Gender Classification using Twitter Feeds

J.V.P.S Avinash and Rakshith Muniraju and Shreyas Shaligraman
CS – 522 – Advanced Data Mining – Final Project
Illinois Institute of Technology, Chicago

ABSTRACT

Accurate prediction of demographic attributes from social media and other informal online content is valuable for marketing, personalization, and legal investigation. This report addresses the task of user's gender classification in social media, with an application to Twitter. Twitter does not collect users' self-reported gender as do other social media sites (e.g., Facebook and Google+), but such information could be useful for targeting a specific audience for advertising, for personalizing content, and for legal investigation. We describe the construction of dataset labeled with gender and investigates machine learning approach for determining the gender of Twitter users. We test the accuracy of our approach by varying various tokenizer options and find the best fit. It is interesting to note that difference in writing patterns is known to exist between the male and female genders.

Keywords: Twitter, Gender Classification, Census, Feature Selection, Tokenization, Accuracy, Confusion Matrix

I. Introduction

Online Social Networks like Twitter play a significant role in the daily life of many peoples, companies and organizations. It is structured to accommodate personal communication across large networks of friends. These social networks produce an enormous amounts of user generated data. This data is openly available and is useful for us to research and analyze the characteristics of user's feeds and profile. Twitter feeds, such as user's tweets and user's profile description has become the subject of many studies like determining age, gender and geographical location.

Unlike traditional authorship analysis problems which are based on samples hundreds of words in length, the analysis of Twitter is hindered by the 140 character limit on tweets. In our current project, we predicted the gender through user's text or description by varying various options like punctuations, URLs, mentions, stop words, lower case, prefix etc.

The remainder of this report is organized as follows. In Section 2, we detail our approach. In Section 3, we describe our twitter corpus, data extraction from twitter and labelling data. In Section 4, we perform data tokenization by considering various options and build a vocabulary. In Section 5, we fit a logistic regression classifier to predict gender from the feed. Finally in Section 6, we summarize our analysis and future work possible.

II. Our Approach

To start with, we collect the data from Twitter API using Python. The properties to establish a twitter connection are stored in a configuration file.

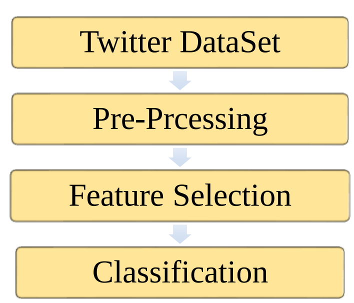

Figure 1 Methodology

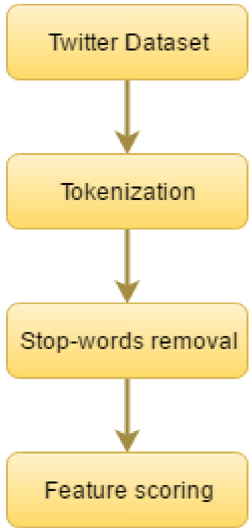

Figure 2 Pre-Processing

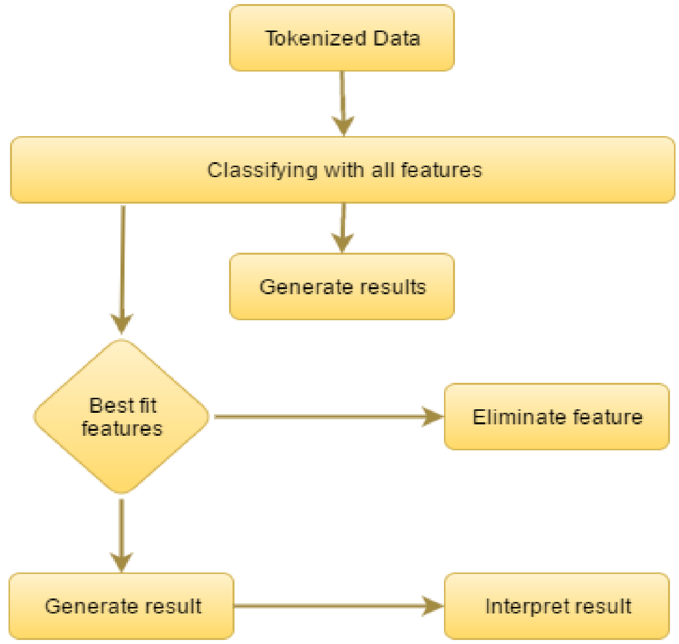

Figure 3 Classification Execution

Figure 1 shows the method we followed in our approach. After collecting the data from twitter, we label the data. We then tokenize the data based on different combinations like lower case, punctuations, prefixes, URLs, mentions, stop words. We tokenize either user's tweet or user's profile description by considering above factors.

Figure 2 and 3 shows how we implemented our pre-processing task and classifier execution. After tokenizing the data, we classify based on all features. Based on the generated results, we evaluate the best feature fit. After choosing the best feature, we eliminate the remaining features. We then apply our analysis on the generated result and then interpret our results.

We observed how tokenization affects accuracy. We came to see that taking mentions and URLs along with collapsing all the stop words, helped us in achieving a best fit for the classifier. We then applied a random junk user's feeds to the classifier and predict whether he is a male or female.

III. Data Collection &Labelling

In this process, we collected "labeled" training data using Census name list. We fetch a list of common male/female sur-names from the census list of year 1990. We implemented a small web crawler to pull the data from U.S census web site.

Next, we sampled data from Twitter using TwitterAPI. It provides programmatic access to public Twitter data. Using Streaming API, we open a continuous connection to Twitter to receive real-time data. We collected 10K tweets of the users whose first name match the sur-names in Census list. To filter the tweets to U.S, we utilized the geographical location attribute provided in the twitter request object.

The "tweet" object returned by the API is a dictionary containing of user related information. For example, tweet['user']['name'] will give us the username and tweet['user']['description'] will give us the user's profile description. We parse the username to get the sur-name.

Next, we created a list of gender labels. We get the sur-name of tweeter and compare it with sur-names in the census list. If the tweeter is found female, we label data as 1 and 0 otherwise. Table 1 summarizes all the analysis.

Observations: -

Census List	
Gender	**Total Count**
Male	1146
Female	4014
Total = 5000	

Number of Tweets by	
Gender	**Total Count**
Male	5182
Female	4818
Total = 10000	

Labelling Data	
Gender	**Total Count**
Male (0)	5182
Female (1)	4818
Total = 10000	

Top 10 Common Names in all tweets	
Name	**Count**
John	112
David	111
Michael	110
Chris	108
Ryan	71
Mike	68
Alex	66
Matt	65
Emily	64
Taylor	62

Table 1 Data Analysis

IV. Tokenize tweets

We perform tokenization that separates words based on various options. These options include a set of (description, lowercase, punctuation, description prefix, URLs, mentions, text tweet, stop words). We take all the possible combinations (True or False) of the above. For example, consider the following tweet

"Keep putting in work #swac #hbcu https://t.co/pdGaGY74ac"

After tokenizing the above tweet, we get *[u'Keep', u'putting', u'in', u'work', u'swac', u'hbcu', u'https', u't', u'co', u'pdGaGY74ac']*.

Let us pick a combination to further process the tweet [*lowercase = True, keep_punctuation = True, collapse_urls = True, collapse_mentions = True, collapse_stop_words = True*]. For the above combination, we get

[u'work', u'keep', u'#hbcu', u'putting', u'#swac', u'THIS_IS_A_URL'].

We tried all the possible combinations for a given feed. If a feed contains URL or Mention, we highlight that with 'THIS_IS_A_URL' or 'THIS_IS_A_MENTION' tags.

Building a Vocabulary: -

Based on the list of tokens generated for all the tweets (similar to the example shown above), we create a vocabulary. Vocabulary is a dictionary from term to index. For each token in a set of tokens for each feed, we make an entry in vocabulary. Then, we find the number of unique terms in the vocabulary for all the possible options. The following are some of the results: -

TOTAL NUMBER OF UNIQUE TERMS IN VOCABULARY							
Use Description and Text	**Lower Case**	**Keep Punctuation**	**Include Description Prefix**	**Collapse URLs**	**Collapse Mentions**	**Collapse Stop Words**	**Count**
True	True	False	True	True	True	True	**28340**
True	True	True	True	True	True	True	**43019**
True	True	True	True	False	True	True	**47062**
True	True	True	True	False	False	True	**53557**
True	True	True	True	False	False	False	**53816**

Table 2 Unique Words in Vocabulary

Table 2 addresses one question "How big is the vocabulary for each combination". We have shown an example for 5 possible combinations. We can see that vocabulary contains 28340 terms if we remove punctuations and the count increases if we include punctuations.

Feature Matrix: -

We build a Compressed Sparse Row Feature Matrix (X) to map each tweet to the frequency of each of the token appearing in it. *X[i,j]* is the frequency of term *j* in tweet *i*. For each token *j* in tweet *i*, we increment the frequency of its occurrence in the matrix. For a vocabulary of 53816 terms and 10,000 tweets, the shape of matrix X will be (10000,53816).

V. Build a Classifier

This section deals with building a logistic regression classifier to predict gender. We trained a logistic regression classifier with L2 regularization. In this model, the probabilities determining the possible outcomes of a single trail are modeled using a logistic function. To fit a model, we need the training vector and a target vector relating to the data. To obtain train and test data sets, we use K-Fold Cross Validation. For our experiment, we fix K to 5. It provides the train or test indices to split the data in train and test sets. It splits the data into K consecutive folds. Each fold is then used as a validation set once while the K-1 remaining folds form the training set.

After generating the training and test index, we fit a model based on X[train_index] and y[train_index] where X is the feature matrix and y is the array of gender labels. After fitting the model, we predict the labels of X[test_index] data and compare it with the y[test_index], resulting in accuracy. Then, we compute the average cross-validation accuracy. The following are some of the reasoning's:

How does tokenization affect accuracy?

To analyze the effect of tokenization on accuracy, we run the above experiment on all the combinations present. Table 3 outlines the result for some of the combinations.

What is the best possible combination?

After all the combinations, we achieved a highest accuracy of 0.7181 (71.8%) for the combination [description = True, Lower case = False, Punctuations = False, Prefix = True, Collapse URLs = False, Collapse Mentions = False, text tweet = False]. As we can observe that removing punctuations and removing mentions in the user's profile description helps to achieve better accuracy.

AVERAGE CROSS VALIDATION ACCURACY								
Use Description	**Lower Case**	**Keep Punctuation**	**Include Description Prefix**	**Collapse URLs**	**Collapse Mentions**	**Collapse Stop Words**	**Use Text**	**Accuracy**
True	False	False	True	False	True	False	False	**0.7181**
True	True	True	True	False	False	False	True	**0.7133**
True	True	True	True	True	False	True	False	**0.7093**
True	True	False	False	True	True	True	True	**0.6999**
False	True	False	True	True	False	False	True	**0.6038**
False	True	True	True	True	False	True	False	**0.5999**
False	True	False	True	False	True	True	True	**0.5797**

Table 3Tokenization vs Accuracy

What is the confusion matrix for our best combination?

A confusion matrix is an error matrix where each column of the matrix represents the instances in a predicted class while each row represents the instances in an actual class. After predicting the labels of the test data, we find the confusion matrix of the predicted labels and actual labels for each fold.

ACTUAL		**PREDICTED**	
		Male	**Female**
	Male	669	364
	Female	180	787

Table 4 Confusion Matrix

Table 4 shows that out of 2000 labelled test data, classifier predicted 1456 correctly resulting in 70% accuracy.

Which decisions had the biggest effect?

Until now, we have calculated the accuracy of classification based on all the possible combinations. Now, we find the maximum score with option = True and option= False for each option. Table 5 shows the result and Figure 3 plots the graph. We can observe that including or deluding the options does not have an impact on accuracy. But is ideal to think that we need to use a combination of all the options in determining the accuracy. But it is good to note that the accuracy while considering any option is around 72%.

Option	**Is included?**	
	Yes	**No**
Use Description	0.7181	0.6061
Lower Case	0.7151	0.7181
Keep Punctuations	0.717	0.7181
Description Prefix	0.7181	0.7181
Collapse URLs	0.7165	0.7181
Collapse Mentions	0.7181	0.717
Use Text	0.717	0.7181
Collapse Stop Words	0.717	0.7181

Table 5 Maximum Score per each option

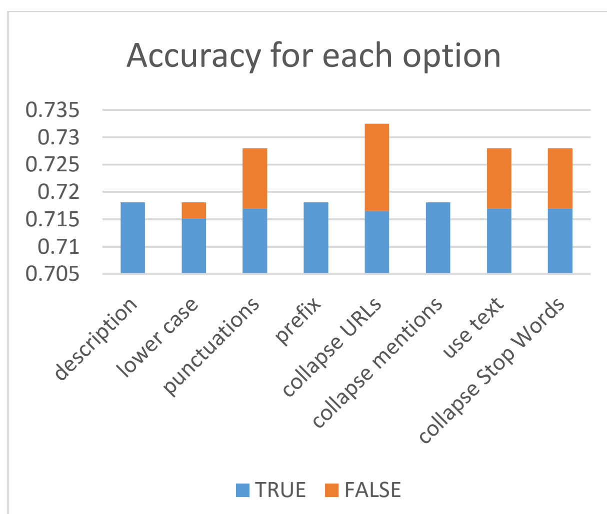

Figure 3 Accuracy for each option

VI. Error Analysis

In this section we try to analyze what terms does our model weigh highly and which examples do we get wrong. In the next part, we try our model on a junk user and then analyze the result.

Which terms does the model weigh highly?

The attribute coef_ in classifier object gives us the coefficient of the features. These coefficients are the weight of each term.

TOP TERMS AND THEIR COEFFICIENTS			
Female	**Weight**	**Male**	**Weight**
d=girl	1.83299	d=pcc	-2.134
d=mom	1.68653	d=devil	-1.909
d=mother,	1.67941	d=guy	-1.883
d=mom,	1.53724	d=father	-1.663
d=✨	1.49101	@thirsty kirstie_	-1.500
d=softball	1.40983	d=husband,	-1.426
d=♡	1.31117	@miranda_ Epley	-1.278
d=❤□	1.27829	d=father,	-1.268
d=princess	1.2105	?	-1.250
d=wild	1.19295	d=husband	-1.190
d=under	1.17859	#aldubbon voyage	-1.130
d=sweet	1.13339	d=sports	-1.100
d=queen	1.11177	d=isn't	-1.074
d=kanjiklub	1.1045	d=some times	-1.049
d=someone	1.04766	boy	-1.049
d=insta:	1.0377	d=guy	-1.036
d=should	1.02199	any	-1.034
d=myself	1.01011	tx	-1.022
d=feckles.	1.00657	tip	-1.021
d=#faithful military women	1.00657	d=former	-0.996

Table 6 Top scored terms

Table 6 illustrates the top weighted terms in both the genders. It is obvious that the female related terms like (girl, mom, mother, princess, sweet, queen, women) are given a high score and are weighted high. Similarly, male related terms like (guy, father, husband, father, sports, and boy) are given a high score and are weighted high. It is interesting to note that smileys represent a higher female coefficient and mentions represent a higher male coefficients.

Which examples did we get wrong?

Let us analyze a tweet which we predicted wrong. The attribute predict_proba in classifier object gives us an estimated probability.

User's Screen Name - kelly nancekivell
Tweet Text :- “grown men fighting to get a foul ball that's still on the field. y'all are the worst kind of baseball fans. #mlb #baseball #toronto”
Tweet User's Description: - “sports. wine. pizza ✌□”

Table 7 Example Tweet

Predicted Gender	*Predicted Probability*	*True Gender*
0 (Male)	*0.962532*	*1 (Female)*

Table 8 Prediction by the classifier

TOP TERMS IN THE TWEET			
Term	**Weight**	**Term**	**Weight**
d=sports.	-0.7607	foul	-0.058
ball	-0.7400	to	-0.054
men	-0.3487	on	-0.052
get	-0.2663	fans.	-0.048
that's	-0.2296	the	-0.028
worst	-0.2188	grown	-0.023
kind	-0.1938	are	-0.012
of	-0.1916	y'all	0.0073
#baseball	-0.1638	baseball	0.0420
#mlb	-0.146	#toronto	0.0601
still	-0.106	fighting	0.0841
a	-0.089	d=✌□	0.0958
field	-0.086	d=pizza	0.1349

Table 9 Weighted score of each term

Table 7, 8 and 9 illustrates how we perform the analysis on the tweet. We can see that tweeter, 'kelly nancekivell' is female but our classifier predicted male with 96% probability. The reason why our classifier predicted incorrectly is that the tweet and user's description contains words which weigh more to male gender. Some of the words like (men, sports, ball, #baseball, foul, field, fans) relate more towards male gender. Hence our classifier predicted male.

Can we predict the gender of some junk user?

In this work around, in addition to our normal 10,000 tweets, we pull an extra of 200 tweets whose screen names does not match with anyone in the census list. We then apply our classifier to predict the gender of that junk user. At last, we get the true gender by manually checking name in twitter.

User's Screen Name - tmj_sea_nursing		
Tweet Text :- "TMJ-SEA Nursing Jobs"		
Tweet User's Description: - "Follow this account for geo-targeted Healthcare-Nursing job tweets in Seattle, WA. Need help? Tweet us at @CareerArc!"		
Predicted Gender	*Predicted Coefficient*	*True Gender*
0 (Male)	*-2.46903*	*NA*

Example 1 Case where the classifier fails

User's Screen Name - JerunkGirl		
Tweet Text :- "Sexy&Sober"		
Tweet User's Description: - "1st day of miracles began XI▪XII▪MMXIII. Call me anything you want but please add 1of the following: sexy, gorgeous, intelligent, sober"		
Predicted Gender	*Predicted Coefficient*	*True Gender*
0 (Male)	*-2.46903*	*1 (Female)*

Example 2 Case where the classifier works

Example 1 and 2 illustrates the cases when the classifier fails to predict and succeed in predicting. It can be observed from example 1 that, "TMJ SEA Nursing" is a job portal aimed to provide employment in the field of health care. Since it is a group/ department, gender for it is invalid. But, our classifier predicts the gender as Male. This can be that the terms in both the text and description like (Jobs, Follow, Need and Help) made the classifier weigh them more towards male category and hence predicting gender as Male. From example 2, we can observe that, "JerunkGirl" implies a female name and hence our classifier was accurate predicting the gender as Female. Also, the terms in text and description like (sexy, gorgeous, intelligent and sober) weigh more towards female and hence the classifier predicted correctly.

The difficulty in this task is that we need to manually check the gender of user by scrolling over their twitter page. This can be either be by verifying the photo or some of re-tweets. For our sample junk data of 200 tweets we attained 80% accuracy, which is fair.

VII. Conclusion and Future Work

The rapid growth of social networks, particularly Twitter, has produced an unprecedented amount of user generated text which may be used for authorship analysis, including gender prediction. Because of the anonymity on the Internet, many times the text is the only data source for gender identification. In our current project, we presented a novel approach to predict gender utilizing the tokens generated from user's feeds. Firstly, we fit a classifier model using the training data and test labels. We then calculated the accuracy of our classifier by applying the test data to the model and predicting the labels. From our analysis, we attained 72% accuracy. Then, we perform error analysis to understand why our classifier failed to predict correctly. We have also performed human verification on the labels predicted by classifier on random junk user. We attained around 80% accuracy in this case. I think we can think of ways to combine labelled and unlabeled using semi-supervised learning approaches and can improve in accuracy.

In future work, we will increase the data, so that we can get more unique tokens from the tweets. To accommodate the large data, we built a HDFS layer. Integration of Python with Hadoop can be done using 'mrjob' package. Additionally, we can explore how well our model carry over to gender identification in other informal online genres such as chat and forum comments. Furthermore, we can also predict other demographical features beside gender, like age, occupation, marital status and religion etc.

Appendix

All the coding for the project was done in Python. I have included all the 'py' files and 'ipynb' files. In addition to that, I have also included some intermediate output files like: - *options.txt* (contains the possible combinations of all options and related tweets outcome), *results.txt* (contains the accuracy for each possible combination), *predicting.txt* (contains the gender predicted for junk user's tweet). The following are the '.ipynb' files: -

- *dataGeneration* – Contains code for pulling data from Twitter
- *tokenizeTweets* – Contains code to tokenize the tweets based on all options
- *sampleClass14* – Contains functions such that they are accessible by all files
- *labelling_and_regression* – Contains the code to label the data and fit a logistic regression classifier.
- *errorAnalysis* – Contains the code that performs error analysis.

The entire code is split into various files as to improve readability and understand each process.

References

[1] https://www.cs.uic.edu/~buy/Asonam2013.pdf
[2] http://firstmonday.org/ojs/index.php/fm/article/view/5216/4113
[3] http://file.scirp.org/pdf/IJIS20122400002_20423388.pdf
[4] http://cs.iit.edu/~culotta/pubs/culotta15predicting.pdf
[5] http://www.aclweb.org/anthology/D11-1120
[6] A Machine Learning Approach to Twitter User Classification
[7] http://www2.cs.uregina.ca/~hilder/my_students_theses_and_project_reports/ugheokeMScProjectReport.pdf
[8] http://www.cs.jhu.edu/~delip/smuc.pdf
[9] http://hltcoe.jhu.edu/uploads/publications/papers/17310_slides.pdf
[10] http://wwbp.org/papers/emnlp2014_developingLexica.pdf