# A Primal/Dual Stump Algorithm for Large Numerical Datasets

Patrick Haffner
AT&T Labs Research, haffner@research.att.com

February 8, 2008

We demonstrate a stochastic gradient algorithm that can handle the very large number of *stump* features generated by considering every possible threshold over numerical, or *continuous*, features.

Our problem is to classify data with continuous features, where small variations in the feature value can result in a different classification decision. Consider for instance packet statistics used for network analysis: features such as packet counts and durations need to be finely thresholded to achieve accurate decisions.

To be used as input to vectorial classifiers, these features are often normalized between -1 and +1, or standardized to the unit variance: much of the separation power of the feature is then lost. A more powerful way to build a finely discriminant classifier looks at every possible partition of the input space that can separate training examples, and perform a weighted combination of these partitions. In practice, only coordinate-wise partitions are possible: given feature f and threshold $\theta$, the data is split into two sets $\{f < \theta\}$ and $\{f \geq \theta\}$. State-of-the-art algorithms that combine partitions include classification trees and boosted stumps [4]. As a matter of fact, a recent comparative study suggests the best algorithm on continuous data is boosted trees [2]. One explanation is that the input space used by boosted trees or stumps is much larger than the space used by SVMs, and that the traditional kernel trick does not help much.

Unfortunately, boosting and classification tree algorithms become very cumbersome to apply on very large datasets, as they require a pass through the entire data for each weak classifier to be added. On the other hand, SVMs have recently been shown to be amenable to online implementations [1, 5]. In this work, we use the SVM paradigm to describe regularized linear classifiers, but one could also apply the Maximum Entropy or the generalized Perceptron paradigms with very similar results.

We show that SVM can handle a stump representation as rich as classification trees, and we demonstrate an algorithm that is mostly stochastic. The selection of stumps starts by looking, for each feature, at every possible threshold that partitions the training data in a different way, that is every different value the training data takes for this feature. This implies that each one of the $n$ continuous feature can correspond to up to $l$ stump features, where $l$ is the number of training examples. The dimension of the resulting stump space can reach a $nl$ dimension. This work shows how to deal with this explosive number of features by playing with alternative representations of the data and the weight vector.

**Kernel trick using an ordinal representation.** Using an ordinal representation of each feature value as its position in a sorted list, one can represent the dot product in the high dimensional stump space as a kernel that only requires $O(n)$ operations. Unlike previously studied stump kernels [3], this kernel only depends on the ordering of the examples along a given feature, like classification trees. It can be further composed with polynomial kernels to obtain the same representation power as trees.

To move from the dual to the primal, we introduce two transforms over the stump space, namely a discrete integration (sigma transform) and derivation (delta transform), that provide powerful tools to transform full dot products into sparse dot products. In particular, we show that the dot-product between a stump-encoded example $\mathbf{x}$ and a weight vector $\mathbf{w}$ can be computed as a dot product between the delta transform of $\mathbf{x}$ and the sigma transform of $\mathbf{w}$, with only $O(n)$ operations. This leads to a fast primal computation of the scores at test time.

**SGD training in the primal space.** The delta transform of $\mathbf{w}$ can be updated using a sparse stochastic gradient descent (SGD) algorithm [5]. However, this algorithm is penalized by the cost of the double integration operation required to construct the sigma transform of $\mathbf{w}$, and is only worth considering for batch updates.

Our solution is a primal/dual training trade-off. We modify the algorithm to maintain 3 representations of the weights: sigma, delta and dual. The dual representation is used in the computation of the score, representing the contribution of the most recent modification to the weights that have not been translated in the sigma weight representation. It acts as a buffer, and allows weight updates at each iteration, while only reconstructing the sigma weight rarely. This is, to our knowledge, the first case where a primal/dual learning strategy pays off.

**Experiments.** SVM classification using stump features was shown to outperform both standard SVMs and classification trees on prosody classification and video segmentation problems. To study scaling, our experimental data consists of more than 500 Million flow records describing Internet traffic[1], and the goal is to classify the record as legitimate, or part of a known class of intrusion. Truth is obtained automatically, with the SNORT software[2]. SNORT has access to packet data, which is much more complete than flow records, therefore the goal of our classifier is to emulate a rule-based system using an input that has been severely summarized. Continuous features include packet and byte counts. AdaBoost with stumps [4] yields excellent performance on this relatively clean data. However, subsampling was necessary to be get Adaboost to learn a single class in less than one day. The SGD primal/dual algorithm was able to reach comparable performance with minutes of training time. Standard SVMs perform badly on this data, as they fail to separate classes that are identified because of a very specific signature.

## References

[1] Antoine Bordes, Seyda Ertekin, Jason Weston, and Léon Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6:1579–1619, September 2005.

[2] Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 161–168, New York, NY, USA, 2006. ACM Press.

[3] Hsuan-Tien Lin and Ling Li. Infinite ensemble learning with support vector machines. In *Machine Learning: ECML 2005*, volume 3720, pages 242–254, 2005.

[4] Robert E. Schapire and Yoram Singer. Improved boosting using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[5] Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 807–814, New York, NY, USA, 2007. ACM Press.

[1] http://en.wikipedia.org/wiki/Netflow

[2] http://en.wikipedia.org/wiki/Snort_(software)