Supplementary Material

A Detailed Proofs

A.1 Proof for Lemma 1

Proof. We first show $||\boldsymbol{u}||_\infty \leq 1$:

For any production of finite basis functions from Φ,

$$||\prod_{1=1}^{L} \phi_{i_l}(x_{j_l})||_\infty \leq \prod_{1=1}^{L} ||\phi_{i_l}(x_{j_l})||_\infty \leq 1$$

Each component of $\boldsymbol{u}$ is a production of finite basis functions from Φ. Thus $||\boldsymbol{u}||_\infty \leq 1$.

Then we show $||\boldsymbol{v}||_1 \leq ||\boldsymbol{w}||_1$ if $||\boldsymbol{w}||_1 \leq 1$ by induction:

$k = 0$, $||v||_1 = ||w||_1$;

Assume that for any $k < K$ and any weighted labeled binary tree $h \in \mathcal{W}_{2k+1}$, $||v_h||_1 \leq ||w_h||_1$. For $k = K$, decompose the tree $h(\boldsymbol{x}; f, \boldsymbol{w}) \in \mathcal{W}_{2K+1}$ by the left subtree $h_l(\boldsymbol{x}; f_l, \boldsymbol{w}_l) = \langle \boldsymbol{v}_l, \boldsymbol{u}_l \rangle$ and the right subtree as $h_r(\boldsymbol{x}; f_r, \boldsymbol{w}_r) = \langle \boldsymbol{v}_r, \boldsymbol{u}_r \rangle$.

If the root is a "+", then $||\boldsymbol{v}||_1 = ||\boldsymbol{v}_l||_1 + ||\boldsymbol{v}_r||_1 \leq ||\boldsymbol{w}_l||_1 + ||\boldsymbol{w}_r||_1 = ||\boldsymbol{w}||_1$.

If the root is a "*", then

$$\begin{aligned}
||\boldsymbol{v}||_1 &= \sum_t \sum_s |v_l^t v_r^s| \\
&= \sum_t |v_l^t| \sum_s |v_r^s| \\
&= \sum_t |v_l^t| ||\boldsymbol{v}_r||_1 \\
&= ||\boldsymbol{v}_l||_1 ||\boldsymbol{v}_r||_1 \\
&\leq ||\boldsymbol{w}_l||_1 ||\boldsymbol{w}_r||_1 \\
&\leq ||\boldsymbol{w}||_1^2 \\
&\leq ||\boldsymbol{w}||_1
\end{aligned}$$

□

A.2 Proof for Lemma 2

Proof. Remind that p is the dimension of the covariate, and q is the number of basis functions. We define $\mathcal{F}^*_{2k+1} \subset \mathcal{F}_{2k+1}$ as the set of labeled binary trees with exactly $2k+1$ nodes. In this step, we will show that $|\mathcal{F}^*_{2k+1}| \leq 2^k (k)! (pq)^{k+1}$.

We first show $|\mathcal{F}^*_{2k+1}| \leq (pq)^{k+1}(k)!2^k$ for all $k = 0, 1, \cdots$:

$k = 0$, $|\mathcal{F}^*_{2*0+1}| = pq \leq (pq)^{0+1}(0)!2^0$;

$k = 1$, $|\mathcal{F}^*_{2*1+1}| = 2(pq)^2 - 2pq < (pq)^{1+1}(1)!2^1$;

Assume that $|\mathcal{F}^*_{2*k+1}| \leq (pq)^{k+1}(k)!2^k$ for all $k < K$, then for $k = K$,

$$\begin{aligned}
|\mathcal{F}^*_{2K+1}| &= 2 \sum_{i \in \{1,3,\cdots,2K-1\}} |\mathcal{F}^*_i||\mathcal{F}^*_{2K-i}| \\
&\leq 2 \sum_{i=0,\cdots,K-1} (pq)^{i+1}(i)!2^i \\
&\quad (pq)^{K-i-1+1}(K-i-1)!2^{K-i-1} \\
&= (pq)^{K+1}2^K \sum_{i=0,\cdots,K-1} (i)!(K-i-1)! \\
&\leq (pq)^{K+1}2^K \sum_{i=0,\cdots,K-1} (K-1)! \\
&\leq (pq)^{K+1}2^K (K)!
\end{aligned}$$

Since for $k \geq 1$, we have $2^{k-1} = \sum_{i=0}^{k-1} \frac{(k-1)!}{i!(k-1-i)!}$, or equivalently, $\frac{2^{k-1}}{(k-1)!} = \sum_{i=0}^{k-1} \frac{1}{i!(k-1-i)!}$, and since $1/x$ is concave, by Jensen's inequality, we have that $\frac{2^{k-1}}{(k)!} = \sum_{i=0}^{k-1} \frac{1}{k} \frac{1}{i!(k-1-i)!} \leq \frac{1}{\sum_{i=0}^{k-1} i!(k-1-i)!/k}$. Thus $\sum_{i=0}^{k-1} i!(k-1-i)! \leq k\frac{(k)!}{2^{k-1}}$ for $k \geq 1$. Except for the root node, a labeled binary tree consists of the left subtree and the right subtree. Thus

$$\begin{aligned}
|\mathcal{F}^*_{2k+1}| &= 2 \sum_{i \in \{1,3,\cdots,2k-1\}} |\mathcal{F}^*_i||\mathcal{F}^*_{2k-i}| \\
&\leq (pq)^{k+1}2^k \sum_{i=0,\cdots,k-1} (i)!(k-i-1)! \\
&\leq (pq)^{k+1}2^k k \frac{(k)!}{2^{k-1}} \\
&= 2k(k)!(pq)^{k+1}
\end{aligned}$$

Finally, we will prove that $|\mathcal{F}_{2k+1}| \leq 4k(k)!(pq)^{k+1}$.

$$\begin{aligned}
|\mathcal{F}_{2k+1}| &= \sum_{i=0}^{k} |\mathcal{F}^*_{2i+1}| \\
&\leq \sum_{i=1}^{k-1} 2i(i)!(pq)^{i+1} + pq + 2k(k)!(pq)^{k+1} \\
&\leq k * 2(k-1)(k-1)!(pq)^{k-1+1} + 2k(k)!(pq)^{k+1} \\
&\leq 4k(k)!(pq)^{k+1}
\end{aligned}$$

□

A.3 Proof for Lemma 3

Proof. Define $M^*_{2k+1} = \max_{f \in \mathcal{F}^*_{2k+1}} M_f$. Since $M^*_{2k+1} = M_{2k+1}$, it is equivalent to show $M^*_{2k+1} < (1.45)^{k+1}$. We will prove the lemma by induction.

$k=0$, $M^*_{2*0+1} = 1 < (1.45)^1$;

$k=1$, $M^*_{2*1+1} = max(1, 1+1) = 2 < (1.45)^2$;

$k=2$, $M^*_{2*2+1} = 3 < (1.45)^3$;

Assume that $M^*_{2k+1} < (1.45)^{k+1}$ for all $k < K$, where $K \geq 3$, then for $k = K$,

$$\begin{aligned} M^*_{2k+1} &= \max_{i\in\{1,3,\cdots,2K-1\}} [\max(M^*_i M^*_{2K-i}, M^*_i + M^*_{2K-i})] \\ &< \max_{i\in\{1,3,\cdots,2K-1\}} [\max(1.45^{\frac{i-1}{2}+1} 1.45^{\frac{2K-i-1}{2}+1}, \\ &\qquad 1.45^{\frac{i-1}{2}+1} + 1.45^{\frac{2K-i-1}{2}+1})] \\ &= (1.45)^{K+1} \end{aligned}$$

□

B Technical Lemma

The following technical lemma regarding the McDiarmid's condition for the supremum can be found in [2].

Lemma 4. *Let z be a random variable of support $\mathcal{Z} = (\mathbb{R}^p, \mathcal{Y})$ and distribution $\mathcal{D}$. Let $S = \{z_1 \dots z_n\}$ be a dataset of n samples. Let $\mathcal{H}$ be a hypothesis class satisfying $\mathcal{H} \subseteq \{h \mid h : \mathcal{Z} \to [0,1]\}$. The function:*

$$\varphi(S) = \sup_{h\in\mathcal{H}} \left(\mathbb{E}_{\mathcal{D}}[h] - \widehat{\mathbb{E}}_S[h] \right) \tag{11}$$

satisfies the following condition:

$$|\varphi(z_1, \dots, z_i, \dots, z_n) - \varphi(z_1, \dots, \widetilde{z}_i, \dots, z_n)| \leq 1/n$$
$$(\forall i, \forall z_1 \dots z_n, \widetilde{z}_i \in \mathcal{Z})$$

C Detailed Greedy Search Algorithm and Illustration Example

For completeness, we present our main greedy search algorithm in detail in Algorithm 1, as well as the algorithm to compute the node weights in Algorithm 2. For simplicity, we assume the covariate $\boldsymbol{x}_m \in [0,1]^p$. As for the set of basis functions Φ, piecewise linear functions, Fourier basis functions, or truncated polynomials could be good choices in practice. We first define $f_{\boldsymbol{w}}(\boldsymbol{x})$ as the output of tree structure f with weights $\boldsymbol{w}$ for input $\boldsymbol{x}$. For instance, let f be the tree structure of Figure 2(a). With a corresponding weight for each leaf, $f_{\boldsymbol{w}}$ can be visualized as in Figure 2(b). Thus $f_{\boldsymbol{w}}(\boldsymbol{x}) = (w_1\phi_1(x_2) + w_2\phi_3(x_1)) * (w_3\phi_3(x_2) + w_4\phi_1(x_3))$ in this specific case. The loss function is defined as $L(f_{\boldsymbol{w}}; \boldsymbol{x}, \boldsymbol{y}) = \sum_{m=1}^{n} (y_m - \hat{y}_m)^2/2$, where $\hat{y}_m = f_{\boldsymbol{w}}(\boldsymbol{x}_m)$. We could explore the interaction structure f by adding and multiplying a basis function on a single dimension of covariate $\boldsymbol{x}$.

Algorithm 1 Greedy search algorithm

Input: $\boldsymbol{X} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n)' \in \mathbb{R}^{n\times p}$: n data points
$\boldsymbol{y} = (y_1, \dots, y_n) \in \mathbb{R}^n$: n observations
Φ: a set of q basis functions, k: the number of iterations
Initialize the tree $f_{\boldsymbol{w}} = w_0 + w_1\phi_{i_1}(x_{j_1})$, where $(w_0, w_1, i_1, j_1) = \underset{(w'_0, w', i', j')}{\arg\min} \sum_{m=1}^{n} (y_m - w'_0 - w'\phi_{i'}(x_{j'}))^2$
for $iters = 1$ **to** $k-1$ **do**
  for $node$ in $f_{\boldsymbol{w}}.leaves$ **do**
    $path = path(f_{\boldsymbol{w}}.root, node)$
    for $m = 1$ **to** n **do**
      Algorithm 2 with input $(\boldsymbol{x}_m, f_{\boldsymbol{w}}, path)$:
      $b_m = b(\boldsymbol{x}_m)$, $k_m = k(\boldsymbol{x}_m)$
      $c_m = node(\boldsymbol{x}_m)$ (If $node$ is $w\phi_i(x_j)$, then $node(\boldsymbol{x}_m) = w\phi_i(x_{mj})$)
    end for
    $(w_0, w_+, i_+, j_+) = \underset{(w'_0, w' \leq 1, i', j')}{\arg\min} \sum_{m=1}^{n} (y_m - w'_0 - b_m - k_m(c_m + w'\phi_{i'}(x_{mj'})))^2$, and define r_+ as the corresponding minimum value attained.
    $(w_0, w_*, i_*, j_*) = \underset{(w'_0, w' \leq 1, i', j')}{\arg\min} \sum_{m=1}^{n} (y_m - w'_0 - b_m - k_m(c_m w'\phi_{i'}(x_{mj'})))^2$, and define r_* as the corresponding minimum value attained.
    if $r_+ < r_*$ **then**
      Insert the new leaf $w_+\phi_{i_+}(x_{j_+})$ at $node$ with "+", and call the new tree $f^{node}_{\boldsymbol{w}}$
      $r_{node} = r_+$
    else
      Insert the new leaf($w_*\phi_{i_*}(x_{j_*})$ at $node$) with "*", and call the new tree $f^{node}_{\boldsymbol{w}}$
      $r_{node} = r_*$
    end if
    Adjust all weights
  end for
  if $r_{node} < r_{best}$ **then**
    $r_{BEST} = r_{node}$, $f^{best}_{\boldsymbol{w}} = f^{node}_{\boldsymbol{w}}$
  end if
  Update $f_{\boldsymbol{w}}$ with $f^{best}_{\boldsymbol{w}}$
end for
Output: $f_{\boldsymbol{w}}$

Algorithm 2 Compute $b(\boldsymbol{x}_m)$ and $k(\boldsymbol{x}_m)$

Input: $\boldsymbol{x}_m \in \mathbb{R}^p$: data point
$f_{\boldsymbol{w}}$: current weighted labeled tree
path: path from the root to the insert position
Initialize *root* as the root of $f_{\boldsymbol{w}}$, k=1, b=0
while *path* is not empty **do**
  Define *subtree* as the !*path*[1] subtree of *root*
  $val = evaluate(subtree, \boldsymbol{x}_m)$, where *evaluate* gives the output of the weighted labeled tree *subtree* with input $\boldsymbol{x}_m$
  if *root* = " + " **then**
    $b = b + val * k$
  else if *root* = " ∗ " **then**
    $k = val * k$
  end if
  Update *root* as its *path*[1] child
  Remove the first element of *path*
end while
Output: (b, k)

An example to illustrate Algorithm 2. Take Figure 7 for example, and assume we are trying to insert a new leaf wx_4^3 with either a "+" or "*" at the Node E, that is to replace the weighted leaf $-.05x_1$ with either $-.05x_1 + wx_4^3$ or $-.05x_1 * wx_4^3$. With an unknown weight w and an unknown intercept w_0, the output $\hat{y}_m$ for the input $\boldsymbol{x}_m$ of the new tree is

$$w_0 + [.1x_{m2}^2 - .05x_{m1} + wx_{m4}^3](.3\sin(\pi x_{m2}) + .02x_{m3})$$
$$\triangleq w_0 + b(\boldsymbol{x}_m) + k(\boldsymbol{x}_m)(wx_{m4}^3 - .05x_{m1})$$

for "+", and

$$w_0 + [.1x_{m2}^2 + wx_{m4}^3(-.05)x_{m1}](.3\sin(\pi x_{m2}) + .02x_{m3})$$
$$= w_0 + b(\boldsymbol{x}_m) + k(\boldsymbol{x}_m)(-.05wx_{m4}^3 x_{m1})$$

for "*".

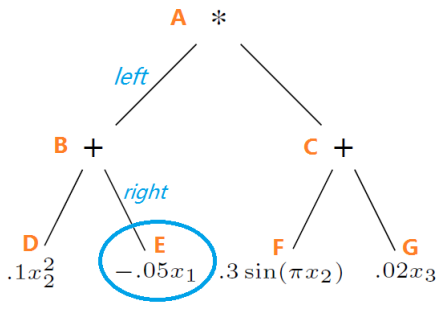

Figure 7: Inserting a new leaf at Node E.

Note that $b(\boldsymbol{x}_m)$ and $k(\boldsymbol{x}_m)$ are constant with respect to the to-be-defined weight, and thus, the optimization problems $\min_w \sum_{m=1}^n (y_m - w_0 - b(\boldsymbol{x}_m) - k(\boldsymbol{x}_m)(wx_{m4}^3 - .05x_{m1}))^2$ and $\min_w \sum_{m=1}^n (y_m - w_0 - b(\boldsymbol{x}_m) - k(\boldsymbol{x}_m)(-.05wx_{m4}^3 x_{m1}))^2$ are both least square problems. We add a constraint $|w| \leq 1$ according to the assumption of Theorem 1, to ensure the uniform convergence. However, it is not straightforward to compute $b(\boldsymbol{x}_m)$ and $k(\boldsymbol{x}_m)$. As shown in Algorithm 2, we compute the value of $b(\boldsymbol{x}_m)$ and $k(\boldsymbol{x}_m)$ iteratively along the path from the root to the insert position. We continue with our current setting, and move on to compute $b(\boldsymbol{x}_m)$ and $k(\boldsymbol{x}_m)$ according to Algorithm 2, assuming $\boldsymbol{x}_m = (1, 1, 1)$.

1. Input: $\boldsymbol{x}_m = (1, 1, 1)$, $f_{\boldsymbol{w}}$ is the tree in Figure 7, $path = (left, right)$
2. Initialize: *root* =Node A, $k = 1$,$b = 0$
3. In a first iteration $path[1] = left$, so define *subtree* as the *right* =!*left* subtree of *root*(consisting of Nodes C, F, G),
   $val_m = evaluate(subtree, \boldsymbol{x}_m) = .3\sin(\pi x_{m2}) + .02x_{m3} = .02$
4. Since *root* = " ∗ ", $k = val_m * k = .02$
5. Update *root* as its left child: *root* =Node B, $path = (right)$ after removing the first element of path
6. In a second iteration $path[1] = right$, so update *subtree* as the *left* =!*right* subtree of *root* (consisting of Node D only)
   $val_m = evaluate(subtree, \boldsymbol{x}_m) = .1x_{m2}^2 = .1$
7. Since *root* = " + ", $b = b + val_m * k = .002$
8. Update *root* as its right child, $path = ()$ after removing the first element of path
9. Stop the iterations since *path* is empty
10. Return $(b(\boldsymbol{x}_m) = .002, k(\boldsymbol{x}_m) = .02)$

D Real World Experiments

Airline Delays. For real-world experiments, we evaluate our algorithm on the US flight dataset. We use a subset of the data with flight arrival and departure times for commercial flights in 2008. The dataset is publicly available at http://statcomputing.org/dataexpo/2009/. The flight delay is the response variable, which is predicted by using the following variables: the age of the aircraft, distance that needs to be covered, airtime, departure time, arrival time, day of the week, day of the month, and

month. We randomly select 800,000 datapoints, using a random subset of 700,000 samples to train the model and 100,000 to test it. Although our method uses only $k = 10$ (i.e., $2k + 1 = 21$ nodes, or $k + 1 = 11$ functions of features), we obtain a test RMSE of 34.89. For comparison, the authors in [7] also randomly selected 800,000 samples (700,000 for training, 100,000 for testing) and obtained an RMSE between 32.6 and 33.5 with 1200 iterations on a *Gaussian processes* approach. In general, Gaussian processes predict the output by memorization of the 700,000 training points. Our tree depends only on evaluating $k + 1 = 11$ functions of features. When predicting, our tree does not need to remember the training set.

World Weather. The world weather dataset contains monthly measurements of temperature, precipitation, vapor, cloud cover, wet days and frost days from Jan 1990 to Dec 2002 (156 months) on a 5×5 degree grid that covers the entire world. The dataset is publicly available at http://www.cru.uea.ac.uk/. The response variable is temperature. We use 19,000 samples for training, 8000 samples for testing, and run 30 iterations. Although our method uses only $k = 30$ (i.e., $2k + 1 = 61$ nodes, or $k + 1 = 31$ functions of features), we obtain a test RMSE of 1.319. Gaussian processes obtained a test RMSE of 1.23. Since the standard deviation of the output variable is 16.98, both our method and Gaussian processes obtain a coefficient of determination of 0.99.