Gender Prediction from Blog Posts

Sheila Christian

December 18, 2012

Abstract

In this paper, I explore the problem of gender identification from blog posts. The data for this project consists of a corpus of pre-classified posts, along with a few additional known features (e.g., age). This project focuses on comparing the performance of term-specific features using LightSIDE [1], including unigrams, bigrams, and trigrams, and general meta-features generated using OpenNLP [2] and Java code, including the average number of words per sentence, the use of strong language, the use of emoticons, and the use of repeated letters. This work emphasizes gender prediction through examining general characteristics of blog entries, such as average word length, instead of prediction through blog wording, e.g., unigrams.

1 Introduction

The problem of gender classification from online posts has been extensively explored in the field of machine learning [3-6]. The foremost attraction of the task is that the data is easy to acquire, but difficult to process. Depending on the post length, the variation in content, the degree to which authors are representative of their gender, and whether or not certain additional information is also known about the content-provider, the problem of gender classification may be more or less challenging.

The task of gender classification is also interesting for its potential applications. For example, algorithms that can reliably detect the gender of an author of an unknown online post could be used to generate statistics of website membership or activity rate. Another example would be using the predicted demographics information for targeted advertisement.

Furthermore, specific features that are found to be best at discriminating males from females could provide insight into psychological differences between the genders. For example, if younger men are found to write more similarly to older women, then it could be concluded that men write more maturely in general, which might be relevant to other fields.

Finally, by identifying underlying features that separate male and female writers, it is also possible to mimic the male or female voice, which could be relevant to the field of AI, or simply used for entertainment purposes [3].

All of the above uses of gender classification depend on a reliable model that can perform statistically better than random. However, an additional concern besides performance accuracy is what data is available to build a given model.

Many text-mining approaches depend heavily on large data sets of pre-classified instances in order to derive a set of valuable n-gram features. While this approach is often very successful, it can be computationally expensive, as well as difficult to interpret. An alternative approach is to focus on features based on known differences between male and female communication patterns, which can be applied to new datasets without the need for feature extraction from a large set of sample text.

2 Related Work

Researchers have previously approached the problem of gender classification from online material. This prior work addresses a range of source text, including instant messages, e-mails, spoken language, status updates, and blog posts. However, consistent patterns emerge in terms of gender-specific communication characteristics across all of these media [3-7].

The models that have been presented in the relevant literature rely on two types of classification data: *content-specific* features and *stylistic* features. Content-specific features include specific words and phrases, which translate to non-stylistic unigrams, bigrams, and trigrams, in the context of this project. For example, a binary feature might be created for the presence of the word "shoes" in the blog text. Thus, this feature correlates with whether or not the author addresses shoes in the subject matter of their post.

Previous research has found that men are more likely to focus on the outside world in their subject matter, whereas females focus on more inner-directed communication [4]. More specifically, men are more likely to write about impersonal subjects, such as locations and journeys, whereas women are more likely to write about relationships and domestic topics, such as shopping and cost [5].

Stylistic features, on the other hand, are not directly associated with *what* is said in the text, but instead place emphasis on *how* it is said. They may include the use of punctuation, strong language, personal pronoun, and also text length. A variety of conclusions exist in the literature regarding the stylistic writing differences between men and women. Among these conclusions are that men and women differ in terms of use of emotion words, assertiveness, reactive words, and humor [6].

In addition to the in-depth investigation of differences between content and stylistic choices of male and female authors, the literature presents additional feature-generation techniques relevant to online NLP. For example, “stretchy patterns”, which is a meta-feature generation technique that involves the addition of "gaps" of zero or more words between relevant n-grams, has been found to perform significantly better than baseline approaches [7].

3 Data Set

The data set for this project consists of 2000 instances of pre-classified blog posts, with 1000 posts by females, and 1000 by males. In addition to the gender class value, the dataset also includes the author's age, occupation, and astrological

sign. The astrological sign was immediately identified as a non-valuable feature, and was thrown out on the basis that it had no correlation with either the gender of the author, or the text they produced.

The average number of words per post was 337, with a standard deviation of 351. The average number of words, and standard deviation of number of words are shown in the table below. The blog post text was used to generate the features described in the following sections.

	Female	Male
Average Words	357.99	316.49
STD Words	359.91	340.68

The fact that the provided corpus was pre-classified allowed for supervised learning. Also, the blogs in this corpus were not adjusted to all focus on the same topic. As a result, the models generated in this project benefitted from the fact that the subject matter varied from one author to the next.

The data for this project was divided into four sets. The *development set*, with 200 instances (10%), was used to closely examine the data, including reading individual blog posts to identify key features and to perform error analysis. The *test set*, with 400 instances (20%), was used to test various models as they were being considered.

The *train* set, with 1000 instances (50%), was used to train various models. The *holdout set*, with 400 instances (20%), was used to test the final result once an optimal model had been determined. Instances were assigned to the sets randomly, which resulted in each of the four sets having nearly 50% male and 50% female instances.

4 Baseline Performance

The baseline performance for this project was a percent correct value of 0.5, and a kappa of 0. This equates to guessing equally between male and female for each prediction, as the dataset contains an equal number of male and female instances.

In this case, the baseline has the same performance as using zeroR. It may be tempting to assume that the baseline performance in all cases of gender identification should be equivalent to a percent correct value of 0.5. However, the proportion of male to female authorship is generally unknown for a new dataset. Furthermore, it would be unlikely that a randomly sampled datasets would have the same number of male and female posters. In fact, there is evidence that women are more likely to post social updates about themselves then men [3].

In addition to the baseline performance with a percent correct value of 0.5, I also compared my results to the performance of a model built from the data using feature extraction in LightSIDE. The performance of the best model that could be achieved in LightSIDE was considered the gold standard for my custom features, as LightSIDE was already known to perform well for machine learning using text.

5 Procedure

In this section, I discuss data exploration, feature development, and model selection.

5.1 Data Exploration

The first stage of the procedure was to

become familiar with the data by analyzing it directly. A subset of ten blog posts by male authors and ten blog posts by female authors was selected from the development dataset for careful analysis. Each post was read and used to generate preliminary feature ideas for distinguishing male writing from female writing.

Men seemed to use more academic words. For example, posts by males of all ages included terms such as "customized," "fortification," "monotonous," "fabrication," and "guttural." By contrast, women seemed more likely to use more common words. This observation suggests that an interesting feature might be to check for the average length of words, and/or their "common-ness" relative to a dictionary of the most common English words.

Another observation was that women seemed more likely to use repeating letters or symbols. For example, they used, "lollll", “xxxxx”, “byeeeeeeeeee”, “heyyyy”, and “booooored." The use of repeating letters within words as a means of emphasis matched my expectations of the typical female online writing style.

However, repeated letters within words might be a blog-specific feature (as opposed to more formal e-mail messages or shorter status updates), since blogs are unique in that authors often write using "stream of consciousness," and may not bother to edit their posts before submitting them.

Finally, the subject matter of the blog posts tended to vary. I had expected that females would be much more likely to write about relationships, whereas males would be more likely to focus on sports, movies, music, or technical topics. Surprisingly, many of the posts by both males and females between the ages of 14 and 20 focused equally on personal emotions and relationships with the opposite gender. However, girls were much more likely to mention a "boyfriend," "hot guy," "cute" clothes, and references to things that were “fun”. On the other hand, guys were more likely to write about typical "male" content, such as drinking, sports, and using the bathroom.

A few other observations from the data exploration stage were also noted. Females were more likely to write "gosh", or "oh my god." Males seemed somewhat more likely to "welcome" people to their blog, for example, by writing "Welcome people," "Hello all," or "Hey guys." Males also seemed to use both an impersonal and a sarcastic tone more heavily than females. However, analyzing the rest of the development data revealed that these features were either difficult to detect, or did not appear sufficiently often for them to contribute significantly to the success of the model, so I did not attempt to generate features based on them.

5.2 Feature Development

The second stage of the procedure was to generate custom features for the training dataset. These features were based mainly on the results of the data exploration stage, but were informed by prior knowledge of blog posts (e.g., almost all blog posts were likely to use personal pronoun).

In order to create custom features with the greatest degree of flexibility, I wrote custom Java code that would read in the set of instances, including existing features, blog text, and class value (in this case, gender), calculate new features based on that information, and output an updated file with the new features.

I used OpenNLP to split blog posts into sentences and tokens, as it provided a stronger framework for parsing text. Although LightSIDE offers the ability to create features based on text by adding customized Java plug-ins, this option did not afford the full range of customization available from an entirely separate Java project.

The generated features were:

- Number of sentences
- Number of tokens
- Number of tokens per sentence
- Average word length
- Rate of words with repeating letters
- Rate of emoticon usage
- Rate of curse word usage

OpenNLP was used to determine both the number of sentences per text, and the number of tokens. A cutoff value of ten characters was used in calculating the average word length, to eliminate the effect of non-word outliers, such as URLs or "words" containing many repeating characters.

Both rate of emoticon usage and rate of curse word usage were determined using custom lists of roughly 15 common curse words and 50 common emoticons. These lists were compiled using a combination of online sources and the development data.

For the features “repeating letters”, “emoticon usage”, and “curse word usage”, I used a normalized rate value instead of either a binary or count-based feature. The rate value was calculated using the number of occurrences divided by the number of tokens in the post, and then normalized (which was approximiated by multiplying by a constant of 1000, for simplicity). A rate value was used because the range of blog post lengths was very large (with a standard deviation of 351 words), so any attempt to use a binary or frequency feature would have been significantly skewed in favor of longer blog posts.

Features were also generated in LightSIDE using the default “TagHelperTools” feature extractor plug-in. Unigrams, bigrams, trigrams, POS bigrams, and punctuation features were all considered. The options to remove stopwords, and to treat all features as binary, were also considered when generating n-gram features.

5.3 Model Selection

The first phase of model selection involved comparing a wide range of different types of classifiers. The goal was to identify which were the most successful when built using a selection of the custom-generated features.

The first features to be tested were: age, number of sentences, number of tokens, and average number of tokens per sentence. However, these features were not found to perform any better than baseline (percent correct 0.5, kappa 0) for any model (see Table 1). This is probably due to the fact that these features are not particularly gender-specific. In fact, both males and females may write posts of any length. All of the results were obtained using default WEKA settings.

Table 1: Text length features

Percent Correct	Kappa	Model
49.96	-0.0035	J48 (tree)
46.87	-0.0652	SMO (SVM)
49.29	-0.0092	Multi-Layer Perception
49.29	-0.0004	Naïve Bayes

In an effort to obtain more significant difference between model types, more features were added. These were: average word length (for words less than ten characters) and the rate of words with repeating letters. These two features were based on observations that men tended to use more advanced vocabulary, while women tended to use words with repeating characters. The results are shown in Table 2. All of these results were obtained using default WEKA settings.

Table 2: Average word length and repeating letters

Percent Correct	Kappa	Model
55.50	0.1194	SMO (SVM)
51.25	0.0123	Naïve Bayes
50.25	0.0100	J48 (Tree)
54.50	0.0920	JRip
57.00	0.1472	AdaBoostM1

In this case, SMO and AdaBoost performed the best. In fact, they performed statistically better than baseline (determined using a two-tailed 90% confidence interval and paired T-test). In fact, SVMs have been shown in previous literature to work very well for text-based classification [6]. It is also not surprising that AdaBoost performed well, given that it achieves higher accuracy by incrementally correcting errors.

Interestingly, both the tree and the rule-based models ignored all features except number of tokens and word length, but performed significantly differently due to choosing different cut-off values (3.748 for word length for JRip, instead of 3.96 for J48).

The next features to be included were the rate of usage of both emoticons and curse words. It was guessed based on the data exploration that emoticon usage would be higher among women, and that curse word usage would be higher among men. Since it was already known that SMO and AdaBoost performed better than the alternative model types, only these two models were used in testing the additional features. The results are shown in Table 3.

Table 3: Emoticons and curse words

Percent Correct	Kappa	Model
54.25	0.0942	SMO (SVM)
56.25	0.1325	AdaBoostM1

To gain a meaningful sense of the performance of the custom features, I also needed to test the optimal performance that could be achieved using the LightSIDE feature extractor.

I started by extracting all of the possible features that were given as default options in LightSIDE. These included: Unigrams, Bigrams, Trigrams, POS Bigrams and Punctuation. I also selected the option to remove stopwords, and to use binary features. After trying a variety of other combinations of settings (Table 4), I determined that the original settings I had selected achieved the optimal performance. The only model that I tested was SMO, since I had previously determined that it performed well for my custom features, and because my review of the relevant literature had pointed to SMO as a particularly successful model for text mining [6].

Table 4: LightSIDE testing

Percent Correct	Kappa	Features
64.000	0.2799	Unigrams, Bigrams, Trigrams, POS Bigrams, Punctuation, Remove Stopwords, Binary Features
55.8559	0.117	Bigrams, Trigrams, POS Bigrams, Punctuation, Remove Stopwords, Binary Features
54.8549	0.0961	Unigrams, Bigrams, POS Bigrams, Non-Binary Features
57.958	0.1673	Unigrams, Bigrams, POS Bigrams, feature selection on top 100 features
58.8589	0.1835	Unigrams, Bigrams, POS Bigrams, Remove Stopwords, feature selection on top 100 features
57.958	0.1649	Bigrams, Trigrams, POS Bigrams, Punctuation, Remove Stopwords, Binary Features, feature selection on top 100 features

The fact that the most successful settings were a combination of all possible options is not surprising, given that LightSIDE was developed to perform text mining for classification problems very similar to this one. In fact, each of the options adds an additional space of information that the model can use to improve its accuracy.

6 Error Analysis

In order to perform error analysis on my features, I used the LightSIDE Metafeatures feature extractor plug-in. I built a model using my set of seven custom features, plus age, and SMO as the classifier. I continued to test on a separate test set of 400 instances, which had been prepared so as to have the same set of features.

I then analyzed the confusion matrices for my overall performance, as well as the matrices for each of my custom features. I found that my model was much more likely to guess that an instance was female than male. Table 5 shows my confusion matrix for overall performance, using SMO built with my seven custom features plus age.

Table 5: Overall performance

Act/Pred	Female	Male
Female	175	22
Male	161	42

This result was surprising given that two of my features were expected to prefer males (word length and curse words), two of my features were expected to prefer females (emoticons and repeating letter), and age, number of sentences, average tokens, and total number of tokens were expected to vary only slightly between the genders. I had therefore expected my model to guess roughly equal numbers of males and females.

My confusion matrix for age suggested that it contributed very little to the model, since it had very low horizontal comparison values as well as very low vertical values (normalized to 0.02 and 0.049, respectively). However, since there did not seem to be a significant degree of confusion, I concluded that age was not detrimental to the performance of my model.

The emoticons feature appeared at first to have a high rate of confusion because it had a very low vertical comparison value, and a relatively high horizontal comparison value. However, upon closer inspection, I realized that the feature had

only been used to classify females. The feature was in fact accurate in about 60% of cases, which was higher than the accuracy of the model as a whole, so I did not consider the feature to be a problem.

The features that I found were causing the most confusion were the repeating letter feature (Table 6), and both the number of tokens feature (Table 7) and the average tokens feature.

Upon closer inspection of the data for the repeating letter feature, I found that one source of confusion could be the use of the repeating '.' character. I had made an exception for the character '.' in my original feature because I had noticed that it was used very often by both males and females. However, after considering the confusion matrix and upon revisiting my development data, I decided not to make an exception for the '.' Character. After making this change, my confusion matrix changed to that shown in Table 6B. Meanwhile, my performance increased from 54.45% to 54.5% correctly classified. Although this change was positive, it was not statistically significant.

Table 6: Repeating letter

Act/Pred	Female	Male
Female	0.279	0.215
Male	0.182	0.214

Table 6B: Repeating letter ('.' Included)

Act/Pred	Female	Male
Female	0.279	0.237
Male	0.181	0.219

I also revisited my development data to understand the confusion of the number of tokens feature. I found that the cause of the confusion for the feature "number of tokens" was likely a few outliers in which a blogs by males had far more than the average number of words. In fact, the feature for average tokens had slightly lower but similar rates of confusion to the feature for number of tokens. I therefore concluded that the feature for average tokens was making mistakes for the same reason.

Table 7: Number of tokens

Act/Pred	Female	Male
Female	321.057	257.591
Male	353.708	249.429

Considering the results of my error analysis, I decided to test dropping either the feature for number of tokens, or the feature for average tokens, or both. When I dropped "number of tokens" as a feature, my performance improved to 56.5% correctly classified, with a kappa statistic of 0.1387.

7 Optimization

In order to optimize my model, I used CVParameterSelection in Weka, while running 10-fold cross validation using SMO on my modified feature set (with the repeating letter feature adjusted, and with the number of tokens feature removed).

When I selected a range of 1 to 5 for my C parameter for SMO, I found an optimal value of 2.333. The performance of my model increased to a correctly classified 57.458%, with a kappa value of 0.1371.

8 Result Comparison

My final performance, using my optimized SMO parameter setting of 2.333,

and tested on the original holdout data set of 400 instances, achieved a percent correct value of 54.75%, and a kappa statistic of 0.1061.

In order to determine the success of my custom features, I had to compare the performance of my final model with both my baseline performance (0.5 percent correct, and kappa value of 0 for zeroR) and my gold standard performance (roughly 0.64 percent correct, with a kappa value of 0.2799 for LightSIDE with all default options selected).

My final results performed better than the performance of zeroR, however, they still performed worse than the model built using feature extraction in LightSIDE.

9 Conclusions

The purpose of this project was to identify whether a small subset of features based on text structure and expected differences between male and female writing styles could perform comparably to an algorithm that relied entirely on n-gram features derived using specialized text-mining software.

The features used in this project did not achieve performance that was comparable to that which is achieved through feature extraction. However, it is still reasonable to conclude that an equally small yet more powerful set of features could be devised that would achieve comparable performance.

This comparison is valuable because feature extraction from a large corpus of text samples can be computationally expensive and difficult to understand. In contrast, features such as those presented in this paper can be shared more easily, understood at a higher conceptual level, and calculated with lower time and space complexities.

Security and privacy issues could also make the features analyzed in this project preferable to those generated using feature extraction. For example, a relevant situation might arise if an owner of a dataset were willing to provide meta-data about the content of their text samples, such as average word length, number of uses of curse words, instead of the actual text. Future work can build on these findings to provide privacy-aware gender prediction through examining only characteristics of blogs instead of actual blog content.

References

[1] Rosé, C. and Mayfield, E. LightSIDE: Machine Learning for Text. <http://www.cs.cmu.edu/~emayfiel/side.html>

[2] OpenNLP. The Apache Software Foundation. <http://opennlp.apache.org >

[3] Keeshin, J., Galant, Z., Kravitz, D. 2010 Machine Learning and Feature Based Approaches to Gender Classification of Facebook Statuses. <http://thekeesh.com/cs224n/final_writeup.pdf>

[4] Argamon, S., Koppel, M., Pennebaker, J., and Schler, J. Mining the Blogosphere: Age, gender and the varieties of self–expression. *First Monday,* volume 12, number 9 (September 2007), <http://firstmonday.org/issues/issue12_9/argamon/index.html>

[5] Colley, A. and Todd, Z. 2002. Gender Linked Differences in the Style and Content of E-mails to Friends. Journal of Language and Social Psychology. <http://jls.sagepub.com/content/21/4/380>

[6] Corney, M., de Vel, O., Anderson, A., Mohay, G. Gender-Preferential Text Mining of E-mail Discourse. 2002. Computer Security Applications Conference. <http://eprints.qut.edu.au/8014/1/8014.pdf>

[7] Gianfortoni, P. Adamson, D. and Rosé, C. 2011. Modeling of stylistic variation in social media with stretchy patterns. In *EMNLP Workshop on Modeling of Dialects and Language Varieties.*