Adaptive Dual Greedy: Using an LTF evaluation algorithm to reduce the cost of using SVM

Devorah Kletenik DKLETENIK@CS.POLY.EDU
Lisa Hellerstein HSTEIN@POLY.EDU
Polytechnic Institute of NYU, 2 Metrotech Center, Brooklyn, NY 11201

Abstract

During test-time, SVM evaluation can be expensive if the features are time-consuming to extract. This work uses an approximation algorithm for the submodular set cover problem to select a subset of all features that may be used during test-time to arrive at the same accuracy as if all features were used. On synthetic and UCI datasets this approach shows reductions in prediction cost, while maintaining prediction accuracy.

1. Introduction

Support Vector Machines (SVM) are a popular machine learning classifier. SVMs labels examples as positive and negative by computing a hyperplane in multidimensional space such that the margin between the positive and negative examples is maximized. SVMs have high accuracy and are frequently used in many applications. However, a drawback to the use of SVMs is the high cost associated with their use. In order to classify an example, the SVM must use all of the features of that example. Hence, the cost of classification is the total cost of all of the features. In applications where there is a cost associated with procuring the value of a feature (e.g. medical diagnostics and quality control tests in a factory setting), using an SVM can be prohibitively expensive. The impetus behind our work was to find a way to decrease the cost of testing using an SVM while retaining the high accuracy.

2. Adaptive Dual Greedy

The problem of finding the optimal evaluation strategy for linear threshold functions (LTF) is NP-hard (Cox et al., 1989). In previous work (Deshpande et al., 2013), we have developed an evaluation algorithm for linear threshold functions, which we call Adaptive Dual Greedy, that is based on the submodular dual greedy algorithm by Fujito (Fujito, 1999). This algorithm provably outputs an evaluation tree that is within a factor of 3 of the optimal evaluation tree. Although the bounds in (Deshpande et al., 2013) apply to boolean input, the bounds extend to real-valued input as well. We propose using this algorithm as a postprocessor for SVMs (or other learning algorithms that output LTF classifers) on binary attributes. To apply our algorithm, we assume independence of all attributes. We use the input data to estimate, for each attribute and for each value of that attribute in the input data, the probability of that attribute having that value in the test data.

We reproduce the pseudocode of Adaptive Dual Greedy below.

Algorithm 1 Adaptive Dual Greedy

$b \leftarrow (*, * \ldots, *)$, $y_S \leftarrow 0$ for all $S \subseteq N$
$F^0 = \emptyset$, $l \leftarrow 0$
while b is not a solution to SSSC ($g(b) < Q$) **do**
  $l \leftarrow l + 1$
  $j_l \leftarrow \arg\min_{j \notin F^{l-1}} \frac{c_j - \sum_{S: y_S \neq 0}(E[g_{S,b}(j)])y_S}{E[g_b(j)]}$
  $y_{F^{l-1}} \leftarrow \frac{c_{j_l} - \sum_{S: y_S \neq 0}(E[g_{S,b}(j)])y_S}{E[g_b(j)]}$
  $k \leftarrow$ the state of j_l {‘}‘test” j_l
  $F^l \leftarrow F^{l-1} \bigcup \{j_l\}$ {}$F^l = dom(b)$
  $b_{j_l} \leftarrow k$
end while
return b

The algorithm is based on the dual of the LP relaxation of Wolsey's IP for the (deterministic) Submodular Set Cover Problem. The main idea behind the algorithm is to iteratively assign values to a subset of the y_S variables, which are all set initially to 0. The next variable to be assigned a value is chosen in a greedy way, and the assignment is done to maintain the invariant that the current assignment to the y_S variables is a feasible solution to the dual LP. Each assignment causes another constraint of the dual to be

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

tight. The item j corresponding to that constraint is then added to an initially empty set J. When J is a cover, the algorithm halts and outputs J. Please refer to (Deshpande et al., 2013) for a fuller description and analysis.

3. Comparison of Adaptive Dual Greedy to other strategies for evaluating LTF

We compare the performance of our algorithm on LTFs to three naive heuristics for evaluating linear threshold functions. The first evaluates all attribute in sequential order (x_i will be tested before x_j for all $i < j$); the second tests according to the fixed order of decreasing costs (x_i will be tested before x_j for all $c_i < c_j$ where c_i refers to the cost of testing attribute x_i); and the third tests according to the fixed order of increasing coefficient over cost (x_i will be tested before x_j for all $\frac{|a_i|}{c_i} > \frac{|a_j|}{c_j}$ where $|a_i|$ refers to the absolute value of the coefficient in the LTF of attribute x_i).

It is clear that the above heuristics have no theoretical guarantees as to the optimality of their solutions, and in fact, the cost of evaluating an LTF according to one of the heuristics can be as much as n times the cost of evaluating it according to Adaptive Dual Greedy. For example, consider the case of boolean data labeled according to an OR function, where all costs and coefficients are 1, and the probability that $x_i = 1$ is .0001 for $0 < i < n-2$ and .9999 for x_{n-1}. Then, assuming that ties are broken in favor of smaller subscripts (x_i before x_j for $i < j$), all three heuristics of the previous paragraph will find solutions of cost $= O(n)$. In contrast, with high probability, ours will find a solution of cost equal to 1.

We compare all three methods of evaluating linear threshold functions both over synthetic data (where the probabilities, costs, and coefficients are all randomly generated) and on two UCI datasets (with randomly generated costs). As the figures below demonstrate, our algorithm finds a solution that is comparable in cost to the solutions found by the heuristics.

4. Results

We first present our results for synthetic real-valued data. This dataset has twenty attributes, each with a random real-valued value between 0 and 9.99. Costs (between 1 and 10) and coefficients (between -10 and +10, with a threshold at most 10) were generated randomly as well. The data was labelled according to the value of the linear threshold function induced by the coefficients.

Table 1 depicts the results of the various LTF evaluation algorithms, averaged over five runs of 500 examples each. (500 training examples were generated as well, to be used in determining the probability distribution of each attribute.)

Table 1. Synthetic real-valued data

method of evaluation	cost
Heuristic 1 (increasing i)	184.5
Heuristic 2 (increasing cost)	150.5
Heuristic 3 (decreasing $\frac{|a_i|}{c_i}$)	117.5
Adaptive Dual Greedy	117.5
Cost of evaluating all attributes	240

Next, we evaluated our algorithms on several UCI datasets (Bache & Lichman, 2013). Due to space constraints, we present here the results from two datasets: congressional voting records and heart disease (R. Detrano, 1988). We randomly split the data into 2/3 training to train the Weka SMO and 1/3 test. Costs, between 1 and $n/2$, where n is the number of attributes for the dataset, were randomly assigned as well. Real-valued attributes were standardized (scaled so that the mean is centered and there is unit variance). We present the results, averaged over five rounds of splitting and testing. Table 2 presents the results of evaluating the congressional voting records dataset (16 boolean attributes, 435 examples).

Table 2. congressional voting records dataset

method of evaluation	cost
Heuristic 1 (increasing i)	42
Heuristic 2 (increasing cost)	47
Heuristic 3 (decreasing $\frac{|a_i|}{c_i}$)	24
Adaptive Dual Greedy	32
Cost of running SVM alone	72

Table 3 presents the results of evaluating the Cleveland heart disease dataset (14 boolean and real-valued attributes, 303 examples).

Table 3. Cleveland heart disease dataset

method of evaluation	cost
Heuristic 1 (increasing i)	35
Heuristic 2 (increasing cost)	22
Heuristic 3 (decreasing $\frac{|a_i|}{c_i}$)	20
Adaptive Dual Greedy	23
Cost of running SVM alone	78

As can be seen, the use of Adaptive Dual Greedy as a postprocessor on an SVM can reduce the costs by approximately 50%-70% without sacrificing any accuracy.

5. Conclusion

The use of Adaptive Dual Greedy to classify examples instead of using the SVM output directly can reduce the cost of classification by 50%-70% without sacrificing any accuracy. The cost of the evaluation strategy produced by Adaptive Dual Greedy is comparable to the cost of the strategies produced by naive heuristics. However, the naive heuristics lack the theoretical guarantees of the cost of the solution relative to the optimal cost that Adaptive Dual Greedy has, and in fact can fail miserably on some input.

We would like to consider extending the algorithm to classification examples that have k classes for $k > 2$. We look forward to testing its use on other datasets and anticipate additional favorable results. We view the use of Adaptive Dual Greedy as a postprocessor to SVMs to maintain high accuracy without incurring high costs, as a novel way of reducing test-time cost of classification.

References

Bache, K. and Lichman, M. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml.

Cox, L., Qiu, Y., and Kuehner, W. Heuristic least-cost computation of discrete classification functions with uncertain argument values. *Annals of Operations Research*, 21:1–29, 1989. ISSN 0254-5330. URL http://dx.doi.org/10.1007/BF02022091.

Deshpande, A., Hellerstein, L., and Kletenik, D. Approximation algorithms for stochastic boolean function evaluation and stochastic submodular set cover. *http://arxiv.org/abs/1303.0726*, 2013.

Fujito, T. On approximation of the submodular set cover problem. *Operations Research Letters*, 1999. ISSN 0167-6377. doi: 10.1016/S0167-6377(99)00045-0. URL http://www.sciencedirect.com/science/article/pii/S0167637799000450.

R. Detrano, M.D., Ph.D. V.A. medical center, Long Beach and Cleveland Clinic Foundation, 1988.