Adventures in Feature Selection on an Industrial Dataset... and Ensuing General Discoveries

George Forman

HP Laboratories
HPL-2012-161R1

Keyword(s):

text feature selection; text classification; document categorization; lessons learned

Abstract:

We relate the story of an interesting failure of text feature selection methods on an industrial dataset of technical documents. Our detailed dissection and ultimate understanding of the failure led to the creation of general solutions that not only solved the robustness problem we faced, but were also able to improve classification accuracy for simpler, public datasets, which was crucial to enable the works' publishability.

External Posting Date: September 21, 2012 [Fulltext] Approved for External Publication
Internal Posting Date: September 21, 2012 [Fulltext]
To be published in the proceedings of Silver 2012: The Silver Lining: learning from unexpected results, ECML/PKDD 2012 Workshop.

Adventures in Feature Selection on an Industrial Dataset ...and Ensuing General Discoveries (Extended Abstract)

George Forman
HP Labs, Palo Alto, CA, USA

Abstract. We relate the story of an interesting failure of text feature selection methods on an industrial dataset of technical documents. Our detailed dissection and ultimate understanding of the failure led to the creation of general solutions that not only solved the robustness problem we faced, but were also able to improve classification accuracy for simpler, public datasets, which was crucial to enable the works' publishability.[1]

1 The Story

We were developing some simple text classification software for a Hewlett-Packard business division that wanted to sort a large collection of internal tech-support documents into various topic categories. Their previous *rule-based* system had, over the years, grown hard to maintain and was perceived as having poor accuracy. The rules consisted of over 8000 lines for English documents alone, containing several types of pattern matching, many variants of product names including third-party software, and prioritized, hierarchical Boolean logic for categorization into 100+ topics. This was the old state-of-the-art. Given an emailed report of some misclassified documents, it was quite difficult for the rule maintainers (different domain experts for different product lines) to know what to change about the many rules. Plus, any changes might lead to new misclassifications, damaging overall accuracy. It was hard to know, and at the time there were no ground-truth labels recorded from which we could measure its accuracy...nor try out a machine learning solution.

We pushed for machine learning, with hopes for better accuracy and a *much easier* way to improve the system whenever misclassified documents are found—just add them to the training set and retrain. Encouraged, the division worked with their domain experts to provide us a sample dataset of 10 classes, each with about 100 documents, on which we could develop our software and see how its accuracy compared with the existing solution.

This was, we thought, a simple multi-class (single-label) problem, to which we were applying straightforward techniques: traditional Naive Bayes with a bag-of-words vector representation of the text. Though feature extraction is conceptually simple, we had to make some hard decisions about whether '-' and '/' were punctuation or word characters: they were used in many technical names, such

as 'HP-UX', 'MPE/XL', and 'FDDI/9000', and in many part numbers, such as 'C3166-69017'—but of course, there were many situations where they were better thought of as punctuation. We found that feature selection improved classification accuracy substantially, and we believed it would eventually be important for scalability on the full-size problem. We experimented with highly scalable feature filtering methods using Mutual Information, Chi-Squared, or Information Gain, promoted by publications such as Yang and Pedersen [2]. After cross-validation tests, we determined to ship the software using Information Gain to the internal division. Everything had gone smoothly...so far.

1.1 The Persistent Failure

After we delivered the software to the division, they slowly built up their labeled training set for over 230 technical topics. At first the classifications looked rather poor, but we had expected this initially and knew that acquiring additional training labels was necessary. Through discussions, we found some classes were trained inadequately, e.g. collecting its training documents by searching for only a single keyword or two, providing little diversity to learn from. At some point, their persistent perception of its poor results moved us to look for a problem. We first re-tested their use of our software on our sample data to verify its correct operation. After some delay, we were able to obtain a copy of their training data in order to reproduce the problem locally and perform cross-validation. Looking into the features selected, we saw oceans of unfamiliar technical words (e.g. s700_800, PHCO_3238, MIRRORDISK/UX, PHKL_1921), which seemed reasonable enough with our lack of knowledge about the large-scale domain problem. It took awhile for us to notice that words we might reasonably expect to find were missing, e.g. JETDIRECT or JETADMIN. We verified that such features did occur in the labeled training documents, and that the feature extraction routines did offer them. We found that the feature selection algorithm consistently threw them out, even if we wildly increased the number of selected features. We tested other feature selection functions and tried adjusting various parameters, such as the rare word cutoff and an assortment of choices for Laplace correction. We tried programming an assortment of additional methods for feature selection from the literature: different functions as well as different ways of aggregating them over classes. In measuring the Information Gain of a feature, one can alternatively measure it separately for each class vs. all others, and then aggregate the measurements for each of the 230+ classes in various ways: average, maximum, etc.[1] Nothing worked to bring in the expected features. Perhaps they weren't such good features as we had thought, since every method refused to include them.

[1] Wrapper methods that search over power-sets of features were completely out of the question for the scale of the problem.

1.2 The Root Cause

It is rather difficult to scrutinize a confusion matrix from cross-validation with over 230 classes, but with some effort, we drew our attention to a class that was nicely accurate. Perhaps we might find a clue if we found out what was different about this class? It contained reports of standard HP/UX patch bundles.[2] These documents looked somewhat unusual to our inexperienced eye, and they contained listings of many inscrutable patch names that were included or superseded by the current focus patch bundle. As a result, many early patch names occurred in many of the documents for this class. These made for excellently predictive features that rarely occurred in the other classes. *Aha!* Because there were incredibly many of these strong features, their excellent Information Gain scores—by whatever method we used—consistently crowded out most of the highly predictive features needed to distinguish other classes. Who would have thought that an abundance of good features available should be a problem?

1.3 The Solution

Clearly, the Information Gain scores of those patch features were excellent, but there was another selection criterion we needed to add: some notion of *fairness*. A difficult class with only weakly predictive features available would still need to get some of its best features selected, even if they had Information Gain scores that were completely uncompetitive with the scores of other features for other classes. This led to the development of the *SpreadFx* algorithm family [1], which is highly scalable and includes the idea of round-robin feature selection, giving each class (proportionate to its need or importance) equal chances to nominate features to be included. This gave good performance and proved much more robust for selecting features on such odd datasets. But there's more to the story.

1.4 The Research Path to its Eventual Publication

Though the solution was quite useful to us for this and other asymmetric classification tasks—and perhaps useful to others facing such anomalous datasets—would it be publishable? We suspected and confirmed that reviewers would balk at publishing our 'heuristic' and 'ad hoc' solution to such an 'atypical' problem, which 'should have treated the odd class separately to begin with.' Of course, this is perhaps good advice if you can afford to tailor the solution for each dataset. We regretted to be informed that our submission, though interesting, was not expected to be 'workable in other datasets.' Naturally, benchmark classification

[2] Although this class had not previously been present in the rules solution, it had been added to the problem for the 'go-forward' solution, in addition to taking the opportunity for reorganizing and refining some of the topic categories offered by the rules classifier. This, of course, made it difficult to compare results directly with the old rules classifier, but by examining the precision for some categories that were in common, we were ultimately able to validate the superiority of the machine learning solution.

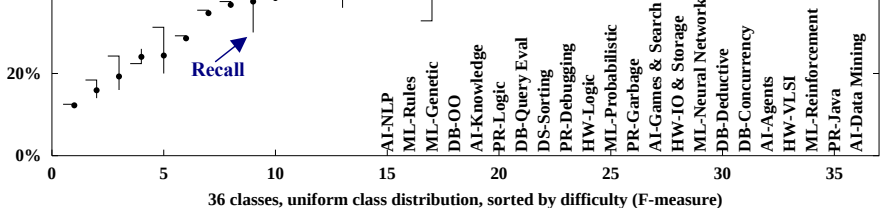

Figure 1. Baseline precision, recall and F-measure performance for each class of the Cora dataset. The 36 classes each have 50 training examples and are ordered here by their F-measure. The classifier—SVM on 500 features selected by Information Gain—obtained 50% accuracy (& F-measure) overall.

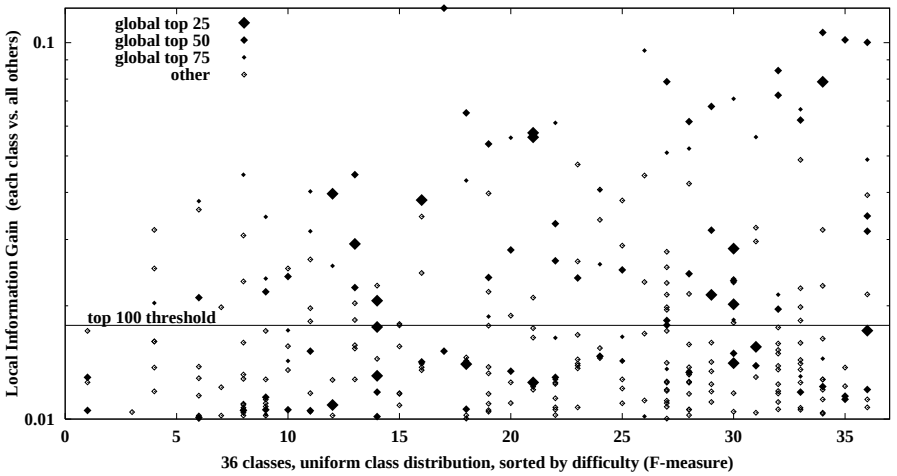

Figure 1. Information Gain scores of top features. Each column corresponds to one class (sorted by F-measure). We plot the local Information Gain of each feature in distinguishing that class from all other classes. Additionally, we indicate the top *global* Information Gain scores via point shapes, e.g. large diamonds mark features included in the global top 25 features.

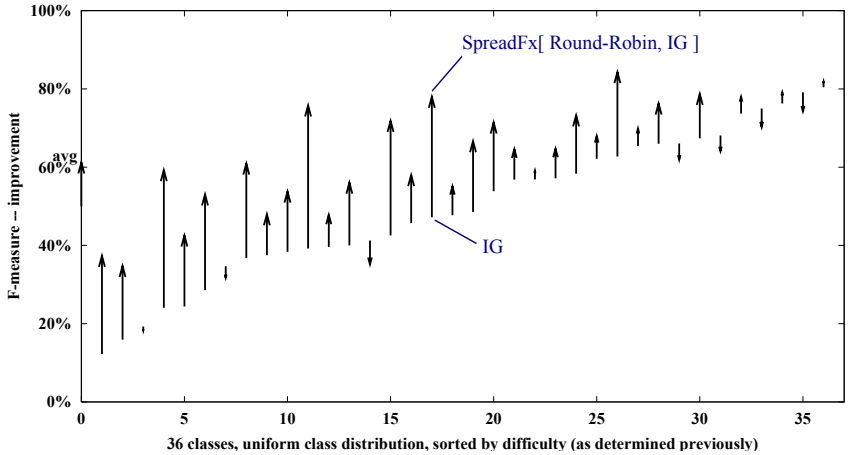

Figure 2. F-measure achieved by SpreadFx[Round-Robin, IG] (at arrow tip) vs. the traditional IG (at arrow tail).

in this case because the class distribution is uniform.) In this experiment, we achieved an overall F-measure of 61.2%, up 22% from the previous baseline of 50% for traditional IG. (We repeated this experiment for Naïve Bayes and saw a similar improvement of 12% overall.)

Sets from UCI and elsewhere didn't exhibit such weird asymmetries, so there appeared to be little call for a method to treat it. Or was there?

To investigate its value more broadly, we had to go deeper. We suspected that even with typical symmetric datasets, there would always be some classes that are easier than others, and that the feature selection phase could help the classifier by making sure to collect plenty of features for the difficult classes. We constructed an artificially balanced, 36-class dataset of computer science topics such that it had exactly the same number of documents in each class. Even on such an abnormally symmetric dataset, the difficulty of the individual classes varied, some having much better features available than others, as shown in the Figure 1 (reproduced from [1]). Using this dataset, we found that SpreadFx improved the classification accuracy for most of the classes (Figure 2), resulting in a substantial gain overall. The performance advantage for this

Table 1. Benchmark Datasets

Dataset	Source	Docs	Words	Classes
Cora	WhizBang	8500	31710	36
fbis	TREC	2463	2000	17
La1	TREC	3204	31472	6
La2	TREC	3075	31472	6
Oh0	OHSUMED	1003	3182	10
Oh5	OHSUMED	918	3012	10
Oh10	OHSUMED	1050	3238	10
Oh15	OHSUMED	913	3100	10
ohscal	OHSUMED	11162	11465	10
Re0	Reuters	1504	2886	13
Re1	Reuters	1657	3758	25
tr11	TREC	414	6429	9
tr12	TREC	313	5804	8
tr21	TREC	336	7902	6
tr23	TREC	204	5832	6
tr31	TREC	927	10128	7
tr41	TREC	878	7454	10
tr45	TREC	690	8261	10
wap	WebACE	1560	8460	20

4.1 Improvement over all 19 Datasets

Next we present an evaluation over a large classification benchmark to test the merit of SpreadFx applied to the widely practiced IG and CHI methods. Certainly as we increase the number of features to a very large number, any feature selection algorithm will begin to provide many predictive features for all classes. So the primary hypothesis to test is whether the benefit of SpreadFx is beneficial at smaller numbers of selected features. That said, we would also like to see a gain for larger numbers of features selected.

For the induction algorithm, we chose the multi-class Support Vector Machine (SVM), as it is considered among the best in class for text classification, and quite popular (e.g. Yang & Liu, 1999; Joachims, 1998). We initially expected that it would be difficult to improve SVM results. To show that the results are not particular to SVM, we also demonstrate similar improvement for the traditional Naïve Bayes classifier, which is more highly sensitive to feature selection.

We obtained our benchmarks as the Cora dataset plus 18 other text datasets provided by Han and Karypis (2000). Refer to Table 1. The classification tasks are drawn from standard benchmarks sought for Reuters, OHSUMED, and TREC, among others. The datasets range from M=6 to 36 classes, 2,000 to 31,000 binary features, and have uneven class distributions with a median of 1 positive to 17 negative training examples (and average 1:31). No class is duplicated in different datasets. For a detailed exposition of the datasets, please refer to their paper or else Forman (2003). We will gladly make the feature vectors available on request.

For each dataset and feature selection scheme, we perform 4-fold cross-validation runs, obtaining the macro-averaged F-measure across all the classes of the dataset. We then average these results across five random stratified cross-validation splits for each of the 19 datasets. (The results for accuracy and even micro-

unnaturally balanced dataset and 18 more normal datasets finally persuaded reviewers that the method was worth publishing, and it has gained over 40 citations to date.

2 Conclusions

There are three main take-away messages, each for a different audience. For *data mining practitioners*, there is the useful result of the SpreadFx algorithm [1] for robust, performant, and scalable multi-class feature selection. Additionally, the anecdote reminds us that we need to test our systems in settings and on datasets that are as realistic as possible; simplified versions may not reveal serious, lurking problems. For *individual researchers*, we need to get our hands on real-world, unsimplified datasets and tasks in order to discover those problems and to drill down into them to gain useful insights for innovation. For the *broader research community*, we face a persistent issue that it is difficult to publish failures or to publish results on proprietary datasets. Reviewers far prefer elegant work on a conceptually simple problem with improvements demonstrated on public benchmarks for reproducible science. But these simpler problems and public datasets are limited in scope and usually exclude the messy, real-world structure that comes with many interesting and worthwhile problems. Consider the SpreadFx work: what reviewer would take seriously a test problem concocted from public datasets with 100 topic classes plus one extremely different class, say, German documents? The work was only made publishable because it could *also* demonstrate gains for normal, public datasets as well. Yet it has a robustness benefit that is needed for at least some real-world business problems. For the ongoing progress of our science, we increasingly need to face real-world, sometimes messy datasets and, where important new problems are exposed, to accept publication on datasets that will not be publicly available.

References

1. G. Forman. A pitfall and solution in multi-class feature selection for text classification. In *Proceedings of the 21st International Conference on Machine Learning*, ICML '04, pages 38–45, Banff, Canada, 2004. See also HP Labs Technical Report 2004-86, www.hpl.hp.com/techreports/2004/HPL-2004-86.html.
2. Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, ICML '97, pages 412–420, Nashville, USA, 1997.