Feature-Budgeted Random Forest

Feng Nan FNAN@BU.EDU
Joseph Wang JOEWANG@BU.EDU
Venkatesh Saligrama SRV@BU.EDU
Boston University, 8 Saint Mary's Street, Boston, MA

Abstract

We seek decision rules for *prediction-time cost reduction*, where complete data is available for training, but during prediction-time, each feature can only be acquired for an additional cost. We propose a novel random forest algorithm to minimize prediction error for a user-specified *average* feature acquisition budget. While random forests yield strong generalization performance, they do not explicitly account for feature costs and furthermore require low correlation among trees, which amplifies costs. Our random forest grows trees with low acquisition cost and high strength based on greedy minimax cost-weighted-impurity splits. Theoretically, we establish near-optimal acquisition cost guarantees for our algorithm. Empirically, on a number of benchmark datasets we demonstrate competitive accuracy-cost curves against state-of-the-art prediction-time algorithms.

1. Introduction

In many applications such as surveillance and retrieval, we acquire measurements for an entity, and features for a query in order to make a prediction. Features can be expensive and complementary, namely, knowledge of previously acquired feature values often renders acquisition of another feature redundant. In these cases, the goal is to maximize prediction performance given a constraint on the average feature acquisition cost. Our proposed approach is to learn decision rules for prediction-time cost reduction (Kanani & Melville, 2008) from training data in which the full set of features and ground truth labels are available for training.

We propose a novel random forest learning algorithm to minimize prediction error for a user-specified *average* feature acquisition budget. Random forests (Breiman, 2001) construct a collection of trees, wherein each tree is grown by random independent data sampling & feature splitting, producing a collection of independent identically distributed trees. The resulting classifiers are robust, are easy to train, and yield strong generalization performance.

Although well suited to unconstrained supervised learning problems, applying random forests in the case of prediction-time budget constraints presents a major challenge. First, random forests do not account for feature acquisition costs. If two features have similar utility in terms of power to classify examples but have vastly different costs, random forest is just as likely to select the high cost feature as the low cost alternative. This is obviously undesirable. Second, a key element of random forest performance is the diversity amongst trees (Breiman, 2001). Empirical evidence suggest a strong connection between diversity and performance, and generalization error is bounded not only with respect to the strength of individual trees but also the correlation between trees (Breiman, 2001). High diversity amongst trees constructed without regard for acquisition cost results in trees using a wide range of features, and therefore a high acquisition cost (See Section 3).

Thus, ensuring a low acquisition cost on the forest hinges on growing each tree with high discriminative power and low acquisition cost. To this end, we propose to learn decision trees that incorporates feature acquisition cost. Our random forest grows trees based on greedy minimax cost-weighted-impurity splits. Although the problem of learning decision trees with optimally low-cost is computationally intractable, we show that our greedy approach outputs trees whose cost is closely bounded with respect to the optimal cost. Using these low cost trees, we construct random forests with high classification performance and low prediction-time feature acquisition cost.

Abstractly, our algorithm attempts to solve an empirical risk minimization problem subject to a budget constraint. At each step in the algorithm, we add low-cost trees to the random forest to reduce the empirical risk until the budget constraint is met. The resulting random forest adaptively

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37. Copyright 2015 by the author(s).

acquires features during prediction time, with features only acquired when used by a split in the tree. In summary, our algorithm is greedy and easy to train. It can not only be parallelized, but also lends itself to distributed databases. Empirically, it does not overfit and has low generalization error. Theoretically, we can characterize the feature acquisition cost for each tree and for the random forest. Empirically, on a number of benchmark datasets we demonstrate superior accuracy-cost curves against state-of-the-art prediction-time algorithms.

Related Work: The problem of learning from full training data for prediction-time cost reduction (MacKay, 1992; Kanani & Melville, 2008) has been extensively studied. One simple structure for incorporating costs into learning is through detection cascades (Viola & Jones, 2001; Zhang & Zhang, 2010; Chen et al., 2012), where cheap features are used to discard examples belonging to the negative class. Different from our apporach these approaches require a fixed order of features to be acquired and do not generalize well to multi-class. Bayesian approaches have been proposed which model the system as a POMDP (Ji & Carin, 2007; Kapoor & Horvitz, 2009; Gao & Koller, 2011), however they require estimation of the underlying probability distributions. To overcome the need to estimate distributions, reinforcement learning (Karayev et al., 2013; Busa-Fekete et al., 2012; Dulac-Arnold et al., 2011) and imitation learning (He et al., 2012) approaches have also been studied, where the reward or oracle action is predicted, however these generally require classifiers capable of operating on a wide range of missing feature patterns.

Supervised learning approaches with prediction-time budgets have previously been studied under an empirical risk minimization framework to learn budgeted decision trees (Xu et al., 2013; Kusner et al., 2014; Trapeznikov & Saligrama, 2013; Wang et al., 2014b;a). In this setting, construction of budgeted decision cascades or trees has been proposed by learning complex decision functions at each node and leaf, outputting a tree of classifiers which adaptively select sensors/features to be acquired for each new example. Common to these systems is a decision structure, which is a priori fixed. The entire structure is parameterized by complex decision functions for each node, which are then optimized using various objective functions. In contrast we build a random forest of trees where each tree is grown greedily so that global collection of random trees meets the budget constraint. (Xu et al., 2012) incorporates feature acquisition cost in stage-wise regression during training to achieve prediction-time cost reduction.

Construction of simple decision trees with low costs has also been studied for discrete function evaluation problems (Cicalese et al., 2014; Moshkov, 2010; Bellala et al., 2012). Different from our work these trees operate on discrete data to minimize function evaluations, with no notion of test time prediction or cost.

As for Random forests despite their widespread use in supervised learning, to our knowledge they have not been applied to prediction-time cost reduction.

2. Feature-Budgeted Random Forest

We first present the general problem of learning under prediction-time budgets similar to the formulation in (Trapeznikov & Saligrama, 2013; Wang et al., 2014b). Suppose example/label pairs (x, y) are distributed as $(x, y) \overset{d}{\sim} H$. The goal is to learn a classifier f from a family of functions $\mathcal{F}$ that minimizes expected loss subject to a budget constraint:

$$\min_{f \in \mathcal{F}} E_{xy} \left[L(y, f(x)) \right], \text{ s.t. } E_x \left[C\left(f, x\right) \right] \leq B, \quad (1)$$

where $L(y, \hat{y})$ is a loss function, $C(f, x)$ is the cost of evaluating the function of f on example x and B is a user specified budget constraint. In this paper, we assume that the feature acquisition cost $C(f, x)$ is a modular function of the support of the features used by function f on example x, that is acquiring each feature has a fixed constant cost. Without the cost constraint, the problem is equivalent to a supervised learning problem, however, adding the cost constraint makes this a combinatorial problem (Xu et al., 2013). In practice, we are not given the distribution but instead are given a set of training data $(x_1, y_1), \ldots, (x_n, y_n)$ drawn IID with $(x_i, y_i) \overset{d}{\sim} H$. We can then minimize the empirical loss subject to a budget constraint:

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)), \text{ s.t. } \frac{1}{n} \sum_{i=1}^{n} C\left(f, x_i\right) \leq B. \quad (2)$$

In our context the classifier f is a random forest, $\mathcal{T}$, consisting of K random trees, $D_1, D_2, \ldots, D_K$, that are learnt on training data. Consequently, the expected cost for an instance x during prediction-time can be written as follows:

$$E_f \left[E_x \left[C\left(f, x\right) \right] \right] \leq \sum_{j=1}^{K} E_{D_j} \left[E_x \left[C\left(D_j, x\right) \right] \right] \quad (3)$$

where, in the RHS we are averaging with respect to the random trees. As the trees in a random forest are identically distributed the RHS scales with the number of trees. This upper-bound captures the typical behavior of a random forest due to the low feature correlation among trees.

As a result of this observation, the problem of learning a budgeted random forest can be viewed as equivalent to the problem of finding decision trees with low expected evaluation cost and error. This motivates our algorithm BUDGETRF, where greedily constructed decision trees with provably low feature acquisition cost are added until the budget constraint is met according to validation data. The returned random forest is a feasible solution to (1) with strong empirical performance.

2.1. Our Algorithm

During Training: As shown in Algorithm 1, there are seven inputs to BUDGETRF: impurity function F, prediction-time feature acquisition budget B, a cost vector $C \in \Re^m$ that contains the acquisition cost of each feature, training class labels y_{tr} and data matrix $X_{tr} \in \Re^{n \times m}$, where n is the number of samples and m is the number of features, validation class labels y_{tv} and data matrix X_{tv}. Note that the impurity function F needs to be *admissible*, which essentially means monotone and supermodular. We defer the formal definition and theoretical results to Section 2.2. For now it is helpful to think of an impurity function F as measuring the heterogeneity of a set of examples. Intuitively, F is large for a set of examples with mostly different labels and small for a set with mostly the same label.

Algorithm 1 BUDGETRF

```
1: procedure BUDGETRF($F, B, C, ytr, Xtr, ytv, Xtv$)
2:     $\mathcal{T} \leftarrow \emptyset$.
3:     while Average cost using validation set on $\mathcal{T} \leq B$ do
4:         Randomly sample $n$ training data with replacement to form $X^{(i)}$ and $y^{(i)}$.
5:         Train $T \leftarrow$ GREEDYTREE($F, C, y^{(i)}, X^{(i)}$).
6:         $\mathcal{T} \leftarrow \mathcal{T} \cup T$.
7:     return $\mathcal{T} \backslash T$.
```

Subroutine - GREEDYTREE

```
8: procedure GREEDYTREE($F, C, y, X$)
9:     $S \leftarrow (y, X)$                ▷ the current set of examples
10:    if $F(S) = 0$ then return
11:    for each feature $t = 1$ to $m$ do
12:        Compute $R(t) := \min_{g_t \in \mathcal{G}_t} \max_{i \in \text{outcomes}} \frac{c(t)}{F(S) - F(S^i_{g_t})}$,   ▷ risk for feature $t$
13:        where $S^i_{g_t}$ is the set of examples in $S$ that has outcome $i$ using classifier $g_t$ with feature $t$.
14:    $\hat{t} \leftarrow \text{argmin}_t R(t)$
15:    $\hat{g} \leftarrow \text{argmin}_{g_{\hat{t}} \in \mathcal{G}_{\hat{t}}} \max_{i \in \text{outcomes}} \frac{c(\hat{t})}{F(S) - F(S^i_{g_{\hat{t}}})}$
16:    Make a node using feature $\hat{t}$ and classifier $\hat{g}$.
17:    for each outcome $i$ of $\hat{g}$ do
18:        GREEDYTREE($F, C, y^i_{\hat{g}}, X^i_{\hat{g}}$) to append as child nodes.
```

BUDGETRF iteratively builds decision trees by calling GREEDYTREE as a subroutine on a sampled subset of examples from the training data until the budget B is exceeded as evaluated using the validation data. The ensemble of trees are then returned as output. As shown in subroutine GREEDYTREE, the tree building process is greedy and recursive. If the given set of examples have zero impurity as measured by F, they are returned as a leaf node. Otherwise, compute the *risk* $R(t)$ for each feature t, which involves searching for a classifier g_t among the family of classifiers $\mathcal{G}_t$ that minimizes the maximum impurity among its outcomes. Intuitively, a feature with the least $R(t)$ can uniformly reduce the impurity among all its child nodes the most with the least cost. Therefore such a feature $\hat{t}$ is chosen along with the corresponding classifier $\hat{g}$. The set of examples are then partitioned using $\hat{g}$ to different child nodes at which GREEDYTREE is recursively applied. Note that we allow the algorithm to reuse the same feature for the same example in GREEDYTREE.

During Prediction: Given a test example and a decision forest $\mathcal{T}$ returned by BUDGETRF, we run the example through each tree in $\mathcal{T}$ and obtained a predicted label from each tree. The final predicted label is simply the majority vote among all the trees.

Different from random forest, we incorporate feature acquisition costs in the tree building subroutine GREEDYTREE with the hope of reducing costs while maintaining low classification error. Our main theoretical contribution is to propose a broad class of *admissible* impurity functions such that on any given set of n' examples the tree constructed by GREEDYTREE will have max-cost bounded by $O(\log n')$ times the optimal max-cost tree.

2.2. Bounding the Cost of Each Tree

Given a set of examples S with features and corresponding labels, a classification tree D has a feature-classifier pair associated with each internal node. A test example is routed from the root of D to a leaf node directed by the outcomes of the classifiers along the path; the test example is then labeled to be the majority class among training examples in the leaf node it reaches. The feature acquisition cost of an example $s \in S$ on D, denoted as $cost(D, s)$, is the sum of all feature costs incurred along the root-to-leaf path in D traced by s. Note that if s encounters a feature multiple times in the path, the feature cost contributes to $cost(D, s)$ only once because subsequent use of a feature already acquired for the test example incurs no additional cost. We define the total max-cost as

$$Cost(D) = \max_{s \in S} cost(D, s).$$

We aim to build a decision tree for any given set of examples such that the max-cost is minimized. Note that the max-cost criterion bounds the expected cost criterion of Eq. 3. While this bound could be loose we show later (see Sec. 2.4) that by parameterizing a suitable class of impurity functions, the max-costs of our GREEDYTREE solution can be "smoothened" so that it approaches the expected-cost.

First define the following terms: n' is the number of examples input to GREEDYTREE and m is the number of features, each of which has (a vector of) real values; F is the given impurity function; $F(S)$ is the impurity on the set of examples S; D_F is the family of decision trees with $F(L) = 0$ for any of its leaf L; each feature has a cost $c(t)$; a family of classifiers $\mathcal{G}_t$ is associated with feature t; $Cost_F(S)$ is the max-cost of the tree constructed by GREEDYTREE using impurity function F on S; and assume no feature is used more than once on the same example in the *optimal* decision tree among D_F that achieves the minimum max-cost, which we denote as $OPT(S)$ for the given input set of examples S. Note the assumption here is a natural one if the complexity of $\mathcal{G}_t$ is high enough. We show the $O(\log n')$ approximation holds for the max-cost of the optimal testing strategy using the GREEDYTREE subroutine if the impurity function F is admissible.

Definition A function F of a set of examples is *admissible* if it satisfies the following five properties: (1) Non-negativity: $F(G) \geq 0$ for any set of examples G; (2) Purity: $F(G) = 0$ if G consists of examples of the same class; (3) Monotonicity: $F(G) \geq F(R), \forall R \subseteq G$; (4) Supermodularty: $F(G \cup j) - F(G) \geq F(R \cup j) - F(R)$ for any $R \subseteq G$ and example $j \notin R$; (5) $\log(F(S)) = O(\log n')$.

Since the set S is always finite, by scaling F we can assume the smallest non-zero impurity of F is 1. Let τ and $\hat{g}_\tau$ be the first feature and classifier selected by GREEDYTREE at the root and let $S^i_{\hat{g}_\tau}$ be the set of examples in S that has outcome i using classifier $\hat{g}_\tau$. Note the optimization of classifier in Line (12) of Algorithm 1 needs not to be exact. We say GREEDYTREE is λ-*greedy* if $\hat{g}_\tau$ is chosen such that

$$\max_{i \in \text{outcomes}} \frac{c(\gamma)}{F(S) - F(S^i_{\hat{g}_\tau})} \leq \min_{g_t \in \mathcal{G}_t} \max_{i \in \text{outcomes}} \frac{\lambda c(t)}{F(S) - F(S^i_{g_t})},$$

for some constant $\lambda \geq 1$. By definition of max-cost,

$$\frac{Cost_F(S)}{OPT(S)} \leq \frac{c(\tau) + \max_i Cost_F(S^i_{\hat{g}_\tau})}{OPT(S)},$$

because feature τ could be selected multiple times by GREEDYTREE along a path and the feature cost $c(\tau)$ contributes only once to the cost of the path.

Let q be such that $Cost_F(S^q_{\hat{g}_\tau}) = \max_i Cost_F(S^i_{\hat{g}_\tau})$. We first provide a lemma to lower bound the optimal cost, which will later be used to prove a bound on the cost of the tree.

Lemma 2.1 *Let F be monotone and supermodular; let τ and $\hat{g}_\tau$ be the first feature and classifier chosen by* GREEDYTREE *λ-greedily on the set of examples S, assume no feature is used more than once on any path of the optimal tree, then*

$$c(\tau)F(S)/(F(S) - F(S^q_{\hat{g}_\tau})) \leq \lambda OPT(S).$$

Proof Let $D^* \in D_F$ be a tree with optimal max-cost. Let v be an arbitrarily chosen internal node in D^*, let γ be the feature associated with v and g^*_γ the corresponding classifier. Let $R \subseteq S$ be the set of examples associated with the leaves of the subtree rooted at v. Let i be such that $c(\tau)/(F(S) - F(S^i_{\hat{g}_\tau}))$ is maximized. Let $g^{min}_\gamma = \operatorname{argmin}_{g_\gamma \in \mathcal{G}_\gamma} \max_{i \in \text{outcomes}} \frac{c(\gamma)}{F(S) - F(S^i_{g_\gamma})}$. Let w be such that $c(\gamma)/(F(S) - F(S^w_{g^{min}_\gamma}))$ is maximized; similarly let j be such that $c(\gamma)/(F(S) - F(S^j_{g^*_\gamma}))$ is maximized. We then have:

$$\frac{c(\tau)}{F(S) - F(S^q_{\hat{g}_\tau})} \leq \frac{c(\tau)}{F(S) - F(S^i_{\hat{g}_\tau})} \leq \frac{\lambda c(\gamma)}{F(S) - F(S^w_{g^{min}_\gamma})}$$
$$\leq \frac{\lambda c(\gamma)}{F(S) - F(S^j_{g^*_\gamma})} \leq \frac{\lambda c(\gamma)}{F(R) - F(R^j_{g^*_\gamma})}. \quad (4)$$

The first inequality follows from the definition of i. The second inequality follows from the λ-greedy choice at the root. The third inequality follows from the minimization over classifiers given feature γ. To show the last inequality, we have to show $F(S) - F(S^j_{g^*_\gamma}) \geq F(R) - F(R^j_{g^*_\gamma})$. This follows from the fact that $S^j_{g^*_\gamma} \cup R \subseteq S$ and $R^j_{g^*_\gamma} = S^j_{g^*_\gamma} \cap R$ and therefore $F(S) \geq F(S^j_{g^*_\gamma} \cup R) \geq F(S^j_{g^*_\gamma}) + F(R) - F(R^j_{g^*_\gamma})$, where the first inequality follows from monotonicity and the second follows from the definition of supermodularity.

For a node v, let $S(v)$ be the set of examples associated with the leaves of the subtree rooted at v. Let $v_1, v_2, \ldots, v_p$ be a root-to-leaf path on D^* as follows: v_1 is the root of the tree, and for each $i = 1, \ldots, p-1$ the node v_{i+1} is a child of v_i associated with the branch of j that maximizes $c(t_i)/(F(S) - F(S^j_{g^*_{t_i}}))$, where t_i is the test associated with v_i. It follows from (4) that

$$\frac{[F(S(v_i)) - F(S(v_{i+1}))]c(\tau)}{\lambda(F(S) - F(S^q_{\hat{g}_\tau}))} \leq c_{t_i}. \quad (5)$$

Since the cost of the path from v_1 to v_p is no larger than the max cost of the D^*, we have that

$$OPT(S) \geq \sum_{i=1}^{p-1} c_{t_i}$$
$$\geq \frac{c(\tau)}{\lambda(F(S) - F(S^q_{\hat{g}_\tau}))} \sum_{i=1}^{p-1} (F(S(v_i)) - F(S(v_{i+1}))$$
$$= \frac{c(\tau)(F(S) - F(S(v_p))}{\lambda(F(S) - F(S^q_{\hat{g}_\tau}))} = \frac{c(\tau)F(S)}{\lambda(F(S) - F(S^q_{\hat{g}_\tau}))},$$

where the first inequality follows by the assumption that no feature is used more than once on any path of the optimal tree.

The main theorem of this section is the following.

Theorem 2.2 GREEDYTREE *constructs a decision tree achieving $O(\log n')$-factor approximation of the optimal max-cost in D_F on the set S of n' examples if F is admissible and no feature is used more than once on any path of the optimal tree.*

Proof This is an inductive proof:

$$\frac{Cost_F(S)}{OPT(S)} \leq \frac{c(\tau) + Cost_F(S^q_{\hat{g}_\tau})}{OPT(S)} \tag{6}$$

$$\leq \frac{c(\tau)}{OPT(S)} + \frac{Cost_F(S^q_{\hat{g}_\tau})}{OPT(S^q_{\hat{g}_\tau})} \tag{7}$$

$$\leq \lambda\frac{F(S) - F(S^q_{\hat{g}_\tau})}{F(S)} + \frac{Cost_F(S^q_{\hat{g}_\tau})}{OPT(S^q_{\hat{g}_\tau})} \tag{8}$$

$$\leq \lambda \log(\frac{F(S)}{F(S^q_{\hat{g}_\tau})}) + \lambda \log(F(S^q_{\hat{g}_\tau})) + 1 \tag{9}$$

$$= \lambda \log(F(S)) + 1 = O(\log(n')). \tag{10}$$

The inequality in (7) follows from the fact that $OPT(S) \geq OPT(S^q_{\hat{g}_\tau})$. (8) follows from Lemma 2.1. The first term in (9) follows from the inequality $\frac{x}{x+1} \leq \log(1+x)$ for $x > -1$ and the second term follows from the induction hypothesis that for each $G \subset S$, $Cost_F(G)/OPT(G) \leq \lambda \log(F(G)) + 1$. If $F(G) = 0$ for some set of examples G, we define $Cost_F(G)/OPT(G) = 1$.

We can verify the base case of the induction as follows. if $F(G) = 1$, which is the smallest non-zero impurity of F on subsets of examples S, we claim that the optimal decision tree chooses the feature with the smallest cost among those that can reduce the impurity function F:

$$OPT(G) = \min_{t|\exists g_t, \text{s.t. } F(G^i_{g_t})=0, \forall i \in \text{outcomes}} c(t).$$

Suppose otherwise, the optimal tree chooses first a feature t with a child node G' such that $F(G') = 1$ and later chooses another feature t' such that all the child nodes of G' by $g_{t'}$ has zero impurity, then t' could have been chosen in the first place to reduce all child nodes of G to zero impurity by supermodularity of F. On the other hand, $R(t) = \infty$ in GREEDYTREE for the features that cannot reduce impurity and $R(t) = c(t)$ for those features that can. So the algorithm would pick the feature among those that can reduce impurity and have the smallest cost. Thus, we have shown that $Cost_F(G)/OPT(G) = 1 \leq \lambda \log(F(G)) + 1$ for the base case.

2.3. Admissible Impurity Functions

A wide range of functions falls into the class of admissible impurity functions. We employ a particular function called threshold-Pairs in our paper defined as

$$F_\alpha(G) = \sum_{i \neq j} [[n^i_G - \alpha]_+ [n^j_G - \alpha]_+ - \alpha^2]_+, \tag{11}$$

where n^i_G denotes the number of objects in G that belong to class i, $[x]_+ = \max(x, 0)$ and α is a threshold parameter. We include the proof of the following lemma in the Supplementary Material.

Lemma 2.3 $F_\alpha(G)$ *is* admissible.

Neither entropy nor Gini index satisfies the notion of admissibility because they are not monotonic set functions, that is a subset of examples does not necessarily have a smaller entropy or Gini index compared to the entire set. Therefore traditional decision tree learning algorithms do not incorporate feature costs and have no guarantee on the max-cost as stated in our paper. We have studied more impurity functions that are admissible such as the polynomials and Powers family of functions. After conducting experiments on smaller datasets we noted that they do not offer significant advantage over the threshold-Pairs used in this paper. Please see Supplementary Material for more details.

2.4. Discussions of the Algorithm

Before concluding the BUDGETRF algorithm and its analysis, we discuss further various design issues as well as their implications.

Choice of threshold α. In subroutine GREEDYTREE, each tree is greedily built until a minimum leaf impurity is met, then added to the random forest. The threshold α can be used to trade-off between average tree depth and number of trees. A lower α results in deeper trees with higher classification power and acquisition cost. As a result, fewer trees are added to the random forest before the budget constraint is met. Conversely, a higher α yields shallower trees with poorer classification performance, however due to the low cost of each tree, many are added to the random forest before the budget constraint is met. As such, α can be viewed as a bias-variance trade-off. In practice, it is selected using validation dataset.

Minimax-splits. The splitting criterion in the subroutine GREEDYTREE is based on the worst case impurity among child nodes, we call such splits minimax-splits as opposed to expected-splits, which is based on the expected impurity among child nodes. Using minimax-splits, our theoretical guarantee is a bound on the max-cost of individual trees. Note such minimax-splits have been shown to lead to expected-cost bound as well in the setting of GBS

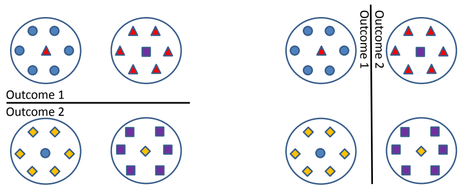


Figure 1. A synthetic example to show max-cost of GREEDYTREE can be "smoothened" to approach the expected-cost. The left and right figures above show the classifier outcomes of feature t_1 and t_2, respectively.

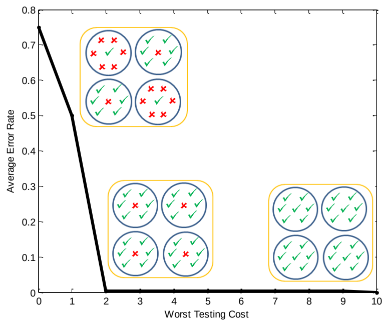


Figure 2. The error-cost trade-off plot of the subroutine GREEDYTREE using threshold-Pairs on the synthetic example. 0.39% error can be achieved using only a depth-2 tree but it takes a depth-10 tree to achieve zero error.

(Nowak, 2008); an interesting future research direction is to show whether minimax-splits can lead to a bound on the expected-cost of individual trees in our setting.

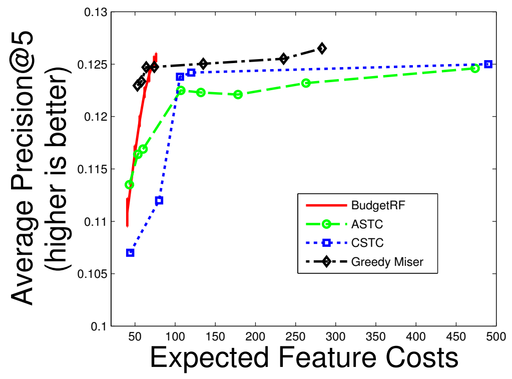
Smoothened Max-Costs. We emphasize that by adjusting α in threshold-Pairs function - essentially allowing some error, the max-costs of the GREEDYTREE solution can be "smoothened" so that it approaches the expected-cost. Consider the synthetic example as shown in Figure 1. Here we consider a multi-class classification example to demonstrate the effect of "smoothened" max-cost of the tree approaching the expected-cost. Consider a data set composed of 1024 examples belonging to 4 classes with 10 binary features available. Assume that is no two examples that have the same set of feature values. Note that by fixing the acquisition order of the features, the set of feature values maps each example to an integer in the range $[0, 1023]$. From this mapping, we give the examples in the ranges $[1, 255]$, $[257, 511]$, $[513, 767]$, and $[769, 1023]$ the labels 1, 2, 3, and 4, respectively, and the examples 0, 256, 512, and 768 the labels 2, 3, 4, and 1, respectively (Figure 1 shows the data projected to the first two features). Suppose each feature carries a unit cost. By Kraft's Inequality (Cover & Thomas, 1991), the optimal max-cost in order to correctly classify every object is 10, however, using only t_1 and t_2 as selected by the greedy algorithm, leads to a correct classification of all but 4 objects, as shown in Figure 2. Thus, the max-cost of the early stopped tree is only 2 - much closer to the expected-cost.

3. Experiments

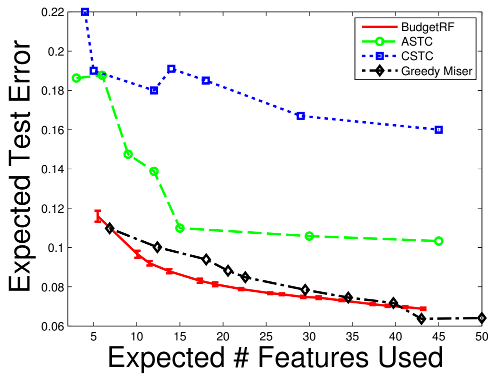
For establishing baseline comparisons we apply BUDGETRF on 4 real world benchmarked datasets. The first one has varying feature acquisition costs in terms of computation time and the purpose is to show our algorithm can achieve high accuracy during prediction while saving massive amount of feature acquisition time. The other 3 datasets do not have explicit feature costs; instead, we assign a unit cost to each feature uniformly. The purpose is to demonstrate our algorithm can achieve low test error using only a small fraction of features. Note our algorithm is adaptive, meaning it acquires different features for different examples during testing. So the feature costs in the plots should be understood as an average of costs for all test examples. We use GreedyMiser (Xu et al., 2012), CSTC (Xu et al., 2013) and ASTC (Kusner et al., 2014) for comparison because they have been shown to have state-of-the-art cost-error performance. For comparison purposes we use the same configuration of training/validation/test splits as in ASTC/CSTC. The algorithm parameters for ASTC are set using the same configuration as in (Kusner et al., 2014). We report values for CSTC from (Kusner et al., 2014). We use the code provided by the authors for GreedyMiser, tuning the learning rate and loss-cost trade-off parameter λ using grid search. In all our experiments we use the threshold-Pairs (11) as impurity function. We use stumps as the family of classifiers $\mathcal{G}_t$ for all features t. The optimization of classifiers in line 12 of Algorithm 1 is approximated by randomly generating 80, 40 and 20 stumps if the number of examples exceeds 2000, 500 and less than 500, respectively and select the best among them. All results from our algorithm were obtained by taking an average of 10 runs and standard deviations are reported using error bars.

Yahoo! Learning to Rank: (Chapelle et al.) We evaluate BUDGETRF on a real world budgeted learning problem: Yahoo! Learning to Rank Challenge [1]. The dataset consists of $473, 134$ web documents and $19, 944$ queries. Given a set of training query-document pairs together with relevance ranks of documents for each query, the Challenge is to learn an algorithm which takes a new query and its set of associated documents and outputs the rank of these documents with respect to the new query. Each example x_i contains 519 features of a query-document pair. Each of these features is associated with an acquisition cost in the set $\{1, 5, 20, 50, 100, 150, 200\}$, which represents the units of time required for extraction and is provided by a Yahoo! employee. The labels are binarized so that $y_i = 0$ means the document is unrelated to the query in x_i whereas $y_i = 1$ means the document is relevant to the query. There are $141, 397/146, 769/184, 968$ examples

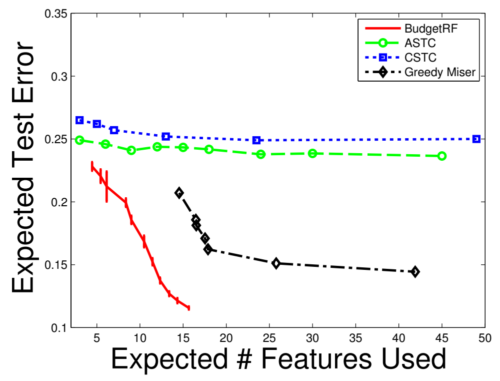
[1]http://webscope.sandbox.yahoo.com/catalog.php?datatype=c



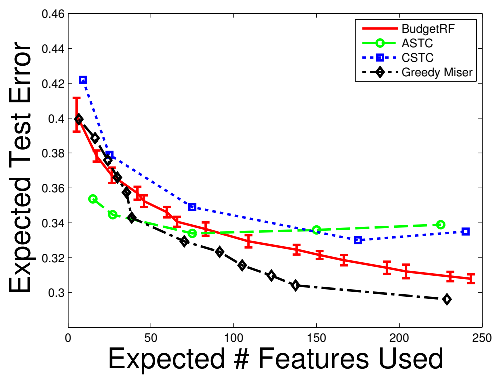
(a) Yahoo! Rank



(b) MiniBooNE



(c) Forest Covertype



(d) CIFAR-10

Figure 3. Comparison of BUDGETRF against ASTC (Kusner et al., 2014) and CSTC (Xu et al., 2013) on 4 real world datasets. BUDGETRF has a clear advantage over these state-of-the-art methods as it achieves high accuracy/low error using less feature costs.

in training/validation/test sets. We use the Average Precision@5 as performance metric, same as that used in (Kusner et al., 2014). To evaluate a predicted ranking for a test query, first sort the documents in decreasing order of the predicted ranks - that is, the more relevant documents predicted by the algorithm come before those that are deemed irrelevant. Take the top 5 documents in this order and reveal their true labels. If all of the documents are indeed relevant ($y = 1$), then the precision score is increased by 1; otherwise, if the first unrelated document appears in position $1 \leq j \leq 5$, increase the precision score by $\frac{j-1}{5}$. Finally, the precision score is averaged over the set of test queries. We run BUDGETRF using the threshold $\alpha = 0$ for the threshold-Pairs impurity function. To incorporate prediction confidence we simply run a given test example through the forest of trees to leaf nodes and aggregate the number of training examples at these leaf nodes for class 0 and 1 seperately. The ratio of class 1 examples over the sum of class 1 and 0 examples gives the confidence of relevance. The comparison is shown in plot (a) of Figure 3. The precision for BUDGETRF rises much faster than ASTC and CSTC. At an average feature cost of 70, BUDGETRF already exceeds the precision that ASTC/CSTC can achieve using feature cost of 450 and more. GreedyMiser has an initial precision higher than BUDGETRF but rises more slowly. In this experiment the maximum number of trees we build is 140; the precision is set to rise even higher if we were to use more trees. BUDGETRF thus represents a better ranking algorithm requiring much less wait time for users of the search engine.

MiniBooNE Particle Identification Data Set: (Frank & Asuncion) The MiniBooNE data set is a binary classification task, with the goal of distinguishing electron neutrinos (signal) from muon neutrinos (background). Each data point consists of 50 experimental particle identification variables (features). There are $45,523/19,510/65,031$ examples in training/validation/test sets. We apply BUDGETRF with a set of 10 values of $\alpha = [0,2,4,6,8,10,15,25,35,45]$. For each α we build a forest of maximum 40 trees using BUDGETRF. Each point on the BUDGETRF curve in (b) of Figure 3 corresponds to a α setting and the number of trees that meet the budget level. The final α is chosen using validation set. Our algorithm clearly achieves lower test error than both ASTC and CSTC on every point of the budget level. Indeed, using just about 6 features on average out of 50 , BUDGETRF achieves lower test error than what can be achieved by ASTC or CSTC using any number of features. GreedyMiser achieves similar performance with BudgetRF.

Forest Covertype Data Set: (Frank & Asuncion) The Forest data set contains cartographic variables to predict 7 forest cover types. Each example contains 54 (10 continuous and 44 binary) features. There are $36,603/15,688/58,101$ examples in training/validation/test sets. We use the same α values as in MiniBooNE. The final α is chosen using validation set. In (c) of Figure 3, ASTC and CSTC struggles to decrease test error even at high feature budget whereas the test error of BudgetRF decreases rapidly as more features are acquired. GreedyMiser again does better than ASTC/CSTC but uses much more features than BudgetRF for similar test error. We believe this dramatic performance difference is partly due to the distinct advantage of BudgetRF in handling mixed continuous and discrete (categorical) data where the optimal decision function is highly non-linear.

CIFAR-10: (Krizhevsky, 2009) CIFAR-10 data set consists of 32x32 colour images in 10 classes. 400 features for each image are extracted using technique described in (Coates & Ng, 2011). The data are binarized by combining the first 5 classes into one class and the others into the second class. There are $19,761/8,468/10,000$ examples in training/validation/test sets. As shown in (d) of Figure 3 BudgetRF and GreedyMiser initially have higher test error than ASTC when the budget is low; from a budget about 90 onward BudgetRF and GreedyMiser outperform ASTC while they outperform CSTC on the entire curve. An important trend we see is that the errors for both ASTC and CSTC start to increase after some budget level. This indicates an issue of overfitting with these methods. We do not see such an issue with BudgetRF and GreedyMiser.

As a general comment, we observe that in low-cost regions using higher α achieves lower test error whereas setting $\alpha = 0$ leads to low test error at a higher cost. This is consistent with our intuition that setting a high value for α terminates the tree building process early and thus saves on cost, as a consequence more trees can be built within the budget. But as budget increases, more and more trees are added to the forest, the prediction power does not grow as fast as setting α to low values because the individual trees are not as powerful.

Comments on standard Random Forest Cost is not incorporated in the standard random forest (RF) algorithm. One issue that arises is how to incorporate budget constraint. Our strategy was to limit the number of trees in the RF to control the cost. But this does not work well even if the acquisition costs are uniform for all features. We run Breiman's RF as implemented in Matlab's TreeBagger using default settings: fraction of input data to sample with replacement from the input data for growing each new tree is 1; number of features to select at random for each decision split is square root of the total number of features; minimum number of observations per tree leaf is 1. We report our findings in Table 1. Each entry in the table contains the cost as percentage of total number of features used, together with test error in parenthesis. All results are averaged over 10 runs. The test errors of the 3 methods are quite close in all 3 datasets with increasing number of trees. But BudgetRF using threshold-Pairs impurity with $\alpha = 0$ has significantly lower costs than RF(Gini) and RF(entropy) uniformly across all datasets. For example in Forest dataset, after building 40 trees, RF(Gini) uses 76.63% of total number of features for an average test example whereas BudgetRF uses only 29.01%. In terms of test error BudgetRF achieves 0.1156, slightly better than 0.1196 obtained by RF(Gini). For Yahoo! Rank dataset, RF does even worse because some features have very high cost and yet RF still uses them just like the less expensive features, resulting in high cost.

	Num. Trees	1	10	20	40
MiniB	RF(Gini)	27.32(0.1278)	85.06(0.0803)	94.83(0.0730)	98.42(0.0693)
	RF(entropy)	20.67(0.1230)	75.54(0.0775)	90.35(0.0713)	97.54(0.068)
	BudgetRF	16.4(0.1241)	57.8(0.0786)	73.57(0.0721)	86.78(0.0689)
Forest	RF(Gini)	22.57(0.2107)	63.04(0.1307)	70.76(0.1238)	76.63(0.1196)
	RF(entropy)	21.68(0.2045)	63.56(0.1313)	72.37(0.1239)	79.34(0.1200)
	BudgetRF	11.37(0.2122)	23.21(0.1364)	25.92(0.1232)	29.01(0.1156)
CIFAR	RF(Gini)	3.8(0.4284)	30.35(0.3604)	49.85(0.3349)	72.20(0.3106)
	RF(entropy)	3.46(0.4267)	28.34(0.3561)	46.45(0.3296)	68.47(0.3080)
	BudgetRF	2.62(0.4264)	21.09(0.3600)	35.45(0.3332)	54.24(0.3125)

Table 1. Percentage of average number of features used together with test error in parenthesis for different number of trees.

4. Conclusion and Future Work

We propose a novel algorithm to solve the budgeted learning problem. Our approach is to build a random forest of low cost trees with theoretical guarantees. We demonstrate that our algorithm performance is competitive to the state-of-the-art algorithms on 4 real world benchmarked datasets. While we have explored the greedy algorithm based on minimax-splits, similar algorithm can be proposed based on expected-splits. An interesting future work is to examine the theoretical and empirical properties of such algorithms.

Acknoledgement: This work is supported by NSF grant CNS-1330008.

References

Bellala, G, Bhavnani, S.K., and Scott, C. Group-based active query selection for rapid diagnosis in time-critical situations. *Information Theory, IEEE Transactions on*, Jan 2012.

Breiman, L. Random forests. *Machine Learning*, 45(1): 5–32, 2001.

Busa-Fekete, R., Benbouzid, D., and Kégl, B. Fast classification using sparse decision dags. In *Proceedings of the 29th International Conference on Machine Learning*, pp. 951–958, 2012.

Chapelle, O, Chang, Y, and Liu, T (eds.). *Proceedings of the Yahoo! Learning to Rank Challenge, held at ICML 2010, Haifa, Israel, June 25, 2010*.

Chen, M., Xu, Z., Weinberger, K. Q., Chapelle, O., and Kedem, D. Classifier cascade: Tradeoff between accuracy and feature evaluation cost. In *International Conference on Artificial Intelligence and Statistics*, 2012.

Cicalese, F, Laber, E. S, and Saettler, A. M. Diagnosis determination: decision trees optimizing simultaneously worst and expected testing cost. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China*, 2014.

Coates, A. and Ng, A. G. The importance of encoding versus training with sparse coding and vector quantization. In *Proceedings of the 28th International Conference on Machine Learning*. ACM, 2011.

Cover, T. M. and Thomas, J. A. *Elements of Information Theory*. Wiley-Interscience, New York, NY, USA, 1991.

Dulac-Arnold, G., Denoyer, L., Preux, P., and Gallinari, P. Datum-wise classification: a sequential approach to sparsity. In *Machine Learning and Knowledge Discovery in Databases*, pp. 375–390. 2011.

Frank, A. and Asuncion, A. UCI machine learning repository.

Gao, T. and Koller, D. Active classification based on value of classifier. In *Advances in Neural Information Processing Systems (NIPS 2011)*, 2011.

He, H, Daume III, H, and Eisner, J. Imitation learning by coaching. In *Advances In Neural Information Processing Systems*, 2012.

Ji, S and Carin, L. Cost-sensitive feature acquisition and classification. *Pattern Recognition*, 2007.

Kanani, P. and Melville, P. Prediction-time Active Feature-Value Acquisition for Cost-Effective Customer Targeting. In *Advances In Neural Information Processing Systems (NIPS)*, 2008.

Kapoor, A and Horvitz, E. Breaking boundaries: Active information acquisition across learning and diagnosis. In *Advances In Neural Information Processing Systems*, 2009.

Karayev, S, Fritz, M, and Darrell, T. Dynamic feature selection for classification on a budget. In *International Conference on Machine Learning (ICML): Workshop on Prediction with Sequential Models*, 2013.

Krizhevsky, Alex. Learning Multiple Layers of Features from Tiny Images. Master's thesis, 2009.

Kusner, M, Chen, W, Zhou, Q, Zhixiang, E, Weinberger, K, and Chen, Y. Feature-cost sensitive learning with submodular trees of classifiers. In *AAAI Conference on Artificial Intelligence*, 2014.

MacKay, D. J. C. Information-based objective functions for active data selection. *Neural computation*, 4(4):590–604, 1992.

Moshkov, M. J. Greedy algorithm with weights for decision tree construction. *Fundam. Inf.*, 104(3):285–292, August 2010.

Nowak, Robert. Generalized binary search. In *In Proceedings of the 46th Allerton Conference on Communications, Control, and Computing*, pp. 568–574, 2008.

Trapeznikov, K and Saligrama, V. Supervised sequential classification under budget constraints. In *International Conference on Artificial Intelligence and Statistics*, pp. 581–589, 2013.

Viola, P and Jones, M. Robust Real-time Object Detection. *International Journal of Computer Vision*, 4:34–47, 2001.

Wang, J., Bolukbasi, T., Trapeznikov, K, and Saligrama, V. Model selection by linear programming. In *European Conference on Computer Vision*, pp. 647–662, 2014a.

Wang, J, Trapeznikov, K, and Saligrama, V. An lp for sequential learning under budgets. In *International Conference on Artificial Intelligence and Statistics*, 2014b.

Xu, Z, Kusner, M, Chen, M, and Weinberger, K. Q. Cost-sensitive tree of classifiers. In *Proceedings of the 30th International Conference on Machine Learning*, 2013.

Xu, Zhixiang Eddie, Weinberger, Kilian Q., and Chapelle, Olivier. The greedy miser: Learning under test-time budgets. In *Proceedings of the 29th International Conference on Machine Learning, ICML*, 2012.

Zhang, C. and Zhang, Z.˙ A Survey of Recent Advances in Face Detection. Technical report, Microsoft Research, 2010.