NLP for Text-Based Classification of Astrological Sign

Rachel Rosenberg
https://github.com/rrosenb1/rrosenb1_msia490_2019/tree/project

Abstract

This document contains information about the background, methods, and results of a Natural Language Processing model built to classify an author's astrological sign based on text from his or her blog post. Data comes from a corpus of over 600,000 blog posts from over 20,000 Blogger.com authors in 2004.

1 Credits

This project is the final submission for Emilia Apostolova's Text Analytics course, MSiA 414, for the Master of Science in Analytics program at Northwestern University. It takes advantage of Natural Language processing techniques from preprocessing to modeling to prediction and presentation, all of which were covered in the class. Data comes from The Blog Authorship Corpus, developed by researchers from Bar-Ilhan University, the Illinois Institute of Technology, and the University of Texas (Schler, Koppel, Argamon, and Pennebaker, 2006). Koppel and Schler are well-known authorities on multi-class text classification.

2 Introduction

The basic method used in this paper is multi-class text classification, a classic and effective NLP and supervised learning technique that seeks to group text by an output variable. This paper will discuss relevant literature, preprocessing methods, feature creation steps, modeling to choose at a high level which kind of model to use, and fine tuning for a final model. The final four-class model has an accuracy of about 40%, with a similar F1 score, indicating some pattern in the data identifying an author's astrological sign; a baseline model would perform at a 25% accuracy on average. The dataset contains labels for the gender, age, category, and astrological (zodiac) sign of the author. It comes with one file (labelled file name) per author, with all of the author's blog posts included in one text file.

3 Literature Review

There are many types of algorithms that can be used in text classification, as in other applications of classification problems. These can include standard methods like Decision Trees and Logistic Regression, ensemble methods like Random Forests and Boosted Trees, and deep learning methods like Convolutional Neural Networks and LSTMs (Khan, Baharudin, Lee, and Khan, 2010). Different algorithms are good for different use cases (Pawar and Gawande, 2012); for example, kNN clustering for grouping of news articles (Tam, Santoso, and Setiono, 2002), sentence importance labelling and Naïve Bayes clustering for a similar set of news articles (Ko, Park, and Seo, 2004), and dictionary-based categorization of chemical web pages (Liang *et. al*, 2006).

The variety of approaches taken in the past speaks to the long history and extreme flexibility of text classification. It is possible to combine, munge, and parse data in a huge number of ways; the way that works the best for a given application will depend heavily on the data's structure and future use cases of the model.

There does not appear to be an existing body of work on any kind of Machine Learning for astrological sign prediction; this is likely due to the nature of astrology being considered a pseudoscience and therefore not easy or

reasonable to classify or predict (Carlson, 1985). While this was expected, it is also interesting to note that there was *some* signal in the noise, as indicated by the better-than-baseline accuracy of the models built using these methods.

4 Methods

Methods used for preprocessing include standard tokenization, lemmatization, and cleaning (removal of numbers, etc.) Modeling methods used include Logistic Regression, Decision Tree Classification, Random Forest Classification, Support Vector Machines, K Nearest Neighbors Classification, and MLPC Classification.

4.1 Preprocessing

To clean for modeling, the data was manipulated to have one row per post (rather than one row per author) and to have the label as a category. Labels are one of Fire, Water, Air, and Earth, the four general types of astrological signs (Higgins, 1997); this decreases the number of classes that must be present in the classification.

Standard NLP preprocessing steps of stripping, converting to lowercase, removing numbers, removing stop words, and lemmatizing were then done to clean the data. Originally this was done by stemming words; however, the final version uses NLTK's WordNetLemmatizer, which is much faster and achieves a human-readable result.

Features were created by using TF-IDF vectorization, which is standard practice in NLP. Much of the experimentation for this project involved manipulating TF-IDF parameters; this will be discussed in more detail in Section 3.3.

The following TF-IDF parameters were manipulated (Pedregosa *et al.*, 2011):

- Min_df – tells vectorizer to ignore words with less than a minimum frequency. For example, min_df = 25 will ignore words that are used fewer than 25 times in the corpus.
- Ngram_range – the range of n-gram values to use (bigrams, trigrams, etc), from 1 to N.
- Normalization method (norm) – the regularization method used on the output. Options are L1 (ridge regression) or L2 (lasso regression).

The following TF-IDF parameters were constant across all experiments (Pedregosa *et al.*, 2011):

- Max_df – tells vectorizer to ignore words with greater than the maximum frequency. For example, max_df = 0.8 will ignore words that occur in >80% of the documents. A max_df of 0.8 was used as a standard.
- Encoding – Latin encoding was used for all documents.
- Binary – Sets the "tf" term to binary (0 or 1) values. This was always set to False, so real word counts are used.
- Stop words – English stop words were removed.

4.2 Model Type Experiments

For the model type experiments phase, a subset of the whole dataset was vectorized, split into training and testing sets, and cleaned for use as a standard training set. Then, each of several types of models were built once (Pedregosa *et al.*, 2011):

- Logistic Regression – a standard, parametric classification method which is simple and widely used.
- Decision Tree Classifier – a slightly more complicated, non-parametric classifier.
- Random Forest Classifier – an ensemble ML method built on decision trees.
- Support Vector Machine – a boundary-based method that is common for text classification.
- K Nearest Neighbors – an unsupervised clustering method that compares each text vector to similar others.
- Multi-Layer Perceptron Classifier (MLPC) – Sklearn's built-in implementation of a neural network for classification with stochastic gradient descent.

Overall, logistic regression performed the best of these six types. Specific results will be discussed in Section 4.

4.3 Model Tuning

Once the best-performing model had been determined, it needed to be tuned to produce a truly optimal model. This came in two phases: choosing

the best TF-IDF parameters, and choosing the best modeling parameters.

To choose the best TF-IDF parameters, the parameters discussed in Section 3.1 were manipulated in gridsearch with the following options:

- Min_df = [25, 50, 75, 100]
- Ngram_range = [(1, 1), (1, 2), (1, 3)]
- Norm = ['L1', 'L2']

The most successful combination was min_df = 25, ngram_range = (1, 3), and norm = 'L2', because this maximized the number of features available and then sharply cut out any that were not useful.

To choose the best modeling parameters, the following parameters were manipulated in gridsearch with these options:

- Max_iterations – the number of iterations that the regression will complete before converging = [100, 200, 500]
- C – strength of regularization = [0.5, 1, 5]
- Multi-class method – method used to solve the multi-class problem. One-vs-rest ('ovr') solves separately for each label, while multinomial minimizes loss across the whole probability distribution. Method = ['ovr', 'multinomial']

The most successful combination was then max_iterations = 100, C = 1, and multi-class method = 'o vr'. This signals that the regression problem was fairly straightforward for the solver, because it converged quickly without needing to significantly tighten regularization constraints.

5 Results

Generally, all Logistic Regression models performed with an accuracy between 0.340 and 0.404. The three best-performing and two worst-performing combinations of TF-IDF parameters for Logistic Regression are found in Appendix A. Logistic Regression was likely a successful model because it is not too complex, and does not attempt to dramatically overfit the training set.

Results for the best experiments tried are seen below:

Model Name	**Accuracy**	**F1-Score**
Logistic Regression	0.371	0.370
Decision Tree Classifier	0.314	0.314
Random Forest Classifier	0.321	0.319
Support Vector Machine	0.368	0.367
K Nearest Neighbors	0.267	0.257
MLPC Classifier	0.265	0.105
Logistic Regression with min_df = 25, ngram range = (1,3), norm = L2	0.404	0.404
Logistic Regression with min_df = 25, ngram range = (1,3), norm = L2, max_iter = 100, C = 1, multi-class method = 'ovr'	0.405	0.405

The best model was Logistic Regression with min_df = 25, ngram range = (1,3), norm = L2, max_iter = 100, C = 1, and multi-class method = 'ovr'. Therefore, this model was used for all prediction and for the API.

6 API

The final step of the project was to serve the model in a REST API (Ngo, 2018). This allows for a user to test a specific section of text from the command line prompt; the model returns which type of astrological sign the text is most likely to come from, and its confidence in predicting that class. Some results are below:

Text: "That movie was boring."
Result: "prediction: Water, and confidence: 0.319"

Text: "That movie was so exciting. I loved it."
Result: "prediction: Water, and confidence: 0.493"

Text: "Our trip to coding camp was very memorable. I really enjoyed our time there."
Result: "prediction: Fire, and confidence: 0.448"

Generally, the model performs better when more text is used in the query (so that it has more features to draw from) and when there are strong sentiments or strong words present in the query.

7 Conclusion

It is important to note that TF-IDF parameters had a far greater impact to model performance than model parameters did. Even so, the greatest accuracy was 40.5%, which is fairly low, even considering that this is four-class classification.

Accuracy in multi-class classification will always be a challenge, since there will tend to be some overlap between clusters and the gap between predicted probabilities will shrink (Koppel and Schler, 2006). For the topic of classification of astrological signs, this is particularly difficult, since it is notoriously regarded as a pseudoscience (Carlson, 1985); it was of more interest *if* there was any pattern, rather than *what* the pattern in the data would be. While scientifically and mathematically vigorous, the topic chosen for this project was generally for fun.

A Appendix

High and Low Results for TF-IDF Parameter Experiment:

Model	Accuracy
{'min_df': 25, 'ngram_range': (1, 1), 'penalty': 'l2'}	0.404
{'min_df': 25, 'ngram_range': (1, 2), 'penalty': 'l2'}	0.400
{'min_df': 25, 'ngram_range': (1, 3), 'penalty': 'l2'}	0.403
{'min_df': 75, 'ngram_range': (1, 2), 'penalty': 'l1'}	0.349
{'min_df': 100, 'ngram_range': (1, 2), 'penalty': 'l1'}	0.348

References

S. Carlson. 1985. *A double-blind test of astrology.* Nature, 318, 419-425.

X. Ding, B. Liu, and P. S. Yu. 2008. *A Holistic Lexicon-Based Approach to Opinion Mining*. Proceedings of the 2008 International Conference on Web Search and Data Mining.

C. Higgins. 1997. Astrology and the Four Elements. Accessnewage.com.

A. Khan, B. Baharudin, L. H. Lee, and K. Khan. 2010. *A Review of Machine Learning Algorithms For Text-Documents Classification*. Journal of Advances in Information Technology, 1(1), 4-20.

Y. Ko, J. Park, and J. Seo. 2004. *Improving text categorization using the importance of sentences.* Information processing & management, 40(1), 65-79.

M. Koppel and J. Schler. 2006. *The Importance of Neutral Examples for Learning Sentiment*. Computational Intelligence.

C.Y. Liang, L. Guo, Z.J. Xia, F.G. Nie, X.X. Li, L. Su, and Z.Y. Yang. 2006. *Dictionary-based text categorization of chemical web pages*. Information processing & management, 42(4), 1017-1029.

B. Liu. 2010. *Sentiment Analysis and Subjectivity*. Handbook of Natural Language Processing (Second ed.).

P.Y. Pawar and S.H. Gawande. 2012. *A comparative study on different types of approaches to text categorization.* International Journal of Machine Learning and Computing, 2(4), 423.

F. Pedgregosa *et al*. 2011. Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830. http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html.

J. Schler, M. Koppel, S. Argamon and J. Pennebaker. 2006. *Effects of Age and Gender on Blogging*. Proceedings of 2006 AAAI Spring Symposium on Computational Approaches for Analyzing Weblogs.

V. Tam, A. Santoso, and R. Setiono. 2002. *A comparative study of centroid-based, neighborhood-based and statistical approaches for effective document categorization.* Object recognition supported by user interaction for service robots (Vol. 4, pp. 235-238). IEEE.