264     97      2    225
postcount       0    356    389      0
ppl          5155    791     54      0
rick          164    513    229    945
school      21845  13784   3026    667
shep            3      0      2    310
sherry         17     54     16    195
spanners        0      1    104      0
teri           14     14      3    108
u           24316   5753    329     61
ur           4854    873     21      1
urllink     29240  93701  24976   4465
wanna        8377   2986    268     54
work        13242  40902   8489   1807
```

2.2. About Unknown data

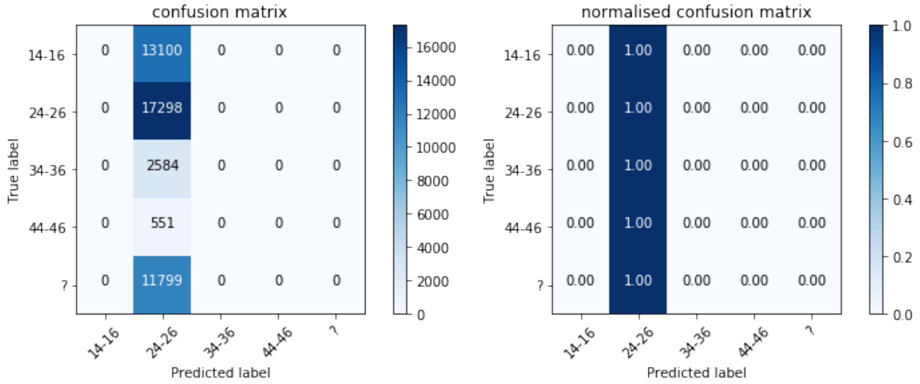
There are 11799 of ‘?’ data in the dataset that are currently not labelled. One could either consider this as a label for all other classes or could treat this as unlabelled data. The former tends to overfit the model and introduces too much noise into the dataset. ‘?’ contains too many potential unknown classifications like 17-20, 27-30 and etc. Making the classifier to classify data with such broad distribution would be too difficult and is likely to overfit the data. The later, however, will always misclassify data with ‘?'. There are 11799 ‘?' data, mislabel every instance in this class will put a hard ceiling of 0.74 on the accuracy of any classifier. The author chooses the second method, as a systematic error is consistent, but a random error is unpredictable. The risk of overfitting makes it difficult to improve classifier performance.

3. Basic classifiers

3.1. Zero-R

Using a basic Zero-R dummy classifier based on the most frequent label yield a result of 0.38. The result of using a common classifier like decision tree is about 0.43, which makes the result of a Zero-R classifier seems relatively high. This will be discussed later. As for now, the result of a Zero-R could serve as our baseline.

Here are the confusion matrices of Zero-R:



3.2. Decision Tree

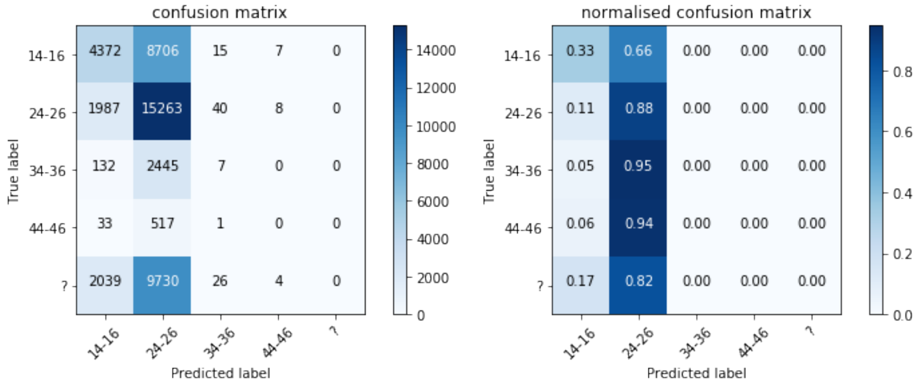
A decision tree stores data in its tree structure and makes predictions by going down level by level until it reaches a leaf node. It uses the 'ID3' algorithm to split its nodes. The value for splitting is based on 'entropy', which is a measurement of the chaos of a splitting. This classifier aims to reduce the overall entropy of the entire dataset by splitting into their own categorisations. The author used a 'DecisionTreeClassifier' from the sklearn.tree package[1].

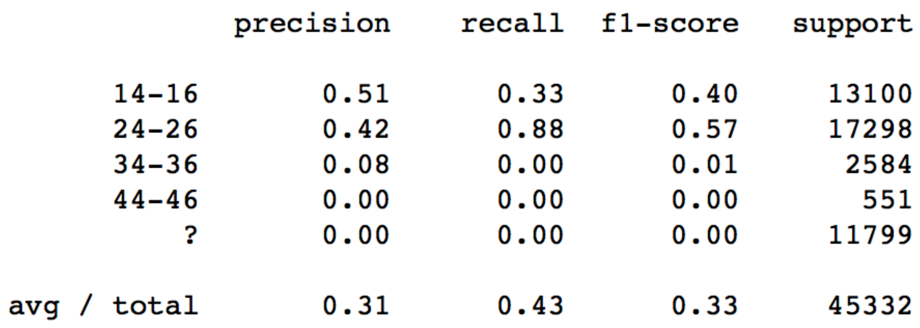
A decision tree is good against the imbalanced distribution of classes (Deshpande, B. (2012, November 16)), which is very common in this dataset. There are too many instances falls within the twenties group while too few for the forties. Since the access to more data is limited, resampling the data may produce some under-sampling of the less frequent data. Therefore, a decision tree is a decent choice to address this issue.

The hyperparameter for splitting is changed to 'entropy' rather than the default 'Gini impurity'. This should not make much difference, as they both behave similarly. For the sake of this subject, I opted for 'entropy' for its accuracy and my familiarity with its underlying principle with some sacrifices on its speed (Wang, H. (2014, August 29)). The max depth of its branching behaviour is not limited, so it is allowed to branch infinite times, which may help but may also produce too many leaf nodes. After training with the training data, DecisionTreeClassifier scores 0.433 on the development data, which is an improvement to the previous baseline classifier.

Yet, a five percent increase in accuracy is not impressive enough. A detail error analysis is shown below:

[1] http://ogrisel.github.io/scikit-learn.org/sklearn-tutorial/support.html





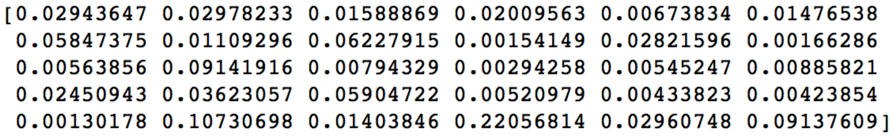
```
             precision    recall  f1-score   support

      14-16       0.51      0.33      0.40     13100
      24-26       0.42      0.88      0.57     17298
      34-36       0.08      0.00      0.01      2584
      44-46       0.00      0.00      0.00       551
          ?       0.00      0.00      0.00     11799

avg / total       0.31      0.43      0.33     45332
```

This confusion matrix explains the DecisionTreeClassifier’s low accuracy. This classifier is able to predict 88% of the 24-26 age group right. However, this classifier is always predicting labels to be ‘24-26' in other age groups. It has difficulty differentiating classes of tens, the thirties and forties from the twenties.

The feature importance data is shown below. Each feature has its own importance, so there is no unused feature for splitting.



```
[0.02943647 0.02978233 0.01588869 0.02009563 0.00673834 0.01476538
 0.05847375 0.01109296 0.06227915 0.00154149 0.02821596 0.00166286
 0.00563856 0.09141916 0.00794329 0.00294258 0.00545247 0.00885821
 0.02450943 0.03623057 0.05904722 0.00520979 0.00433823 0.00423854
 0.00130178 0.10730698 0.01403846 0.22056814 0.02960748 0.09137609]
```

One can see the confusion matrices of these two classifiers (DecisionTreeClassifier & Zero-R) are very similar. Theoretically, a DecisionTreeClassifer recursively performs splitting to form distinct classes. The abovementioned decision tree’s tendency to predict things into one single category means that the feature of this model is not informative enough to reflect the real word classification.

Also, there are many entries that are not valid. For examples, there are empty entries and entries writing in other languages. The pre-processor will not find any information and will simply put zero for every feature. A decision tree has no information about this type of entries and will predict the majority class. Therefore, they might cause the prediction to be biased towards majority.

4. **Feature analysis**

The author believes that the naïve feature selection method by default, which is to tokenise sentences into words and then count the words by its frequency, has its fatal limitations. For example:

The meaning of words differs in different contexts, for example, the word ‘consummate’ has different meanings when used as a verb or an adjective. The naïve approach would categories both cases above as one and would make this feature inconsistent; The word ‘visualise’ and ‘visualize’ are identical in meaning but differs by spelling. The naïve approach will separate these two words, which will lower a feature’s ranking. Therefore, words should be ‘tagged’ with part-of-speech information and then ‘lemmatised’ (to convert similar words into a unique representation) before frequency-based selection.

There are many words in the English language that have no particular meaning, but people use them frequently in their speech. These words like ‘emm’ and ‘uh’ are ‘stopping words’ and appear with very high frequencies. They shall be removed before feature selection because they do not contribute the contextual meaning and content of a sentence. Therefore, they should not be treated as features in order to reduce noise in a text.

5. **Building a new classifier**

The new classifier will be built based on the aforementioned suggestions.

5.1. First, the text section of raw training data is used. For each text entry, tokenisation is applied with wordpuct_tokenize() from NLPT[2] package, this method is better than conventional stripping using delimiter as it can recognise words like ‘Mr.' and separate words like "I'm" into ‘I' " 'm " and punctuations. After tokenisation, a list of ‘tokens' is created. Every token will be striped of characters and converted to lower cases.

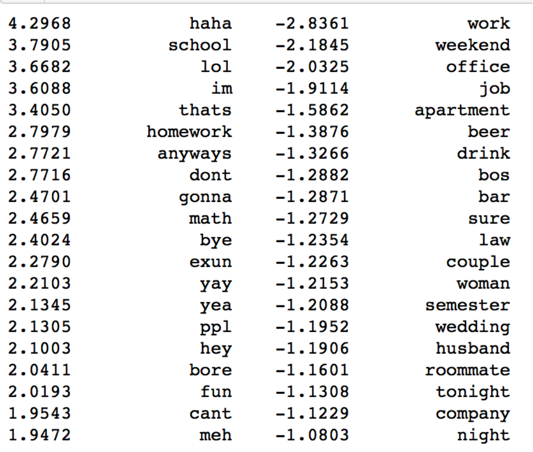
5.2. A tagging of part-of-speech is then applied to every word in this entry. After that, stopping words and punctuations are removed and the rest of the words will be lemmatised using WordNetLemmatizer() from Wordnet and NLTK. Similar words will be converted to a single representation, based on their part-of-speech tags. The result is a list of processed words that contains content-based information.

5.3. Vectorisation will be applied afterwards, which is to convert text to frequency-inverse document frequency (TF-IDF) features. This feature is better than the naïve approach only considered word frequencies, but it fails to address the issue that words appears more often in longer texts. Any sklearn classifier can then be trained from this point onwards using these newly created features.

[2] https://www.nltk.org

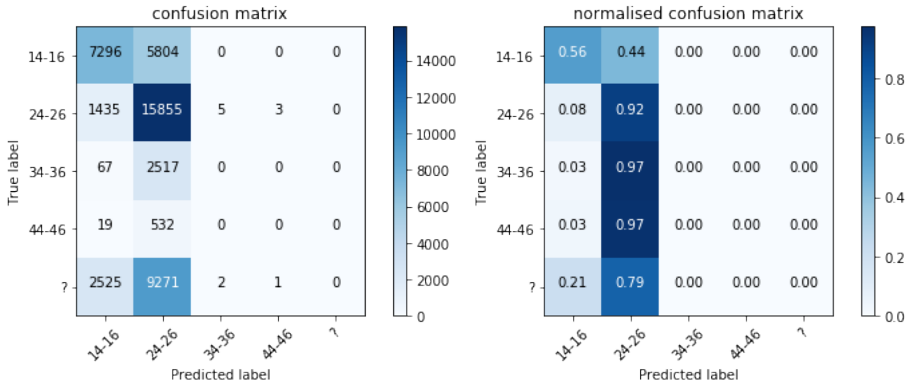
5.4. Using an SGDClassifier() from sklearn, this new classifier is able to achieve 0.51 accuracy score, an improvement to the previous classifier. It is identical to a linear support vector machine with loss = 'hinge' and penalty = 'l2'.

On the right are the new features selected by this model, which makes better sense: As age goes up, people start to care about fun, weekend and beer rather than school and homework. People then start to care about wedding, husband and company in the later stage of their lives.



```
4.2968           haha    -2.8361        work
3.7905         school    -2.1845     weekend
3.6682            lol    -2.0325      office
3.6088             im    -1.9114         job
3.4050          thats    -1.5862   apartment
2.7979       homework    -1.3876        beer
2.7721        anyways    -1.3266       drink
2.7716           dont    -1.2882         bos
2.4701          gonna    -1.2871         bar
2.4659           math    -1.2729        sure
2.4024            bye    -1.2354         law
2.2790           exun    -1.2263      couple
2.2103            yay    -1.2153       woman
2.1345            yea    -1.2088    semester
2.1305            ppl    -1.1952     wedding
2.1003            hey    -1.1906     husband
2.0411           bore    -1.1601    roommate
2.0193            fun    -1.1308     tonight
1.9543           cant    -1.1229     company
1.9472            meh    -1.0803       night
```

This classifier is better at distinguishing '14-16' from '24-26' than the previous classifier, but it still cannot work on other classes. The bias towards majority class has similar reason as decision tree, as lemmatisation only works on English entries. A classifier has no information to predict a foreign language entry, thus makes it subjective to bias towards majority.



6. Future improvements

Other optimisations like grid search could be made to improve the performance of the classifier. Features like sentence structure could be assessed to provide more information to the classifier as well.

7. Conclusion

The author has demonstrated that blog age-group identification could be done with machine learning. The accuracy of any classifier is very dependent on feature selection and good machine learning algorithms. Being able to collect more data is always crucial to improving the accuracy of classifiers.

8. Bibliography

Deshpande, B. (2012, November 16). Decision tree accuracy: Effect of unbalanced data.

Retrieved May 10, 2018, from http://www.simafore.com/blog/bid/111124/Decision-tree-accuracy-effect-of-unbalanced-data

Schler, Jonathan, Moshe Koppel, Shlomo Argamon, and James Pennebaker (2006) Effects of Age and Gender on Blogging. In Proceedings of 2006 AAAI Spring Symposium on Computational Approaches for Analyzing Weblogs. Stanford, USA.

Wang, H. (2014, August 29). [ML] Decision Tree rule selection: Information Gain v.s. Gini Impurity. Retrieved May 9, 2018, from http://haohanw.blogspot.com.au/2014/08/ml-decision-tree-rule-selection.html