Modeling Portfolio Risk by Risk Discriminatory Trees and Random Forests

Yang, Bill Huajian

1 August 2013

MODELING PORTFOLIO RISK BY RISK DISCRIMINATORY TREES AND RANDOM FORESTS[*]

(Pre-typeset version)

(Final version is published in "Journal of Risk Model Validation", vol.8 / No.1, 2014)

Bill Huajian Yang

Abstract

Common tree splitting strategies involve minimizing a criterion function for minimum impurity (i.e. difference) within child nodes. In this paper, we propose an approach based on maximizing a discriminatory criterion for maximum risk difference between child nodes. Maximum discriminatory separation based on risk is expected in credit risk scoring and rating. The search algorithm for an optimal split, proposed in this paper, is efficient and simple, just a scan through the dataset. Choices of different trees, with options either more or less aggressive in variable splitting, are made possible. Two special cases are shown to relate to the Kolmogorov Smirnov (KS) and the intra-cluster correlation (ICC) statistics. As a validation of the proposed approaches, we estimate the exposure at default for a commercial portfolio. Results show, the risk discriminatory trees, constructed and selected using the bagging and random forest, are robust. It is expected that the tools presented in this paper will add value to general portfolio risk modelling.

Key words: Exposure at default, probability of default, loss given default, discriminatory tree, CART tree, random forest, bagging,, KS statistic, intra-cluster correlation, penalty function, risk concordance

1. Introduction

Let $D = \{(z, y)\}$ denote a dataset, where $z = (x_1, x_2, ..., x_m)$ is a list of explanatory variables, and y a dependent variable. The dataset D can be a sample from the historical data of a portfolio, and y a risk target. The risk target y can be binary, ordinal discrete, or continuous, including the following cases:

(a) The exposure at default for a facility in 12 months
(b) The credit utilization for a facility in 12 months
(c) The default indicator for a borrower in 12 months
(d) The loss given default for a facility
(e) Active return (net risk free return) for an equity in a horizon

Risk target (e) applies to active portfolio management, while (a) and (b) are for exposure at default (EAD), (c) for probability of default (PD), and (d) for loss given default (LGD). Risk targets PD, EAD, and LGD are important for risk assessments and capital allocation decisions ([2], [3], [14]).

While the above risk targets can be estimated by a logit or a probit model using parametric approaches ([17], [19]), our focus in this paper is on a non-parametric approach by a type of risk discriminatory trees, using random forest ([7], [8], [12], [15]) and bagging (bootstrap aggregating) technique ([1], [4], [11], [12], [13]).

A tree-model construction starts at the training sample, i.e., the root node, by selecting an appropriate explanatory variable x and splitting it at an appropriate location $x = v$. The splitting results in two child bins (or child nodes): $\{x \le v\}$ and $\{x > v\}$. The process is repeated, each time at a node, and stops at an appropriate stage. Eventually, the training sample is partitioned into mutually exclusive sub-groups (i.e. tree leaves) in such a way that it is risk homogeneous within groups and heterogeneous between groups.

Finding an optimal location to split for a variable at a node is a key step for a tree-model construction. This can be achieved by minimizing a splitting criterion, such as the least squares or least deviation for

[*] Bill Huajian Yang, Ph. D in mathematics, Wholesale Credit Methodologies, Bank of Montreal. Mail address: 51st floor, First Canadian Place, 100 King Street West, Toronto, Canada, M5X 2A1

The views expressed in this article are not necessarily those of Bank of Montreal or any of its affiliates. Please direct any comment to bill.yang@bmo.com, phone 416-643-1922

CART regression trees ([8], [10]), and criteria based on various impurity functions for classification trees, including Gini index, entropy, and twoing rule ([6], [8], [10]).
Instead of minimizing a risk criterion function for minimum impurity within groups, we maximize a discriminatory criterion of the form:

$$p(v)\,(D(v))^{\alpha},\ \alpha > 0 \tag{1.1}$$

for a given α for maximum risk dissimilarity between child bins, where $p(v)$ acts as a penalty function to the size unbalance between child bins, and $D(v)$ is the distance in risk between child bins (see section 2).

One advantage of the binary splitting criteria (1.1) is the simplicity and efficiency of the search algorithm for an optimal split we propose in the next section. We call a binary regression tree constructed by maximizing criterion (1.1) for a given α a risk discriminatory tree. In risk management practice, maximum risk separation between groups is expected.

The parameter α plays an important role. As shown in the next section, the smaller the α, the higher the penalty to the size unbalance between two child bins. Different values of α define a family of distinct trees. Of particular interest are the cases when α has values 1 and 2. As shown in the next section, maximizing expression (1.1) for $\alpha = 1$ (resp. $\alpha = 2$), is equivalent to maximizing for the Kolmogorov Smirnov (KS) (resp. intra-cluster correlation (ICC)) statistic, and the resulting trees for $\alpha = 2$ are actually the CART binary regression trees given by the least squares splitting criterion.

To avoid over-fitting and increase performance stability, we impose two constraints for variable splitting: (a) minimum leaf size as a percentage of the total training sample size, (b) risk concordance. Constraint (b) is to ensure that each split complies with our general risk expectation for the variable (see section 4.2).

Advantages of the proposed approaches, based on discriminatory tree and random forest, include:

(1) Instead of minimizing a criterion for minimum impurity between child bins in variable splitting as usual, risk dissimilarity between child bins is maximized, as required in risk scoring and rating.
(2) Choices of different trees are made possible, with options to be either more or less aggressive than the CART tree in variable splitting (Proposition 2.1).
(3) Search for an optimal split is simple and efficient, requiring just a scan through the dataset (Algorithm 2.2).
(4) The risk of over-fitting for a tree model is minimized by constraints, in particular the risk concordance constraint (section 4.2).
(5) Performance stability is enhanced by selecting the champion tree from a random forest (section 5)

A CART binary regression tree, at the earlier stage of a tree-model construction, can split so aggressively that no further splits at a node are possible due to the dry up of records (see section 5.3 for an example). In these cases, a less aggressive discriminatory tree with $0 < \alpha < 2$ can be chosen as an alternative.

The paper is organized as follows: Risk discriminatory trees are proposed in section 2. In section 3, we focus on the discriminatory trees for two special cases: $\alpha = 1, 2$. Implementation considerations are included in section 4. In section 5, we validate the proposed tree models by estimating the exposure at default for a commercial portfolio, where we generate two random forests for cases $\alpha = 1, 2$ using the bagging technique, select the champion tree from each forest, and compare the performance to logit models.

The author thanks the referees and Clovis Sukam for many valuable comments and conversations.

2. Risk Discriminatory Trees

2.1 Splitting Criteria

Given a training sample *D* with a risk target *y*, we assume $0 \le y \le 1$. For a numerical variable *x* and a split at $x = v$, let $\bar{y}_L(v)$ denote the average of *y* values on the left child bin $B_L(v) = \{x \le v\}$, and $\bar{y}_R(v)$ the average of *y* values on the right child bin $B_R(v) = \{x > v\}$. Let n_1 be the size of $B_L(v)$, n_2 the size of $B_R(v)$, and $N(= n_1 + n_2)$ the size of the training sample. Define $p(v) = 4n_1n_2 / N^2$ and $D(v) = | \bar{y}_L(v) - \bar{y}_R(v) |$. The discriminatory splitting criterion $BT(\alpha, v)$ is defined as:

$$BT(\alpha, v) = p(v)\,(D(v))^{\alpha} \qquad (2.1)$$

For a given $\alpha > 0$, a binary discriminatory tree can be constructed by selecting a splitting variable and splitting it at the best location where the criterion reaches its maximum, level by level and node by node, under possibly a set of pre-specified constraints χ (e.g., minimum node size, risk concordance).

We say the bin size is more balanced or the size distribution is more even, if $| (n_1 - n_2) / N |$ is smaller. Bin size balance is described by the function $p(v) = 4n_1n_2 / N^2$, which serves as a penalty function in (2.1). We observe that the function $f(x) = 4x(1-x),\ \ 0 \le x \le 1$, reaches its maximum value of 1at $x = 1/2$, and is increasing for $0 \le x \le 1/2$, while decreasing for $1/2 \le x \le 1$. Since $p(v)$ has the form of $4(n_1 / N)(1 - n_1 / N)$, the value of $p(v)$ is higher when bin size is more balanced.

Notice that $0 \le BT(\alpha, v) \le 1$. Maximizing (2.1) for an optimal split is the same as minimizing:

$$-\log[p(v)\,(D(v))^{\alpha}] = -\log(p(v)) - \alpha \log(D(v)) \qquad (2.2)$$

For a given $\alpha > 0$, the first term $-\log(p(v))$ in (2.2) is smaller when $p(v)$ is higher. Thus the minimization of (2.2) favours the split with more bin size balanced, given the value of $D(v)$.

On the other hand, minimizing (2.2) is the same as minimizing:

$$-(1/\alpha)\log(p(v)) - \log(D(v)) \qquad (2.3)$$

The first term $-(1/\alpha)\log(p(v))$ in (2.3) is higher for smaller α, thus the penalty for bin size unbalance is higher for smaller α.

The following proposition summarizes the above discussions on the roles of the penalty function $p(v)$ and parameter α.

Proposition 2.1. The following statements hold:

(a) For a given $\alpha > 0$, the bin size for a binary split at $x = v$ is more balanced when $p(v)$ is higher.
(b) The maximization of (2.1) favours the split with more bin size balanced, given the values of $D(v)$ and α.
(c) A binary risk discriminatory tree given by a smaller α penalizes the bin size unbalance more than the one given by a higher α.

Consequently, leaf size distribution for a risk discriminatory tree given by a smaller α (e.g., $\alpha = 1$) is more balanced than the one given by a higher α (e.g., $\alpha = 2$).

One advantage for the binary splitting criteria (2.1) is the simplicity and efficiency of the search algorithm for an optimal split as given below:

Algorithm 2.2. Given $\alpha > 0$ and a set of constraints χ, let x be a numerical explanatory variable. Find an optimal split following the steps below:

a) For each distinct value of x, sum up its corresponding y values, count the number of corresponding records, as in the SQL query below:

Create table *T* as select *x*, sum(y) as *sumy*, count (*) as *cntx* from *D* group by *x* order by *x*.

Consequently, we have a table *T* with only one record for each distinct value of *x*, sorted by values of *x* ascending. It has two columns in addition to column *x*:

sumy=Sum of all *y* values for a distinct value of *x*,
cntx=Number of records for a distinct value of *x*.

b) Initially, the cursor (i.e., the pointer) points to the 1st record of table *T*. Set

$$n_1 = cntx,\ n_2 = N - cntx,$$
$$Cum\ Sum\ y = sumy,$$
$$\bar{y}_L(v) = (Cum\ Sum\ y)/n_1,$$
$$\bar{y}_R(v) = (Tot\ Sum\ y - Cum\ Sum\ y)/n_2$$

Check for constraints χ. If all constraints are satisfied, then calculate $BT(\alpha, v)$ by (2.1), and store the current values of $BT(\alpha, v)$ and x as:

$$M = BT(\alpha, v),\ U = x$$

c) The cursor moves to the next record. Increment n_1 by *cntx,* decrement n_2 by *cntx,* and increment *Cum Sum y* by adding an amount of *sumy*. Update $\bar{y}_L(v)$ and $\bar{y}_R(v)$ as:

$$\bar{y}_L(v) = (Cum\ Sum\ y)/n_1,$$
$$\bar{y}_R(v) = (Tot\ Sum\ y - Cum\ Sum\ y)/n_2$$

where *Tot Sum y* denotes the total sum of *y* values over *D*. Check for constraints χ. If all constraints are satisfied, then calculate $BT(\alpha, v)$ by (2.1). If $BT(\alpha, v) > M$ then update

$$M = BT(\alpha, v),\ U = x$$

Repeating step (c) to the last record of the table *T*, one either gets the constrained maximum at $x = U$ or no location is found to be feasible under constraints χ.

For a class variable, re-group the levels when necessary, calculate the average of *y* values over each level (override this average based on business experience when necessary), and transform the variable to a numerical form by mapping each level to its corresponding *y* average. We use the transformed numerical version of a class variable.

During the course of a tree-model construction, we assume that splits are applied sequentially. When a split is applied at a node, two child nodes are generated, and we have a tree after all previous and the current splits (before any later splits) are applied. As usual for a regression tree, we predict *y* in a leaf by the average of *y* values over the leaf. We call this tree model over the training sample a tree at current.

The quality of a split can be measured by statistics RSQ, mean absolute deviation (MAD), Gini, and KSD calculated for the prediction of the tree at current.

Multinomial trees can be constructed by heuristically and recursively calling Algorithm 2.2. For example, to grow a trinomial tree, do the following at each node:

(i) Select an explanatory variable and call the algorithm for an initial split

(ii) For each child bin from step (i), either use the previous splitting variable or select another one, and call the algorithm for an optimal split

(iii) Select among the two splits in step (ii) the best one, measured by accuracy and performance for the tree at current where the split is applied (after applying the split given in (i)) , as the second split

It is worth mentioning that heuristically and recursively calling the algorithm for multiple splits under a node for the same splitting variable does not guarantee the monotone risk trend for consecutive child bins. For example, for two splits we have three consecutive child bins: B_1, B_2, B_3. Then the corresponding y averages $\overline{y}_1, \overline{y}_2, \overline{y}_3$ are not necessarily monotonic. Thus risk concordance is not necessarily satisfied for a multinomial tree, unlike the binary case.

2.2. Variable Selection

Selection of a splitting variable under a node can be based on either business or statistics. With statistical approaches, one can first rank all explanatory variables (before any split, with priority in sequence) by RSQ, Gini, and KSD under the node, then select either the top one, or randomly select from the top *k* ones (e.g., *k*=3 for our random forest construction in section 5). With the efficiency of Algorithm 2.2, we can actually split all candidate variables by calling the algorithm sequentially, and assess the quality of each split by the performance statistics for the tree at current where the split is actually applied. Then select a variable from the ranked list. This is what we do in our implementation (section 5).

3. Risk Discriminatory Tress for Cases $\alpha = 1, 2$

3.1. KS Binary Risk Discriminatory Trees $(\alpha = 1)$

The KS statistic is widely used in credit scoring and risk ranking to measure the ability of a variable in ranking risk. It is defined originally for good and bad binary outcomes. However, it can be generalized to cases where the values of a risk target *y* are in interval [0, 1].

Given a numerical variable *x* and a value $x = v$, let $bcdist(v)$ and $\text{gcd}\,ist(v)$ denote respectively the cumulative distributions for variables *y* and (1-*y*) up to $x \le v$:

$$bcdist(v) = \frac{cumulative\ sum\ of\ y\ values\ \ up\ to\ x \le v}{total\ sum\ of\ y},$$

$$\text{gcd}\,ist(v) = \frac{cumulative\ sum\ of\ (1-y)\ values\ up\ to\ x \le v}{total\ sum\ of\ (1-y)},$$

Let $d\,(v) = |\,bcdist - \text{gcd}\,ist\,|$. The *KS* statistic is defined to be the maximum of $d(v)$:

$$KSD = \max_v\ d(v) = \max_v\ |\,bcdist(v) - \text{gcd}\,ist(v)\,|$$

Clearly, we have $0 \le KSD \le 1$. The higher the *KSD* value, the better the variable *x* in separating higher *y* values from the lower *y* values.

Let

Tot Sum y = the total sum of *y* values over the entire sample.
Tot Sum (1-*y*) = the total sum of (1-*y*) values over the entire sample.
Cum Sum y = the cumulative sum of *y* values over the left bin $\{y\ |x \le v\}$.
Cum Sum (1-*y*) = the cumulative sum of (1-y) values over the left bin $\{y\ |x \le v\}$.

The proposition below (see Appendix for a proof) shows the equivalence between maximizing the criterion (2.1) for $\alpha = 1$ and maximizing $d(v)$. For this reason, we call a binary risk discriminatory tree given by $\alpha = 1$ a KS risk discriminatory tree.

Proposition 3.1. There holds $d(v) = cp(v)D(v)$, where $c > 0$ is a constant given by:

$$c = N^2 /(4 \times Tot\,Sum\; y \times Tot\,Sum\,(1-y))$$

3.2. ICC Binary Risk Discriminatory Trees $(\alpha = 2)$

Given a sample, assume that the values of the risk target *y* divide into two groups:

$$\{ y_{ij} \mid i = 1,\ 2;\ j = 1,\ 2, \dots,\ n_i \}$$

Let $\bar{y}_1$, $\bar{y}_2$, and $\bar{y}$ denote respectively the averages of *y* values over groups 1, 2, and the entire sample. Denote the total sum-of-squares by SST, the sum-of-squares within groups by SSW, and the sum-of-squares between groups by SSB, i.e.:

$$SST = \sum_{i=1}^{2} \sum_{j=1}^{n_i} (y_{ij} - \bar{y})^2, \quad SSW = \sum_{j=1}^{n_1} (y_{1j} - \bar{y}_1)^2 + \sum_{j=1}^{n_2} (y_{2j} - \bar{y}_2)^2 \quad ,$$

$$SSB = n_1(\bar{y}_1 - \bar{y})^2 + n_2(\bar{y}_2 - \bar{y})^2 .$$

Clearly, $SST = SSW + SSB$. The intra-class correlation (*ICC*) for this grouped data is then defined by

$$ICC = \frac{SSB}{SST} = \frac{n_1(\bar{y}_1 - \bar{y})^2 + n_2(\bar{y}_2 - \bar{y})^2}{\sum_{i=1}^{2} \sum_{j=1}^{n_i} (y_{ij} - \bar{y})^2}$$

Now, given a value $x = v$, the *y* values split into two groups: the left bin $\{ y \mid x \le v \}$, and the right bin $\{ y \mid x > v \}$. Thus the corresponding *ICC* statistic, denoted by $ICC(v)$, is defined. The following lemma shows the equivalence between maximizing criterion function $ICC(v)$ and minimizing the least squares criterion.

Lemma 3.2. Maximizing function $ICC(v)$ is equivalent to minimizing *SSW*.
Proof. This follows from the fact that *SST* is a constant for a given a sample *D* and $SST = SSW + SSB$ □

The next proposition shows the equivalence between maximizing functions $ICC(v)$ and $BT(\alpha, v)$ for $\alpha = 2$ (see Appendix for a proof).

Proposition 3.3. $ICC(v) = cp(v)\ (D(v))^2$, where $c = N/(4 \times SST)$ is a constant.

We call a binary risk discriminatory tree given by $\alpha = 2$ an ICC tree. An ICC tree is actually a CART binary regression tree given by the least squares splitting criterion by Lemma 3.2.

4. Implementation Considerations
4.1. Treatment of Missing and Special Values

For each explanatory variable, isolate missing and special values each as a separate group. For each group, calculate the average of the corresponding *y* values (override this average based on business experience when necessary), and do either of the following:

(a) Impute a group by a regular value of the explanatory variable that closely represents the risk of the group (measured by the average of *y* values over the group).
(b) Split the explanatory variable without these groups, then place each group back to a child bin based on the similarity of risk.

4.2. Constraints and Stopping Criteria

Stop and perform no split at a node if a constraint is violated.

Minimum leaf size - Set a minimum threshold value for leaf size by the record count of the node as a percentage of the total training sample size.

Global risk concordance -This constraint is intended to reduce the risk of over-fitting, thus increases the performance stability.

More specifically, we calculate the sign of the global rank correlation ρ between the variable *x* and the risk target *y* over the original training sample *D*. At each subsequent split, the values $\bar{y}_L(v)$ and $\bar{y}_R(v)$ are required to satisfy the following:

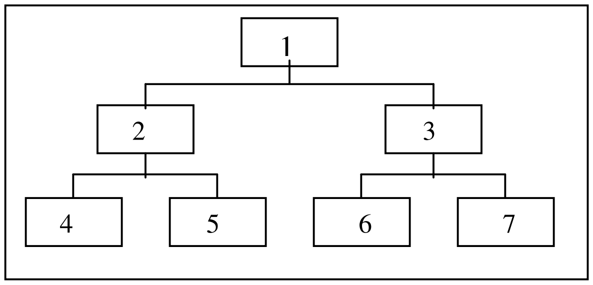
$$\bar{y}_L(v) < \bar{y}_R(v) \text{ if } \rho > 0, \text{ or } \bar{y}_L(v) > \bar{y}_R(v) \text{ if } \rho < 0$$

That is, $\bar{y}_L(v)$ and $\bar{y}_R(v)$ must be concordant with the global risk trend of the variable *x* over the training sample *D*. This is to ensure that each split complies with our general risk expectation for the variable

Maximum level of a tree – Specify the maximum level of the tree, stop if the level of a node reaches this maximum number.

4.3. Heap Tree Structure

The heap structure determines at any time whether a node m is a parent of another node n. This is useful when programming in a global environment. The root node is given the index 1 at level 1. The heap diagram for a 3-level binary tree looks as below:



Lemma 4.1. ([18], pp349-350) The node m is a parent of node n if and only if $n/2^i = m$ for some integer $i > 0$ under the integer division (i.e., the fraction part is discarded).

5. An Empirical Example -Modeling Exposure at Default by Risk Discriminatory Trees

5.1. EAD Target and Transformations

We are to estimate the exposure at default for a facility in a commercial portfolio using risk discriminatory tree. The sample contains the explanatory variables at observation for 1198 facilities that defaulted in 12 months after observation. Facility outstanding amount at the time of default is also given. We impute missing for an explanatory variable by a value that closely represents the risk for the missing group. We label this sample as *D*.

Let *util_fac_d_1* denote the facility utilization at observation, which is calculated as the ratio of facility outstanding amount to facility authorized amount at observation. Let *util_fac_d* be the ratio of facility outstanding amount at default to facility authorized amount at observation. Both variables are capped at 1 and floored at 0.

We choose *util_fac_d* as our EAD modeling target, i.e., we are to estimate the ratio *util_fac_d*, the exposure at default as a percentage of facility authorized amount at observation.

Define

$$\tilde{y} = util\_fac\_d - \beta\ util\_fac\_d\_1 \qquad (5.1)$$

$$y = (\tilde{y} + 1)/2 \qquad (5.2)$$

where β is a parameter to remove the lag-one autocorrelation effect from *util_fac_d* ([16]). The quantity $\tilde{y}$ can be regarded as an additional (to $\beta\ util\_fac\_d\_1$) drawdown, as a percentage of current authorized amount, given current outstanding. With transformation (5.2), we have a transformed risk target *y* that has values in interval [0, 1].

Variable *y* is our tree modeling target. If *y* is estimated, *util_fac_d* can be derived by inverting (5.1) and (5.2). Basel II requires that the final estimate for *util_fac_d* be floored at *util_fac_d_1* ([2, p.94]).

5.2. Random Forests

We generate two random forests following Breiman's algorithm ([7], [8], [12]) and the bagging technique ([1], [4], [11], [12], [13]):

1. Generate a bootstrap sample of the same size as the original sample *D*, and randomly split it into training and validation by 50:50
2. Grow two four-level (level 1 for the root level) binary discriminatory trees over the training sample, one for KS tree and another for ICC (i.e. CART) tree, each follows the steps:
    (a) For each candidate variable, call Algorithm 2.2 to search for an optimal split, constrained by minimum node size of 3% of the total training sample size, and risk concordance. Calculate the RSQ, MAD, Gini, and KSD for the tree at current where the split is applied.
    (b) Rank all candidate variables by RSQ, MAD, Gini, and KSD. Randomly select one from the top 3 variables, and apply the split found in step (a).
    (c) Fully grow the tree to maximum 4 levels with no pruning
3. Predict *y* in a leaf, as usual for a regression tree, by the average of *y* values over the leaf, and estimate *util_fac_d* by inverting (5.1) and (5.2), then flooring at *util_fac_d_1*. Calculate the RSQ, KSD, Gini, and MAD for the final prediction (for *util_fac_d*) over the samples: training (labelled as "Train" in Tables 1 and 2 below), validation ("Val"), and the combined of two ("All").

We iterate the above algorithm 20 times. Two random forests, each with 20 trees, are generated. Performance statistics are shown in Tables 1 and 2 below. One can compare a KS tree at iteration to the ICC tree at the same iteration, as both are trained on the same sample. Performance is measured with priority in sequence by RSQ, MAD, KSD, and Gini and by samples in sequence: (a) Overall, (b) Validation, (c) Training. Overall, iteration by iteration, performance is fairly close between KS and ICC trees, except for iteration 5, where the ICC tree is a winner, and iteration16, where :the KS tree is a winner.

Table1. The KS Random Forest

	RSQ			KSD			Gini			MAD		
Iter	Train	Val	All	Train	Val	All	Train	Val	All	Train	Val	All
1	0.54	0.51	0.52	0.45	0.46	0.46	0.59	0.59	0.59	0.10	0.11	0.11
2	0.55	0.56	0.55	0.43	0.43	0.43	0.60	0.60	0.60	0.10	0.10	0.10
3	0.54	0.52	0.53	0.45	0.45	0.45	0.59	0.59	0.59	0.11	0.10	0.10
4	0.50	0.51	0.49	0.45	0.44	0.45	0.59	0.58	0.58	0.11	0.11	0.10
5	0.56	0.54	0.54	0.45	0.44	0.45	0.60	0.58	0.60	0.10	0.10	0.10
6	0.53	0.51	0.52	0.44	0.44	0.44	0.59	0.58	0.59	0.10	0.11	0.10
7	0.52	0.54	0.53	0.45	0.46	0.45	0.58	0.61	0.59	0.10	0.10	0.11
8	0.58	0.59	0.55	0.47	0.48	0.46	0.61	0.63	0.60	0.10	0.10	0.10
9	0.54	0.54	0.54	0.46	0.44	0.46	0.59	0.58	0.59	0.11	0.10	0.10
10	0.53	0.49	0.52	0.44	0.43	0.44	0.59	0.58	0.59	0.10	0.11	0.11
11	0.53	0.52	0.54	0.44	0.42	0.44	0.58	0.58	0.59	0.10	0.11	0.10
12	0.50	0.52	0.51	0.44	0.46	0.45	0.58	0.60	0.59	0.10	0.10	0.10
13	0.54	0.52	0.52	0.46	0.44	0.45	0.60	0.58	0.59	0.10	0.11	0.10
14	0.54	0.51	0.53	0.45	0.46	0.46	0.59	0.58	0.59	0.10	0.11	0.11
15	0.52	0.51	0.51	0.46	0.46	0.45	0.59	0.60	0.59	0.10	0.11	0.11
16	0.58	0.55	0.56	0.47	0.44	0.45	0.61	0.58	0.59	0.10	0.10	0.10
17	0.52	0.51	0.52	0.45	0.44	0.45	0.59	0.59	0.59	0.10	0.11	0.10
18	0.55	0.50	0.53	0.46	0.45	0.46	0.60	0.58	0.59	0.10	0.11	0.10
19	0.50	0.53	0.51	0.45	0.47	0.46	0.58	0.59	0.59	0.11	0.10	0.11
20	0.57	0.51	0.54	0.46	0.44	0.45	0.61	0.58	0.60	0.09	0.10	0.10

Table 2. The ICC (CART) Random Forest

	RSQ			KSD			Gini			MAD		
Iter	Train	Val	All	Train	Val	All	Train	Val	All	Train	Val	All
1	0.55	0.54	0.54	0.44	0.45	0.44	0.59	0.60	0.59	0.10	0.10	0.10
2	0.54	0.55	0.54	0.43	0.44	0.43	0.60	0.60	0.60	0.10	0.10	0.10
3	0.53	0.51	0.51	0.45	0.45	0.45	0.58	0.59	0.58	0.11	0.11	0.10
4	0.50	0.50	0.50	0.47	0.45	0.46	0.59	0.58	0.59	0.10	0.10	0.10
5	0.58	0.57	0.57	0.47	0.47	0.47	0.62	0.61	0.61	0.10	0.09	0.10
6	0.51	0.51	0.51	0.44	0.43	0.44	0.59	0.58	0.59	0.10	0.10	0.10
7	0.53	0.55	0.54	0.44	0.45	0.44	0.58	0.60	0.59	0.10	0.11	0.10
8	0.57	0.52	0.54	0.45	0.42	0.44	0.61	0.57	0.59	0.10	0.10	0.10
9	0.53	0.52	0.53	0.46	0.45	0.45	0.60	0.59	0.59	0.10	0.10	0.10
10	0.54	0.51	0.53	0.46	0.45	0.46	0.59	0.58	0.59	0.10	0.11	0.10
11	0.53	0.52	0.54	0.45	0.43	0.45	0.58	0.59	0.60	0.10	0.10	0.10
12	0.50	0.51	0.52	0.44	0.45	0.45	0.58	0.59	0.59	0.10	0.10	0.10
13	0.55	0.52	0.53	0.44	0.44	0.44	0.60	0.59	0.59	0.10	0.10	0.10
14	0.54	0.53	0.55	0.46	0.45	0.46	0.60	0.59	0.60	0.10	0.11	0.10
15	0.52	0.50	0.51	0.45	0.44	0.45	0.59	0.58	0.59	0.10	0.11	0.11
16	0.57	0.53	0.55	0.45	0.43	0.44	0.61	0.58	0.59	0.10	0.10	0.10
17	0.53	0.53	0.53	0.44	0.44	0.44	0.59	0.59	0.59	0.10	0.10	0.10
18	0.57	0.53	0.56	0.48	0.45	0.47	0.61	0.59	0.61	0.09	0.10	0.10
19	0.51	0.53	0.52	0.46	0.47	0.47	0.58	0.59	0.59	0.11	0.10	0.11
20	0.56	0.50	0.54	0.45	0.42	0.44	0.60	0.57	0.59	0.10	0.10	0.10

5.3. Model Selection

Instead of predicting the tree target *y* by averaging the output from all trees in a forest, as usual for the random forest technique, we choose to predict *y* by the champion tree from a forest, i.e., the KS tree at iteration 16, and ICC tree at iteration 5. One or more challenger trees from each forest can be chosen for validation comparison when necessary.

Interestingly, both champion trees use the same five splitting variables:

1. Fac_auth_d_1 – Facility authorized amount at observation
2. Primary_coll_typ_lv_pcnt – Lending value as a percentage of primary collateral amount to facility authorized amount
3. Obl_bal_d_1 – Borrower's total outstanding amount at observation
4. Obl_util_d_1 – Borrower's total utilization at observation
5. Total Assets – Borrower total assets at observation

Detailed splitting at each node is shown in Tables 3 and 4 below, where the left child node for each split is given by the corresponding inequality in the 1st column of the tables.

Based on our business risk experience, we expect and require following risk concordance at each node: the tree target y, i.e., the transformed exposure risk after netting out the effect due to *util_fac_d_1*, is to be negatively correlated with: (a) facility authorised amount (the denominator in *util_fac_d*), (b) total assets, (c) obligor outstanding at observation; but positively correlated with: (d) lending value, (e) obligor total utilization.

To justify the negative correlation (c), we note that with the netting of (5.1) the risk target *y* can be regarded as a form of additional (to β *util_fac_d_1*) drawdown in horizon given current outstanding. Additional drawdown in horizon is in general negatively correlated with current outstanding.

To help to visually check for risk concordance, we have labelled the higher risk child bin as the left node under the heap structure. For example, the global rank correlation with *y* for the variable *Obl_util_d_1* is positive, so the child bin $\{x > v\}$ is the labelled as the left node.

Table 3. The Selected KS Tree (5 variables)

Variable/Split	At Node	Leaf Node	Pcnt
Fac_auth_d_1<= $600000	1	8	12.1%
Primary_coll_typ_lv_pcnt> 77.2%	2	9	29.2%
Primary_coll_typ_lv_pcnt> 83.39%	3	10	12.3%
Total_assets<= $244000	4	11	3.1%
Obl_bal_d_1<= $549054	5	12	22.7%
Total_assets<= $10115000	6	13	6.1%
Obl_util_d_1> 55.38%	7	14	7.6%
		15	7.1%
			100%

Table 4. The Selected ICC (CART) Tree (5 variables)

Variable/Split	At Node	Leaf Node	Pcnt
Fac_auth_d_1<= $635081	1	4	3.3%
Obl_bal_d_1<= $7850	2	10	38.8%
Total_assets<= $10115000	3	11	17.0%
No further split	4	12	18.9%
Primary_coll_typ_lv_pcnt> 83.7%	5	13	12.7%
Primary_coll_typ_lv_pcnt> 87.3%	6	14	4.4%
Obl_util_d_1>69.04%	7	15	5.1%

As pointed out in section 2, a KS tree with $\alpha = 1$ penalizes bin size unbalance more than an ICC (CART) tree with $\alpha = 2$. This is reflected in the leaf size distribution (column "Pcnt") in Tables 3 and 4, where the leaf size distribution for the selected KS tree is relatively more even than the ICC tree. More importantly, the ICC tree stops splitting at node 4 (level 3 in the heap structure) due to the node size constraint.

5.4. Model Validation

Performance stability is a key measure for robustness for a tree model. In actual implementation, we recommend one or more challenger trees to be chosen from a random forest along with the champion tree, for further comparison and validation.

We are more interested in performance comparison between the selected tree models and logit models. We thus train a logit model targeting the same variable *y* as in (5.2) using the same five drivers ([19]). The predicted value of *y* is then converted to predict *util_fac_d* using (5.1) and (5.2), and floored at *util_fac_d_1*.

We bootstrap the performance for the two tree models and the logit model as below:

1) Generate a bootstrap sample of the same size as the original sample *D*
2) Assess the performance for each of the two tree models and the logit model over the bootstrap sample
3) Repeat 1) and 2) by 1000 times.

Simulation results are shown in Tables 5-7 below. Both tree models are robust with stable performance. The selected ICC tree performs slightly better than the selected KS tree (due to model selection), with slightly higher RSQ, KSD, and Gini, and slightly lower MAD. For the logit model, the rank order statistics Gini and KSD are close to the tree models, but RSQ is lower, due to the higher variance of the model residuals.

Table 5. Performance for the Selected KS Tree

					Quantile	
	Mean	Std Dev	Min	Max	Low 5%	High 95%
RSQ	0.56	0.012	0.53	0.59	0.54	0.58
KSD	0.45	0.007	0.44	0.47	0.44	0.46
Gini	0.59	0.008	0.57	0.61	0.58	0.61
MAD	0.10	0.002	0.096	0.106	0.098	0.104

Table 6. Performance for the Selected ICC (CART) Tree

					Quantile	
	Mean	Std Dev	Min	Max	Low 5%	High 95%
RSQ	0.58	0.012	0.55	0.61	0.56	0.60
KSD	0.47	0.006	0.46	0.49	0.46	0.48
Gini	0.61	0.007	0.59	0.63	0.60	0.63
MAD	0.09	0.002	0.089	0.098	0.091	0.096

Table 7. Performance for the Logit Model

					Quantile	
	Mean	Std Dev	Min	Max	Low 5%	High 95%
RSQ	0.45	0.015	0.39	0.49	0.42	0.47
KSD	0.41	0.008	0.39	0.43	0.40	0.43
Gini	0.55	0.009	0.52	0.57	0.53	0.56
MAD	0.11	0.002	0.104	0.114	0.105	0.112

Conclusion. Risk discriminatory trees proposed in this paper provide a non-parametric approach for modeling of portfolio risk, which are simple, efficient, and easy to implement. Choices for different trees including KS and ICC (i.e. CART) trees, with options either more or less aggressive in variable splitting, are made possible The empirical example shows, the risk discriminatory trees, constructed and selected using the bagging and random forest technique, are robust, and can be an efficient tool besides the parametric approaches.

REFERENCES

[1] Amit, Y. and Geman, D.(1997). Shape quantization and recognition with randomized trees. *Neural Computation* **9** (7): 1545–1588.
[2]Basel Committee on Banking Supervision (2004). International Convergence of Capital Measurement and Capital Standards http://www.bis.org/publ/bcbs118.pdf
[3] Basel Committee on Banking Supervision (2005). An Explanatory Note on the Basel II IRB Risk Weight Functions. July 2005. http://www.bis.org/bcbs/irbriskweight.pdf
[4] Breiman, L. (1996). Bagging Predictors. Machine Learning 24: 123-140 http://www.cs.utsa.edu/~bylander/cs6243/breiman96bagging.pdf
[5] Breiman, L. (1996). Heuristics of instability and stabilization in model selection. Ann. Statist. 24:2350-2383
[6] Breiman, L. (1996). Technical Note: Some Properties of Splitting Criteria. Machine Learning 24: 41-47 http://www.cba.ua.edu/~mhardin/BreimanMachineLearning1996.pdf
[7] Breiman, L. (2001). Random Forests, Machine Learning 45: 5-32
[8] Breiman, L., Friedman, J. H., Olshen, R. A., Stones, C. J. (1984). Classification and Regression Trees. Chapman & Hall, New York
[9] Buhlmann, P. and Yu, B. (2003). Analyzing bagging, Ann. Statist. 30: 927-961
[10] Chou, P. A. (1991). Optimal Partitioning for Classification and Regression Trees. IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 13 (4): 340-354 http://www.cs.nyu.edu/~roweis/csc2515-2006/readings/chou_pami.pdf
[11] Friedman, J., Hastie, T., and Tibshirani, R. (2008). The Elements of Statistical Learning. 2nd Edition, Springer
[12] Ho, T. K. (1995). Random Decision Forest. Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282.
[13] Ho, T. K.(1998). The Random Subspace Method for Constructing Decision Forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20** (8): 832–844.
[14] Jacobs, M. (2008). An Empirical Study of Exposure at Default, Risk Analysis Division/Credit Risk Modeling. Moody's KMV Credit Practitioner's Conference
[15] Lin, Y. and Jeon, Y. (2006). Random forests and adaptive nearest neighbors. J. Amer. Statist. Assoc. 101:578-590
[16] Pindyck, R. S. and Rubinfeld, D. L. (1998). Econometric Models and Economic Forecasts. 4th Edition, Irwin/McGraw-Hill, pp. 159-175
[17] SAS Institute Inc. (1997). SAS/Stat Software: Changes and Enhancements through Release 6.12, pp.431-432
[18] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brain P. Flannery (2003). Numerical Recipes in C++, Cambridge University Press, 2nd Edition, 2003
[19] Yang, B. H. and Tkachenko, M. (2012). Modeling Exposure at Default and Loss Given Default: Empirical approaches and technical implementation. Journal of Credit Risk, Volume 8 (2), pp. 81-102

Appendix

Proof of Proposition 3.1. We have

$$
\begin{aligned}
d(v) &= |\,bcdist(v) - \mathrm{gcd}\,ist(v)\,| \\
&= \left|\frac{Cum\,Sum\,y}{Tot\,Sum\,y} - \frac{Cum\,Sum(1-y)}{Tot\,Sum\,(1-y)}\right| \\
&= \frac{|\,Tot\,Sum\,(1-y)\times Cum\,Sum\,y - Tot\,Sum\,y\times Cum\,Sum\,(1-y)\,|}{Tot\,Sum\,y\times Tot\,Sum\,(1-y)} \\
&= \frac{|\,(Tot\,Sum\,(1-y) + Tot\,Sum\,y)\times Cum\,Sum\,y - n_1\times Tot\,Sum\,y\,|}{Tot\,Sum\,y\times Tot\,Sum\,(1-y)} \qquad \text{(A.1)} \\
&= \frac{|\,N\times Cum\,Sum\,y - n_1\times Tot\,Sum\,y\,|}{Tot\,Sum\,y\times Tot\,Sum\,(1-y)} \qquad \text{(A.2)} \\
&= \frac{|\,(N-n_1)\times Cum\,Sum\,y - n_1\times(Tot\,Sum\,y - Cum\,Sum\,y)\,|}{Tot\,Sum\,y\times Tot\,Sum\,(1-y)} \\
&= \frac{n_1\times(N-n_1)}{Tot\,Sum\,y\times Tot\,Sum\,(1-y)}\times\left|\frac{Cum\,Sum\,y}{n_1} - \frac{Tot\,Sumy - Cum\,Sum\,y}{N-n_1}\right| \\
&= cp(v)\,|\,\bar{y}_L(v) - \bar{y}_R(v)| \\
&= cp(v)D(v)
\end{aligned}
$$

where (A.1) and (A.2) follow respectively from the facts that *Cum Sum* (1-y)= n_1 - *Cum Sumy* and *Tot Sum* (1-y) + *Tot Sum* y =N. □

Proof of Proposition 3.3. We have:

$$
\begin{aligned}
&\bar{y} = \frac{n_1}{N}\bar{y}_1 + \frac{n_2}{N}\bar{y}_2 \\
&\Rightarrow SSB = n_1(\bar{y}_1 - \bar{y})^2 + n_2(\bar{y}_2 - \bar{y})^2 \\
&= n_1\left(\frac{(n_1+n_2)\bar{y}_1}{N} - \frac{n_1\bar{y}_1 + n_2\bar{y}_2}{N}\right)^2 + n_2\left(\frac{(n_1+n_2)\bar{y}_2}{N} - \frac{n_1\bar{y}_1 + n_2\bar{y}_2}{N}\right)^2 \\
&= \frac{n_1 n_2^{\,2}}{N^2}(\bar{y}_1 - \bar{y}_2)^2 + \frac{n_1^{\,2} n_2}{N^2}(\bar{y}_1 - \bar{y}_2)^2 \\
&= \frac{n_1 n_2(n_1+n_2)}{N^2}(\bar{y}_1 - \bar{y}_2)^2 \\
&= \frac{n_1 n_2}{N}(\bar{y}_1 - \bar{y}_2)^2 \\
&= Np(v)(\bar{y}_L(v) - \bar{y}_R(v))^2\ /4 \\
&= N\ p(v)(D(v))^2\ /4 \\
&\Rightarrow ICC(v) = \frac{SSB}{SST} = \frac{N}{4\times SST}\,p(v)(D(v))^2 = c\ p(v)(D(v))^2\ \square
\end{aligned}
$$