Learning to Recognize Blogs: A Preliminary Exploration

Erik Elgersma and **Maarten de Rijke**
ISLA, University of Amsterdam
Kruislaan 403, 1098SJ Amsterdam, The Netherlands
erik@elgersma.net, mdr@science.uva.nl

Abstract

We present results of our experiments with the application of machine learning on binary blog classification, i.e. determining whether a given web page is a blog page. We have gathered a corpus in excess of half a million blog or blog-like pages and pre-classified them using a simple baseline. We investigate which algorithms attain the best results for our classification problem and experiment with resampling techniques, with the aim of utilising our large dataset to improve upon our baseline. We show that the application of off-the-shelf machine learning technology to perform binary blog classification offers substantial improvement over our baseline. Further gains can sometimes be achieved using resampling techniques, but these improvements are relatively small compared to the initial gain.

1 Introduction

In recent years, weblogs (online journals in which the owner posts entries on a regular basis) have not only rapidly become popular as a new and easily accessible publishing tool for the masses, but its content is becoming ever more valuable as a "window to the world," an extensive medium brimming with subjective content that can be mined and analysed to discover what people are talking about and why. In recent years the volume of blogs is estimated to have doubled approximately every six months. Technorati[1] report that about 11% of internet users are blog readers and that about 70 thousand new blogs are created daily. Popular blogosphere (the complete collection of all blogs) analysis tools estimate the blogosphere to contain anywhere between 20[1] and 24 million[2] blogs at time of writing. Given this growing popularity and size, research on blogs and the blogosphere is also increasing. A large amount of this research is being done on the content provided by the blogosphere and the nature of this content, like for example (Mishne and de Rijke, 2005), or the structure of the blogosphere (Adar et al., 2004).

In this paper, however, we address the task of binary blog classification: given a (web) document, is this a blog or not? Our aim is to base this classification mostly on blog characteristics rather than content. We will by no means ignore content but it should not become a crucial part of the classification process.

Reliable blog classification is an important task in the blogosphere as it allows researchers, ping feeds (used to broadcast blog updates), trend analysis tools and many others to separate real blog content from blog-like content such as bulletin boards, newsgroups or trade markets. It is a task that so far has proved difficult as can be witnessed by checking any of the major blog update feeds such as weblogs.com[3] or blo.gs.[4] Both will at any given time list content that clearly is not a blog. In this paper we will explore blog classification using machine learning to improve blog detection and experiment with several methods to try and further improve the percentage of instances classified correctly.

The main research question we address in this paper is exploratory in nature:

- How hard is binary blog classification?

Put more specifically,

[1] Intelliseek's BlogPulse, http://www.blogpulse.com
[2] Technorati, http://www.technorati.com
[3] Weblogs.com, http://www.weblogs.com
[4] Blo.gs, http://blo.gs

- What is the performance of basic off-the-shelf machine learning algorithms on this task?

and

- Can the performance of these methods be improved using resampling methods such as bootstrapping and co-training?

An important complicating factor is the lack of labeled data. It is widely accepted that given a sufficient amount of training data, most machine learning algorithms will achieve similar performance levels. For our experiments, we will have a very limited amount of training material available. Therefore, we expect to see substantial differences between algorithms.

In this paper we will first discuss related work in the following section, before describing the experiments in detail and reporting on the results. Finally, we will draw conclusions based on the experiments and the results.

2 Related Work

Blog classification is still very much in its infancy and to date no directly related work has been published as far as we are aware. There is, however, work related to several aspects of our experiments.

Nanno et al. (2004) describe a system for gathering a large collection of weblogs, not only those published using one of the many well-known authoring tools but also the hand-written variety. A very much comparable system was developed and used for these experiments. Members of the BlogPulse team also describe blog crawling and corpus creation in some detail (Glance et al., 2004), but their system is aimed more at gathering updates and following active blogs rather than gathering as many blogs in their entirety, as our system is set up to do.

As to the resampling methods used in this paper—bootstrapping and co-training—, Jones et al. (1999) describe the application of bootstrapping to text learning tasks and report very good results applying this method to these tasks. Even though text learning is a very different genre, their results provide hope that the application of this method may also prove useful for our blog classification problem.

Blum and Mitchell (1998) describe the use of separate weak indicators to label unlabeled instances as "probably positive" to further train a learning algorithm and gathered results that suggested that their method has the potential for improving results on many practical learning problems. Indeed their example of web-page classification is in many ways very similar to our binary blog classification problem. In these experiments however we will use a different kind of indicators on the unlabeled data, namely the predictions of several different types of algorithms.

3 Binary blog classification

In our first experiment, we attempted binary blog classification ("is this a blog or not?") using a small manually annotated dataset and a large variety of algorithms. The aim of this experiment was to discover what the performance of readily available, off-the-shelf algorithms is given this task.

We used a broad spectrum of learners implemented in the well-known Weka machine learning toolkit (Witten and Frank, 2005).

3.1 Dataset

For our later resampling experiments, a large amount of data was gathered, as will be explained further on in this paper. To create a dataset for this experiment, 201 blog / blog-like pages were randomly selected from the collection, processed into Weka's arff format and manually annotated. These instances were then excluded from the rest of the collection. This yielded a small but reliable dataset, which we hoped would be sufficient for this task.

3.2 Attribute selection

All pages were processed into instances described by a variety of attributes. For binary blog classification to succeed, we had to find a large number of characteristics with which to accurately describe the data. This was done by manually browsing the HTML source code of several blogs as well as some simple intuition. These attributes range from "number of posts" and "post length" to checking for characteristic phrases such as "Comments" or "Archives" or checking for the use of style sheets. Interesting attributes are the "firstLine" / "lastLine" attributes, which calculate a score depending on the number of tokens found in those lines, which frequently occur in those lines in verified blog posts. The "contentType" attribute does something very similar, but based on the complete clean text of a page rather than particular lines in posts. It counts how many of the 100 most frequent tokens in clean text versions of actual blogs, are found in a page and returns a true

value if more than 60% of these are found, in which case the page is probably a blog. The "frequent terms"-lists for these attributes were generated using a manually verified list gathered from a general purpose dataset used for earlier experiments. A "host"-attribute is also used, which we binarised into a large number of binary host name attributes as most machine learning algorithms cannot cope with string attributes. For this purpose we took the 30 most common hosts in our dataset, which included Livejournal,[5] Xanga,[6] 20six,[7] etc., but also a number of hosts that are obviously not blog sites (but host many pages that resemble blogs). Negative indicators on common hosts that don't serve blogs are just as valuable to the machine learner as the positive indicators of common blog hosts. Last but not least a binary attribute was added that acts as a class label for the instance. This process left us with the following 46 attributes:

Attribute	Type
nrOfPosts	numeric
avgPostLength	numeric
minPostLength	numeric
maxPostLength	numeric
firstLine	numeric
lastLine	numeric
containsBlog	numeric
containsMetaTag	binary
contentType	binary
containsComment	binary
containsPostedBy	binary
containsRSS	binary
containsArchives	binary
containsPreviousPosts	binary
StyleSheetsUsed	binary
livejournal.com	binary
msn.com	binary
wretch.cc	binary
xanga.com	binary
diaryland.com	binary
abazy.com	binary
20six.fr	binary
research101-411.com	binary
search-now700.com	binary
search-now999.com	binary
search-now600.com	binary
20six.co.uk	binary
research-bot.com	binary
blogsearchonline.com	binary
googane.com	binary
typepad.com	binary
findbestnow.com	binary
myblog.de	binary
quick-blog.com	binary
findhererightnow.com	binary
findfreenow.com	binary
websearch010.com	binary
twoday.net	binary
websearch013.com	binary
tracetotal.info	binary
kotobabooks.com	binary
cocolog-nifty.com	binary
20six.de	binary
is-here-online.com	binary
4moreadvice.info	binary
blog	binary

Table 1: Attributes selected for our experiments.

3.3 Experimental setup

For this experiment, we trained a wide range of learners using the manually annotated data and tested using ten-fold cross-validation. We then compared the results to a baseline.

This baseline is based mostly on simple heuristics, and is an extended version of the WWW-Blog-Identify[8] perl module that is freely available online. First of all, a URL check is done which looks for a large number of the well-known blog hosts as an indicator. Should this fail, a search is done for metatags which indicate the use of well-known blog creation tools such as Nucleus,[9] Greymatter,[10] Movable Type[11] etc. Should this also fail, an actual content search is done for other indicators such as particular icons blog creation tools leave on pages ("created using… .gif" etc). Next, the module checks for an RSS feed, and as a very last resort checks the number of times the term "blog" is used on the page as an indicator.

In earlier research, our version of the module was manually tested by a small group of individuals and found to have an accuracy of roughly 80% which means it is very useful as a target to aim for with our machine learning algorithms and a good baseline.

[5] http://www.livejournal.com
[6] http://www.xanga.com
[7] http://www.20six.com

[8] http://search.cpan.org/~mceglows/WWW-Blog-Identify-0.06/Identify.pm
[9] http://nucleuscms.org
[10] http://www.noahgrey.com/greysoft/
[11] http://www.movabletype.org/

3.4 Results: single classifiers

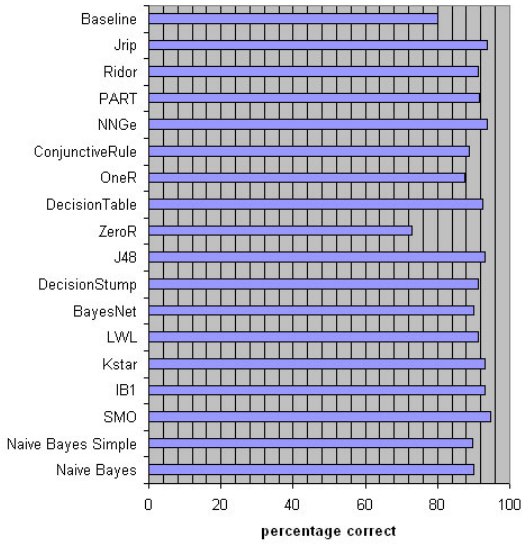

Figure 1: Chart showing the percentage correct predictions for each algorithm tested.

It is clear that all algorithms bar ZeroR perform well, most topping 90%. ZeroR achieves no more than 73%, and is the only algorithm that actually performs worse than our baseline. The best algorithm for this task, and on this dataset, is clearly the support vector-based algorithm SMO, which scores 94.75%. These scores can be considered excellent for a classification task, and the wide success across the range of algorithms shows that our attribute selection has been a success. The attributes clearly describe the data well.

Full results of this experiment can be found in Appendix A.

4 Resampling

Now we turn to the second of our research questions: to what extent can resampling methods help create better blog classifiers.

As reported earlier, the blogosphere today contains millions of blogs and therefore potentially plenty of data for our classifier. However, this data is all unlabeled. Furthermore, we have a distinct lack of reliably labeled data. Resampling may provide us with a solution to this problem and allow us to reliably label the data from our unlabeled data source and further improve upon the results gained using our very small manually annotated dataset.

For these experiments we selected two resampling methods. The first is ordinary bootstrapping, which we chose because it is the simplest way of relabeling unlabeled data on the basis of a machine learning model. Additionally, we chose a modified form of co-training, as co-training is also a well-known resampling method, which was easily adaptable to our problem and seemingly offered a good approach.

4.1 Data set

To gather a large data set containing both blogs and non-blogs, a crawler was developed that included a blog detection module based on the heuristics in our baseline module mentioned earlier. After downloading a page judged likely to be a blog by the module on the basis of its URL, several additional checks were done by the blog detection module based on several other characteristics, most importantly the presence of date-entry combinations. Pages judged to be a blog and those judged not to be even though the URL looked promising, were consequently stored separately. Blogs were stored in html, clean text and single entry (text) formats. For non-blogs only the html was stored to conserve space while still allowing the documents to be fully analysed post-crawling.

Using this system, 227.380 blog- and 285.337 non-blog pages (often several pages were gathered from the same blog, so the actual number of blogs gathered is significantly lower) were gathered in the period from July 7 until November 3, 2005. This amounts to roughly 30Gb of HTML and text, and includes blogs from all the well-known blog sites as well as personal hand-written blogs and in many different languages.

The blog detection module in the crawler was used purely for the purpose of filtering out URLs and webpages that bear no resemblence to a blog. By performing this pre-classification, we were able to gather a dataset containing only blogs and pages that in appearance closely resemble blogs so that our dataset contained both positive examples and useful negative examples. This approach should force the machine learner to make a clear distinction between blogs and non-blogs. However, even though this data was pre-classified by our baseline, we treat it as unlabeled data in our experiments and make no further use of this pre-classification whatsoever.

For our resampling experiments, we randomly divided the large dataset into small subsets containing 1000 instances, one for each iteration. This figure ensures that the training set grows at a reasonable rate at every iteration while preventing the training set from becoming too large too quickly which would mean a lot of unlabeled

instances being labeled on the basis of very few labeled instances and the model building process would take too long after only a few iterations.

For training and test data we turned back to our manually annotated dataset used previously. Of this set, 100 instances were used for the initial training and the remaining 101 for testing.

4.2 Experimental setup: bootstrapping

Generally, bootstrapping is an iterative process where at every iteration unlabeled data is labeled using predictions made by the learner model based on the previously available training set (Jones et al., 1999). These newly labeled instances are then added to the training set and the whole process repeats. Our expectation was that the increase in available training instances should improve the algorithm's accuracy, especially as it proved quite accurate to begin with so the algorihm's predictions should prove quite reliable. For this experiment we used the best performing algorithm from Section 3, the SMO support-vector based algorithm. The bootstrapping method is applied to this problem as follows:

- Initialisation: use the training set containing 100 manually annotated instances to predict the labels of the first subset of 1000 unlabeled instances.
- Iterations: Label the unlabeled instances according to the algorithm's prediction and add these instances to the previous training set to form a new training set. Build a new model based on the new training set and use it to predict the labels of the next subset.

4.3 Results: bootstrapping

We now present the results of our experiment using normal bootstrapping. After every iteration, the model built by the learner was tested on our manually annotated test set.

Iteration	Nr. of training instances	Correctly / incorrectly classified (%)	Precision (yes/no)	Recall (yes/no)
init	100	95.05 / 4.95	0.957 / 0.949	0.846 / 0.987
1	1100	94.06 / 5.94	0.955 / 0.937	0.808 / 0.987
2	2100	94.06 / 5.94	0.955 / 0.937	0.808 / 0.987
3	3100	94.06 / 5.94	0.955 / 0.937	0.808 / 0.987
4	4100	94.06 / 5.94	0.955 / 0.937	0.808 / 0.987
5	5100	94.06 / 5.94	0.955 / 0.937	0.808 / 0.987
6	6100	94.06 / 5.94	0.955 / 0.937	0.808 / 0.987
7	7100	94.06 / 5.94	0.955 / 0.937	0.808 / 0.987
8	8100	93.07 / 6.93	0.952 / 0.925	0.769 / 0.987
9	9100	93.07 / 6.93	0.952 / 0.925	0.769 / 0.987
10	10100	93.07 / 6.93	0.952 / 0.925	0.769 / 0.987
11 - 42	11100 – 42100	92.08 / 7.92	0.95 / 0.914	0.731 / 0.987

Table 2: Overview of results using normal bootstrapping.

After 36 iterations, the experiment was halted as there was clearly no more gain to be expected from any further iterations. Clearly, ordinary bootstrapping does not offer any advantages for our binary blog classification problem. Also, the availability of larger amounts of training instances does nothing to improve results as the results are best using only the very small training set.

Generally, both precision and recall slowly decrease as the training set grows, showing that classifier accuracy as a whole declines. However, recall of instances with class label "no" (non-blogs) remains constant throughout. Clearly the classifier is able to easily detect non-blog pages on the basis of the attributes provided, and is thwarted only by a small number of outliers. This can be explained by the fact that the learner recognizes non-blogs mostly on the basis of the first few attributes having zero values (nrOfPosts, minPostLength, maxPostLength etc.). The outliers consistently missed by the classifier are probably blog-like pages in which date-entry combinations have been found but which nevertheless have been manually classified as non-blogs. Examples of this are calendar pages commonly associated with blogs (but which do not contain blog content), or MSN Space pages on which the user is using the photo album but hasn't started a blog yet. In this case the page is recognized as a blog, but contains no blog content and is therefore manually labeled a non-blog.

4.4 Experimental setup: co-training

As mentioned in Section 2, we will use the predictions of several of the most successful learning algorithms from Section 3 as our indicators

in this experiment. The goal of our co-training experiment is to take unanimous predictions from the three best performing algorithms from Section 3, and use those predictions, which we assume to have a very high degree of confidence, to bootstrap the training set. We will then test to see if it offers an improvement over the SMO algorithm by itself. By unanimous predictions we mean the predictions of those instances, on which all the algorithms agree unanimously after they have been allowed to predict labels using their respective models.

As instances for which the predictions are unanimous can be reasoned to have a very high level of confidence, the predictions for those instances are almost certainly correct. Therefore we expect this method to offer substantial improvements over any single algorithm as it potentially yields a very large number of correctly labeled instances for the learner to train on.

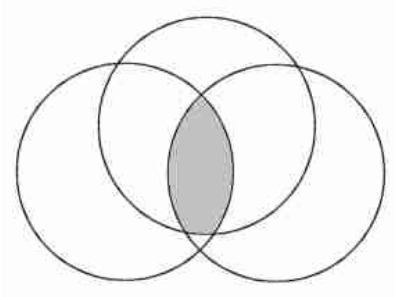

Figure 2: Visual representation of our implementation of the co-training method.

We chose to adapt the co-training idea in this fashion as we believe it to be a good way of radically reducing the fuzziness of potential predictions and a way to gain a very high degree of confidence in the labels attached to previously unlabeled data. Should the algorithms disagree on a large number of instances there would still not be a problem as we have a very large pool of unlabeled instances (133.000, we only used part of our corpus for our experiments as our dataset was so large that there was no need to use all the data available). The potential maximum of 133 iterations should prove quite sufficient even if the growth of the training set per iteration proves to be very small.

The algorithms we chose for this experiment were SMO (support vector), J48 (decision tree, a C4.5 implementation) and Jrip (rule based). We chose not to use nearest neighbour algorithms for this experiment even though they performed well individually as we feared it would prove a less successful approach given the large training set sizes. Indeed, an earlier experiment done during our blog classification research showed the performance of near neighbour algorithms bottomed out very quickly so no real improvement can be expected from those algorithms given larger training sets and given the unanimous nature of this method of co-training it may spoil any gain that might otherwise be achieved.

The process started with the manually annotated training set and used the predictions from the three algorithms, for unlabeled instances they agree unanimously on, to label those instances. Those instances were subsequently added to the trainingset and using this new trainingset, a number of the instances in another unlabeled set (1000 instances per set) were to be labeled (again, only those instances on which the algorithms agree unanimously). Once again, those instances are added to the training set and so on and so forth for as many iterations as possible.

4.5 Results: co-training

We now turn to the results of our experiment using our unanimous co-training method described above. The experiment was halted after 30 iterations, as Weka ran out of memory. The experiment was not re-run with altered memory settings as it was clear that no more gain was to be expected by doing so. Again, testing after each iteration was performed by building a model using the SMO support-vector learning algorithm and testing classifier accuracy on the manually annotated test set.

Iteration	Nr. of training instances	Correctly/ incorectly classified (%)	Precision (yes/no)	Recall (yes/no)
init	100	95.05 / 4.95	0.957 / 0.949	0.846 / 0.987
1	1000	94.06 / 5.94	0.955 / 0.937	0.808 / 0.987
2	1903	93.07 / 6.93	0.952 / 0.925	0.769 / 0.987
3	2798	95.05 / 4.95	0.957 / 0.949	0.846 / 0.987
4	3696	95.05 / 4.95	0.957 / 0.949	0.846 / 0.987
5	4566	95.05 / 4.95	0.957 / 0.949	0.846 / 0.987
6	5458	96.04 / 3.96	0.958 / 0.961	0.885 / 0.987
7	6351	96.04 / 3.96	0.958 / 0.961	0.885 / 0.987
8	7235	95.05 / 4.95	0.957 / 0.949	0.846 / 0.987
9	8149	95.05 / 4.95	0.957 / 0.949	0.846 / 0.987

10	9041	95.05 / 4.95	0.957 / 0.949	0.846 / 0.987
11	9929	95.05 / 4.95	0.957 / 0.949	0.846 / 0.987
12	10810	95.05 / 4.95	0.957 / 0.949	0.846 / 0.987
13 - 43	11684 - 38510	94.06 / 5.94	0.955 / 0.937	0.808 / 0.987

Table 3: Overview of results using our unanimous co-training method.

Even though the "steps" in test percentages shown represent only one more blog being classified correctly (or incorrectly), the classifier does perform better than it did using only the manually annotated training set at some stages of the experiment. This means that gains in classifier accuracy can be achieved by using this method of co-training on this problem. Also the classifier generally performs better than in our bootstrapping experiment, which shows that the instances unanimously agreed on by all three algorithms are certainly more reliable than the predictions of even the best algorithm by itself, as predicted.

Clearly this method offers potential for an improvement even though the SMO algorithm was already very accurate in our first binary blog classification experiment.

5 Discussion

As the title suggests, these experiments are of a preliminary and exploratory nature. The high accuracy achieved by almost all algorithms in our binary classification experiment show that our attribute set clearly defines the subject well. However, these results must be viewed with an air of caution as they were obtained using a small subset and as such the data may not represent the nature of the complete dataset well. Indeed, how stable are the results obtained?

Later experiments using a (disjoin, but) larger manually annotated dataset containing 700 instances show that the results obtained here are optimistic. The extremely diverse nature of the blogosphere means that describing an entire dataset using a relatively small subset is very difficult and as such both the performance and ranking of off-the-shelf machine learning algorithms will vary among different datasets. Off-the-shelf algorithms do however still perform far better than our baseline and the best performing algorithms still achieve accuracy rates in excess of 90%.

Two aspects of our attribute set that need to be worked on in future are date detection and content checks. Outliers are almost always caused by the date detection algorithm not detecting certain date formats, and pages containing date-entry combinations but no real blog content. Therefore, although it is possible to perform binary blog classification based purely on the particular characteristics of blog pages with high accuracy, content checks are invaluable. The rise of blogspam, which cannot be separated from real blogs on the basis of page characteristics at all, further emphasises this. We have already developed a document frequency profile and replaced the contentType attribute used in these experiments, to extend the content-based attributes in our dataset and hopefully improve blog recognition.

6 Conclusion

Our experiments have shown that binary blog classification can be performed successfully if the right attributes are chosen to describe the data, even if the classifier is forced to rely on a small number of training instances. Almost all basic off-the-shelf machine learning algorithms perform well given this task, but support vector based algorithms performed best in this experiment. Notable was that the best algorithms of each type achieved almost the same accuracy, all over 90% and the difference is never larger than a few percent even though they approach the problem in completely different manners.

The performance of these algorithms can be improved by using resampling methods, but not all resampling methods achieve gains and those that do gain very little. The extremely high success rates of the plain algorithms means that there is very little room for improvement, especially as the classification errors are almost always caused by outliers that none of the algorithms manage to classify correctly.

The results of later experiments with larger numbers of manually annotated instances show that a lot of work remains to be done and that although this paper shows that the application of machine learning to this problem offers substantial improvements over our baseline, this problem is still far from solved.

Future work will include further analysis of the results obtained using larger manually annotated subsets as well as a detailed analysis of the contributions of the different features in the feature set described in Section 3.

Acknowledgements

The authors wish to thank Gilad Mishne for his input and valuable comments during these experiments and the writing of this piece.

Maarten de Rijke was supported by grants from the Netherlands Organization for Scientific Research (NWO) under project numbers 017.001.190, 220-80-001, 264-70-050, 354-20-005, 612-13-001, 612.000.106, 612.000.207, 612.066.302, 612.069.006, 640.001.501, and 640.002.501.

References

G. Mishne, M. de Rijke. 2006. Capturing Global Mood Levels using Blog Posts, In: *AAAI 2006 Spring Symposium on Computational Approaches to Analysing Weblogs* (AAAI-CAAW 2006)

E. Adar, L. Zhang, L. Adamic, R. Lukose. 2004. Implicit Structure and the Dynamics of Blogspace, In: *Workshop on the Weblogging Ecosystem, WWW Conference*, 2004

T. Nanno, Y. Suzuki, T. Fujiki, M. Okumura. 2004. Automatic Collection and Monitoring of Japanese Weblogs, In: *Proceeding of WWW2004: the 13th international World Wide Web conference*, New York, NY, USA, 2004. ACM Press.

N. Glance, M. Hurst, T. Tomokiyo. 2004. BlogPulse: Automated Trend Discovery for Weblogs, In: *Proceeding of WWW2004: the 13th international World Wide Web conference*, New York, NY, USA, 2004. ACM Press.

R. Jones, A. McCallum, K. Nigam, and E. Riloff. 1999. Bootstrapping for Text Learning Tasks, In: *IJCAI-99 Workshop on Text Mining: Foundations, Techniques and Applications*, p52-63.

A. Blum, T. Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-Training, In: *Proceedings of the 1998 Conference on Computational Learning Theory*.

I. Witten, E. Frank. 2005. *Data Mining: Practical machine learning tools and techniques*, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

Appendix A. Full results of our binary blog classification experiment

Algorithm	Type	Percentage correct predictions
Naïve Bayes	Bayes	90.07
Naïve Bayes Simple	Bayes	89.64
SMO	**Support Vector**	**94.75**
IB1	Instance based	93.00
KStar	**Instance based**	**93.30**
LWL	Instance based	91.25
BayesNet	Bayes	90.08
DecisionStump	Tree	91.25
J48	**Tree**	**93.29**
ZeroR	Rule-based	73.00
DecisionTable	Rule-based	92.55
OneR	Rule-based	87.60
ConjunctiveRule	Rule-based	88.75
NNGe	Rule-based	93.73
PART	Rule-based	91.67
Ridor	Rule-based	91.26
JRip	**Rule-based**	**93.73**