REGRESSION TREES AND RANDOM FORESTS FOR PREDICTIONS

DUŠAN OBERTA

Abstrakt. Regression trees are widely used in statistics to capture, not always trivial, relationships between predictors (i.e. independent variables) and a response variable (i.e. dependent variable). They can be used in a variety of situations where other statistical tools are not suitable, even in situations where the number of predictors is greater than the number of observations in the set of training data. Random forests generalize the concept of regression trees to reduce variance and improve stability of simple regression trees. Apart from the classical regression trees based on the least squares method, the concept of maximum likelihood with the assumption of gamma distribution of the response variable is described and derived by the author. Compared to literature found, slightly different proofs of theorems regarding pruning of regression trees are offered, as well as a thorough derivation of confidence intervals for the expected value of the response variable is offered as own work of the author. Introduction to the concept of random forests is covered in the last part of the article.

1. Introduction

Nowadays, we are surrounded by huge amounts of data. Considering the fact that more and more data are obtained every day from numerous human activities, it is crucial to know how to process the data and model the relationship between, usually multiple, predictor variables and a response variable properly. Predictor variable (i.e. predictor), also known as an explanatory variable, is an independent variable that can be changed by an observer in order to obtain an outcome (i.e. response variable). Predictors can be either real-valued (called continuous predictors), or they can take only the values from a finite set (in this case, they are called categorical predictors). A response variable is a dependent variable, which is influenced by predictors. Our goal is to create a statistical model which best

2020 *MSC.* Primární 62F40; Sekundární 62F25, 62H30, 68T05.

Klíčová slova. Regression trees, least squares, maximum likelihood estimation, confidence intervals, k-fold cross-validation, bagging, bootstrapping, random forests.

Článok vznikol na základe bakalárskej práce autora v obore Matematické inženýrství na FSI VUT v Brne. Vedúcou práce bola Zuzana Hübnerová z Ústavu matematiky FSI VUT v Brně.

Ďakujem doc. Hübnerovej za ochotu, pomoc a priateľský prístup počas písania bakalárskej práce, v ktorej sa pojednáva (nielen) o problematike regresných stromov a náhodných lesov. Taktiež by som sa chcel poďakovať svojej rodine za podporu počas štúdia aj mimo neho.

captures the relationship, based on some observations made (i.e. a set of training data).

Imagine we know that a response variable Y is explained by an explanatory variable X by the relationship $Y = \alpha + \beta X + \varepsilon$. Based on a set of n observations $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ we have done, we want to compute the estimations $\widehat{\alpha}, \widehat{\beta}$ of the real values of parameters α, β. In real world, each of the observations satisfies the equation $y_i = \alpha + \beta x_i + \epsilon_i$ (for $i = 1, \ldots, n$), where ϵ_i is a random variable known as a random error, which is the difference between the observed and the real value of an experiment. Thought this article we assume that $\epsilon_1, \ldots, \epsilon_n$ are independent, meaning that a random error which occurs while making one observation does not affect any other observations made. This is quite a reasonable assumption, anyway it is not always fulfilled in real world applications. The case where the random errors are not independent is not discussed in this article, and more information regarding this topic can be found in [5]. Also, when deriving confidence intervals for the expected value of the response variable for *regression trees* (see Section 2.6), we assume that $\epsilon_1, \ldots, \epsilon_n$ are normally distributed with zero mean and constant variance. Moreover, it is definitely a reasonable requirement that the computed estimations $\widehat{\alpha}, \widehat{\beta}$ are the "best" (in some sense), whatever the "best" means. Depending on the "best" criterion used, we obtain different statistical models. The two "best" criterions used for *regression trees* in this article are a well-known least squares criterion (see Section 2.3) and a maximum likelihood criterion (see Section 2.7) with the assumption of gamma distribution of the response variable (for simplicity, the words "assumption of gamma distribution" will be sometimes omitted, since in this article we will only consider maximum likelihood estimation with the assumption of gamma distribution of the response variable).

Probably the best-known tool for making predictions is *linear regression*. Being the best-known comes with certain limitations, such as the relationship between predictors and the expected value of the response variable must be linear. In order to derive interval estimates of unknown parameters using *linear regression*, normal distribution of the response variable is assumed. Generalization of *linear regression* is offered by *generalized linear models*, where the distribution of the response variable can be any distribution from the so called *exponential family*, and the expected value of the response variable is connected to the predictors by a link function, which is a strictly monotone, differentiable function. Nevertheless, there are still some limitations concerning this statistical models (e.g. the number of observations must be at least equal to the number of unknown parameters). More detailed information regarding *linear regression* can be found in [1] or [4], and regarding the *generalized linear models*, in [2].

Regression trees, described in Section 2, have no limitations concerning the relationship between predictors and the response variable. Also, as mentioned in the abstract, there are no limitations regarding the number of unknown parameters and the number of observations, which is not true for both *linear regression* and *generalized linear models*. They are also known for their simplicity and interpretability. On the other hand, they might be unstable, meaning that a slightly different data set might result in quite different predictions. Two different approaches of growing the *trees* are described, least squares (see Section 2.3) and maximum likelihood estimation (see Section 2.7), with the latter being own derivation of the author. Confidence intervals for the expected value of the response variable (assuming a fixed partition of the sample space) are derived thoroughly in Section 2.6. Also, *k-fold cross-validation* is shown as a tool for finding optimal values of tuning parameters (not only) for *regression trees*. More information regarding *regression trees* can be found in [4].

As mentioned in the previous paragraph, *regression trees* tend to be unstable. Section 3 provides an introduction to the concept of *random forests*, which are built using the basic concepts of bootstrapping, bagging and *regression trees* in order to reduce variance of the model when compared to single *regression trees* and improve its stability. As in the previous section, confidence intervals for the expected value of the response variable (assuming a fixed partition of the sample space) are stated, with a sketch of their derivation. Also specific algorithm for the practical implementation of *random forests* is described. More detailed information regarding *random forests* can be found in [4] and [8], and regarding the confidence intervals in [3].

2. Regression Trees

In this chapter, we will introduce some basic concepts regarding *regression trees*. Proper mathematical definition of a *tree* (see Definition 2.1) requires defining some terms from graph theory. However, for purposes of this article, more intuitive and vague definitions of the terms and concepts (e.g. Definition 2.2) will be sufficient.

Regression trees are used to model a relationship between predictors and a response variable. They are, for example, suitable to use in situations where either the relationship is too complicated to capture by a simple model, or the number of independent variables is relatively large when compared to the number of observations.

2.1. Basic Concepts

Definition 2.1. A *tree* is an *undirected graph*, in which any two *vertices* are connected by exactly one *path*.

Definition 2.2. A *binary tree T* consists of a non-empty set of *nodes*, such that each *node* (called *parent node*) contains either no *subnodes*, or precisely two *subnodes* (called *child nodes*), and that there is exactly one *node* (called the *root*), which is not a *child node* of any other *node*. *Nodes* containing no *child nodes* are called the *leaves* or *terminal nodes*.

Remark. Since we are interested only in *binary trees*, for simplification, instead of the term *binary tree*, we will be using only the term *tree* (omitting the word "binary").

2.2. Introduction to Regression Trees

The main idea behind *regression trees* is that by starting with all the data at the *root*, we move downwards until a *leaf* is reached. At each *parent node*, we move towards one of its two *child nodes* depending on the values of predictors – we select a predictor and its value, which provide the "best" partition (according to some criterion), and then each data sample, which was originally in the *parent node* is moved into the corresponding *child node* (depending on the selected "best" predictor and its value). The selected predictor is called a *split predictor*, its value is called a *split value*, and together they are called a *split*. We stop the splitting process when some minimal *node size* (i.e., the minimal number of observations in a *node*) is reached.

Let $X_1, \dots, X_l$ be continuous predictors and $X_{l+1}, \dots, X_k$ be categorical predictors with $m_{l+1}, \dots, m_k$ categories (i.e., $X_i \in \mathbb{R}$ and $X_j \in G_j := \left\{g_j^1, \dots, g_j^{m_j}\right\}$, for $i = 1, \dots, l$ and $j = l+1, \dots, k$). Denote the sample space

$$\mathcal{D} := \mathbb{R}^l \times \prod_{i=l+1}^{k} G_i.$$

Let $\{R_i\}_{i=1}^m$ be a finite partition of $\mathcal{D}$, such that for $j = 1, \dots, m$, each region R_j is of the form

$$R_j = \prod_{i=1}^{l} (a_i; b_i] \times \prod_{i=l+1}^{k} S_i,$$

where S_i is a non-empty subset of G_i and $a_i, b_i \in \mathbb{R} \cup \{-\infty, \infty\}$ (in case $b_i = \infty$, the corresponding interval is considered to be open from the right side).

Let $Y_1, \dots, Y_n$ be random variables, $\mathbf{X} \in \mathcal{D}^n$ an $n \times k$ matrix (each of the n rows of $\mathbf{X}$ being an element of $\mathcal{D}$), $\beta_1, \dots, \beta_m$ unknown parameters and $\varepsilon_1, \dots, \varepsilon_n$ random variables such that $\mathrm{E}\left(\varepsilon_i\right) = 0$ and $\mathrm{var}\left(\varepsilon_i\right) = \sigma^2$, for $i = 1, \dots, n$. Denote $\mathbf{x}_i := \left(x_{i1}, \dots, x_{ik}\right)^T \in \mathcal{D}$ and $\mathbf{X}_j := \left(x_{1j}, \dots, x_{nj}\right)^T$. We can write $\mathbf{X}$ in the form

$$\mathbf{X} = \begin{pmatrix} x_{11} & \dots & x_{1k} \\ & \ddots & \\ x_{n1} & \dots & x_{nk} \end{pmatrix} = \begin{pmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_n^T \end{pmatrix} = \begin{pmatrix} \mathbf{X}_1 & \dots & \mathbf{X}_k \end{pmatrix}.$$

Now assume a regression model, where only a constant function is fitted in each of the partition regions, i.e. the corresponding model is of the form

$$Y_i = f\left(\mathbf{x}_i\right) + \varepsilon_i = \sum_{j=1}^{m} \beta_j \chi_{R_j}\left(\mathbf{x}_i\right) + \varepsilon_i; \; i = 1, \dots, n, \tag{2.1}$$

where χ_{R_j} is an indicator function of subset R_j of set $\mathcal{D}$, i.e.

$$\chi_{R_j}\left(\mathbf{x}\right) := \begin{cases} 1, & \text{if } \mathbf{x} \in R_j, \\ 0, & \text{else.} \end{cases}$$

Remark. The model defined in (2.1) is typically used for *regression trees*, when the partition is done according to the least squares criterion (see Section 2.3).

When considering the maximum likelihood criterion with the assumption of gamma distribution of the response variable (see Section 2.7), this model is no longer valid.

Remark. The partition $\{R_i\}_{i=1}^{m}$ described in this section simply means, that the $\mathbb{R}^l$ part of the sample space is divided into l-dimensional rectangles for continuous predictors, and the finite sets corresponding to categorical predictors are divided into non-empty subsets.

2.3. Growing a Regression Tree

2.3.1. Parameters Estimation. Consider a set of training data

$$\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\},$$

where $\mathbf{x}_i = (x_{i1}, \dots, x_{ik})^T$, for $i = 1, \dots, n$. Notice that there are no assumptions regarding the relationship between n and k. Suppose, in this section, that we already have a partition of $\mathcal{D}$ into m regions $R_1, \dots, R_m$. Assuming that the model is of the form (2.1), we want to estimate the values of parameters $\beta_1, \dots, \beta_m$. Denote $f_i := f(\mathbf{x}_i)$ and consider the criterion of minimizing the residual sum of squares, i.e. minimizing the expression

$$RSS := RSS(\beta_1, \dots, \beta_m) = \sum_{i=1}^{n} (y_i - f_i)^2 = \sum_{i=1}^{n} y_i^2 - 2\sum_{i=1}^{n} y_i f_i + \sum_{i=1}^{n} f_i^2. \qquad (2.2)$$

Differentiating both sides of (2.2) with respect to β_u and setting it to 0, we obtain

$$\frac{\partial RSS}{\partial \beta_u} = \cdots = -2 \sum_{\{i \mid \mathbf{x}_i \in R_u\}} y_i + 2\,\mathrm{card}\left(\{i \mid \mathbf{x}_i \in R_u\}\right) \beta_u = 0, \qquad (2.3)$$

where $\mathrm{card}(A)$ denotes the cardinality of set A. Denote $\mathcal{C}_u := \mathrm{card}(\{i \mid \mathbf{x}_i \in R_u\})$. Assuming that $\{i \mid \mathbf{x}_i \in R_u\}$ is non-empty (i.e., each of the partition regions contains at least one element), (2.3) has a unique solution

$$\widehat{\beta}_u = \frac{1}{\mathcal{C}_u} \sum_{\{i \mid \mathbf{x}_i \in R_u\}} y_i. \qquad (2.4)$$

Hence $\widehat{\beta}_u$ is just the average of $y_i \mid \mathbf{x}_i \in R_u$. It is an easy exercise to verify that the *Hessian* matrix of (2.2) is positive definite, thus the expression (2.4) minimizes (2.2).

Remark. Details of (2.3), as well as verifying the positive definiteness of the *Hessian* matrix of (2.2) can be found in [7].

2.3.2. Finding an Optimal Partition. Now, assuming (2.4) is the least squares estimator of β_u, we need to find an optimal partition of the sample space $\mathcal{D}$.

For a categorical predictor X_j ($j \in \{l+1, \dots, k\}$), define the set of all possible non-empty *binary splits* as

$$\mathcal{B}_j := \{\{S; G_j \setminus S\} \mid S \subset G_j \wedge S \notin \{\emptyset; G_j\}\}.$$

Denote J_j the number of all possible non-empty *binary splits* for a categorical predictor X_j

$$J_j := \operatorname{card}(\mathcal{B}_j) = \frac{\operatorname{card}(\mathcal{P}(G_j) \setminus \{\emptyset; G_j\})}{2} = \frac{2^{\operatorname{card}(G_j)} - 2}{2} = 2^{\operatorname{card}(G_j)-1} - 1,$$

where $\mathcal{P}(A)$ denotes the power set of A. Thus we can write $\mathcal{B}_j$ as

$$\mathcal{B}_j = \left\{B_j^i\right\}_{i=1}^{J_j} = \left\{\left\{B_j^{i;1}; B_j^{i;2}\right\}\right\}_{i=1}^{J_j},$$

where B_j^i denotes the i-th element of $\mathcal{B}_j$.

Remark. Remember that G_j denotes the set of all values a categorical predictor X_j can take. Then $\mathcal{B}_j$ is a set of unordered tuples, where each of the tuples contains a non-empty subset of G_j and its complement.

Finding a best binary partition in terms of minimal residual sum of squares is generally computationally infeasible. Instead, we use a greedy algorithm. Starting with all the data samples, consider a *split predictor* X_j $(j \in \{1, \dots, k\})$. For a *split predictor* X_j and *split point* s define $R_1(j,s)$ and $R_2(j,s)$ as

$$R_1(j,s) := \begin{cases} \{\mathbf{x} \mid x_j \leq s\}, & \text{if } X_j \text{ is a continuous predictor,} \\ \left\{\mathbf{x} \mid x_j \in B_j^{s;1}\right\}, & \text{if } X_j \text{ is a categorical predictor,} \end{cases} \tag{2.5}$$

$$R_2(j,s) := \begin{cases} \{\mathbf{x} \mid x_j > s\}, & \text{if } X_j \text{ is a continuous predictor,} \\ \left\{\mathbf{x} \mid x_j \in B_j^{s;2}\right\}, & \text{if } X_j \text{ is a categorical predictor,} \end{cases}$$

where $s \in \mathbb{R}$ for continuous variable and $s \in \{1, \dots, J_j\}$ for categorical variable and x_j denotes the j-th element of $\mathbf{x}$.

It is obvious that for every pair (j,s), $\{R_1(j,s); R_2(j,s)\}$ form a partition of $\mathcal{D}$. For classical *regression trees*, the index j of *split predictor* X_j and *split point* s are chosen as the solution of the optimization problem

$$\min_{j;s}\left(\min_{\beta_1,\dots,\beta_m} \sum_{i=1}^{n} (y_i - f_i)^2\right). \tag{2.6}$$

For every pair (j,s), the inner minimization of (2.6) is solved by (2.4). If for continuous predictors we restrict ourselves to the set of values taken by training data, the outer minimization can be simply solved by iterating through all the predictors and possible values of s. Finding the best split, we divide the data into two regions R_1 and R_2 and repeat the splitting process on both the regions. The whole process is then repeated again and again on all of the resulting regions, until the minimal *node size* (i.e. the number of observations in a *node*).

Remark. Note that we have found *an* optimal partition, not *the* optimal partition. Due to the fact that we have a finite number of observations, for the partition obtained in the previous paragraph, there is an (actually uncountable) infinite number of partitions minimizing (2.5), if for continuous predictors we do not restrict ourselves just to the set of values taken by our training data.

2.4. Pruning a Regression Tree

A very large *tree* might overfit the data, while a *tree* which is too small might not capture the important structure of the data. In this section, we will describe how to grow an optimal sized *tree*. Theorems stated in this section are stated and proved in [8], nevertheless the proofs of Theorem 2.10 and Theorem 2.11 were modified and more clarified by the author.

Definition 2.3. A *subtree* of a *tree* T is a *tree* T_S with *root* a *node* of T, such that each *node* of T_S is also a *node* of T. It is denoted as: $T_S \subseteq T$. Then T_S is called a *rooted subtree* of a *tree* T, if its *root* is the *root* of T.

Remark. If not specified otherwise or being clear from the context, the term *subtree* means the maximum possible *subtree* (in terms of the number of *nodes*).

Definition 2.4. A *branch* T_B at a *non-terminal node* t of a *tree* T is the *subtree* rooted at one of its *child nodes*.

Definition 2.5. The number of *leaves* of a *tree* T is called the *size* of a *tree*, i.e.

$$\operatorname{size}(T) := \operatorname{card}(\{t \mid t \text{ is a leaf of } T\}).$$

Instead of simply finding a *tree* with the minimal residual sum of squares, we need to consider also the *size* of the *tree*. For a given *tree* T and real number α, define the cost complexity criterion as

$$R_\alpha(T) := R(T) + \alpha \operatorname{size}(T), \tag{2.7}$$

where $R(T)$ is the residual sum of squares as defined in (2.2).

Consider growing a large *tree* T_0, stopping the splitting process only when some minimum *node size* (sample size at the specific *node*) is reached. The idea is to find for a given α, a *subtree* $T(\alpha) \subseteq T_0$ minimizing (2.7). Note that for $\alpha \le 0$, the solution is the *full tree* T_0. We will show that for a given *tree* T_0, there is a nested sequence of *subtrees* $\{T_k\}_{k=0}^q$ (i.e. $T_q \subseteq \cdots \subseteq T_0$) and an increasing sequence of real numbers $\{\alpha_k\}_{k=1}^q$ such that for $k = 1, \dots, q-1$, T_k is an *optimal tree* for $\alpha \in [\alpha_k; \alpha_{k+1})$, T_q is an *optimal tree* for $\alpha \ge \alpha_q$ and T_0 is an *optimal tree* for $\alpha < \alpha_1$. We will also provide an algorithm on how to construct the nested sequence $\{T_k\}_{k=0}^q$.

Remark. Consider a *node* t of a *tree* T. Values $R_\alpha(t), R(t)$ and $\operatorname{size}(t)$ are defined in the sense of $R_\alpha(T_{t_0}), R(T_{t_0})$ and $\operatorname{size}(T_{t_0})$, where T_{t_0} is a *trivial subtree* (consisting only of the *root*) rooted at the *node* t. It is easy to see that $\operatorname{size}(t) = 1$.

Remark. For a *node* t of a tree T, denote T_t the *subtree* of T rooted at t.

Definition 2.6. Consider a *tree* T. The reduction function $g(t, T)$ for a *non-terminal node* t of a *tree* T is defined as

$$g(t, T) := \frac{R(t) - R(T_t)}{\operatorname{size}(T_t) - \operatorname{size}(t)}. \tag{2.8}$$

Definition 2.7. *Pruning* of a *tree* T at a *non-terminal node* t is to replace T_t by t.

Lemma 2.8. *Consider a tree T and a non-terminal node t. Then for a given $\alpha \in \mathbb{R}$*

$$g(t,T) > \alpha \text{ if and only if } R_\alpha(t) > R_\alpha(T_t).$$

Proof. The proof follows directly from definitions (2.7) and (2.8). □

Remark. For a *non-terminal node* t of a *tree* T it follows that:

$$R_\alpha(T_t) = \sum_{\{T_B | T_B \text{ is a branch at } t\}} R_\alpha(T_B).$$

Theorem 2.9. *Consider a tree T and $\alpha \in \mathbb{R}$. Suppose that we visit all the nodes of T in the bottom-up order (i.e. starting from the leaves, finishing with the root, and visiting each node before its parent) and prune at a non-terminal node t only if*

$$R_\alpha(t) \leq R_\alpha(T'_t), \tag{2.9}$$

for the current tree T'. Then the resulting tree is $T(\alpha)$.

Proof. The proof will be done using mathematical induction. Suppose that when a *node* t is considered, all the *branches* at t are optimally pruned. It is obvious that this is true for the *leaves*. At *node* t, we either prune it with the value $R_\alpha(t)$, or not with the value $R_\alpha(T'_t)$ if this is strictly smaller. If there is a *subtree* T''_t rooted at t with a smaller value of R_α, it must be non-trivial and there must be a *branch* T_B with $R_\alpha(T''_B) < R_\alpha(T'_B)$ and so T'_B is not optimally pruned, which is a contradiction. Thus after a *node* t is considered, T'_t is optimally pruned. After the *root* is considered, the resulting *tree* is optimally pruned, hence it is $T(\alpha)$. □

Remark. Theorem 2.9 gives us an algorithm to find $T(\alpha)$ for a single $\alpha \in \mathbb{R}$.

Theorem 2.10. *Consider a tree T. Denote*

$$\alpha_1 := \min_{\{t | t \text{ is a non-terminal node of } T\}} g(t,T). \tag{2.10}$$

The optimally pruned tree is T for $\alpha < \alpha_1$ and $T_1 := T(\alpha_1)$ is obtained by pruning at all the nodes t with $g(t,T) = \alpha_1$. Further, $g(t,T_1) > \alpha_1$ for all non-terminal nodes t of T_1.

Proof. Assume $\alpha < \alpha_1$. From (2.10) it follows that $g(t,T) \geq \alpha_1 > \alpha$, for all *non-terminal nodes* t, thus $R_\alpha(t) > R_\alpha(T_t)$ (see Lemma 2.8). The optimality of T is then a direct consequence of Theorem 2.9. Now consider $\alpha = \alpha_1$ and pruning by Theorem 2.9. Since $g(t,T) \geq \alpha_1$, then also $R_\alpha(t) \geq R_\alpha(T_t)$ for all *non-terminal nodes* t, hence pruning is only applied at all *nodes* t with $g(t,T) = \alpha_1$ (see condition (2.9)). Thus the second part of the theorem is proved. As a consequence, whenever the *tree* is pruned, $R_\alpha(T'_s)$ is unchanged for all the *nodes* s of the new *tree*. Since there were no *nodes* with $R_{\alpha_1}(t) < R_{\alpha_1}(T_t)$ and all the *nodes* with $R_{\alpha_1}(t) = R_{\alpha_1}(T_t)$ were pruned, all the remaining *non-terminal nodes* satisfy $R_{\alpha_1}(t) > R_{\alpha_1}(T_{1t})$, which, according to Lemma 2.8 is equivalent to $g(t,T_1) > \alpha_1$ for all *non-terminal nodes* t of T_1. □

Theorem 2.11. *For $\beta > \alpha$, $T(\beta)$ is a subtree of $T(\alpha)$ and is the result of β-pruning of $T(\alpha)$.*

Proof. To show that $T(\beta)$ is a *subtree* of $T(\alpha)$, it is sufficient to show that $T_t'^{\beta}$ is a *subtree* of $T_t'^{\alpha}$ for every *node* t during β-pruning and α-pruning, respectively. It is obviously true at the *leaves*. At *node* t we compare $R_\alpha(t)$ to $R_\alpha\big(T_t'^{\alpha}\big)$ ($R_\beta(t)$ to $R_\beta\big(T_t'^{\beta}\big)$) and prune the *tree* if the first is weakly smaller. We must show that if $R_\alpha(t) \le R_\alpha\big(T_t'^{\alpha}\big)$ then $R_\beta(t) \le R_\beta\big(T_t'^{\beta}\big)$. Suppose that $R_\alpha(t) \le R_\alpha\big(T_t'^{\alpha}\big)$. During α-pruning, the current *tree* $T_t'^{\alpha}$ is optimal at each step, thus also $R_\alpha\big(T_t'^{\alpha}\big) \le R_\alpha\big(T_t'^{\beta}\big)$ at each step. Then

$$\begin{aligned}
R_\beta(t) &\overset{(2.7)}{=} R(t) + \beta \operatorname{size}(t) \overset{(2.7)}{=} R_\alpha(t) + (\beta-\alpha)\operatorname{size}(t)\\
&\le R_\alpha\big(T_t'^{\alpha}\big) + (\beta-\alpha)\operatorname{size}(t) \le R_\alpha\big(T_t'^{\beta}\big) + (\beta-\alpha)\operatorname{size}(t)\\
&\overset{(2.7)}{=} R\big(T_t'^{\beta}\big) + \alpha \operatorname{size}\big(T_t'^{\beta}\big) + (\beta-\alpha)\operatorname{size}(t)\\
&\overset{(2.7)}{=} R_\beta\big(T_t'^{\beta}\big) + (\alpha-\beta)\operatorname{size}\big(T_t'^{\beta}\big) + (\beta-\alpha)\operatorname{size}(t)\\
&= R_\beta\big(T_t'^{\beta}\big) - (\beta-\alpha)\Big(\operatorname{size}\big(T_t'^{\beta}\big) - \operatorname{size}(t)\Big)\\
&\overset{\beta>\alpha \,\&\, \operatorname{size}\left(T_t'^{\beta}\right)\ge\operatorname{size}(t)}{\le} \qquad R_\beta\big(T\_t'^{\beta}\big).
\end{aligned}$$

Thus $T(\beta)$ is a *subtree* of $T(\alpha)$. Since $T(\beta)$ minimizes $R_\beta(T')$ over all the *rooted subtrees* T' of T and is a *subtree* of $T(\alpha)$, it also minimizes $R_\beta(T')$ over all the *rooted subtrees* T' of $T(\alpha)$. □

The algorithm from Theorem 2.10 can be applied to the new *tree* $T_1 := T(\alpha_1)$ to find $\alpha_2 > \alpha_1$ and $T_2 := T(\alpha_2)$ and so on until T_q is the *trivial tree* (i.e. the *root* of $T_0 := T$). From Theorem 2.10 and Theorem 2.11 it follows that for $\alpha_1 \le \alpha < \alpha_2$, $T(\alpha) = T_1$ and $T(\alpha_2) = T_2$. Repeating the process, $\alpha_1 < \alpha_2 < \cdots < \alpha_q$ and $T_0 \supset T_1 \supset \cdots \supset T_q$ are obtained such that $T(\alpha) = T_i$ for $\alpha_i \le \alpha < \alpha_{i+1}$ $(i = 1, \ldots, q-1)$. Finally, the *full tree* T_0 is optimal for $\alpha < \alpha_1$ (see Theorem 2.10) and the *trivial tree* T_q is optimal for $\alpha \ge \alpha_q$, which follows from Theorem 2.11.

Algorithm 2.12. The following algorithm can be used to find for a given *tree* T, the *tree* sequence $\{T_k\}_{k=0}^{q}$ and $\{\alpha_k\}_{k=1}^{q}$ as described above:

1. Set $k := 0$ and $T_0 := T$.
2. Visit all the *non-terminal nodes* t in bottom-up order, compute $g(t, T_k)$ using (2.8) and set α_{k+1} as
$$\alpha_{k+1} := \min_{\{t \mid t \text{ is a non-terminal node of } T_k\}} g(t, T_k).$$
3. Visit all the *nodes* in top-down order and prune whenever $g(t, T_k) = \alpha_{k+1}$.
4. Set $T_{k+1} := T_k{}'$, where T_k' is the *tree* obtained by α_{k+1}-pruning of T_k.
5. If T_{k+1} is a *non-trivial tree*, set $k := k+1$ and go to Step 2.

2.5. k-Fold Cross-validation

Assume using Algorithm 2.12 to obtain a sequence of nested *trees* $\{T_k\}_{k=0}^{q}$, each *tree* being optimal for some $\alpha \in I_\alpha \subseteq \mathbb{R}$. Our goal is to choose the best *subtree* for our data. We already know that T_0 might overfit the data, but the *trivial tree* T_q is probably not the right model either. For choosing the best *subtree*, *k-fold cross-validation* can be used (more information regarding this topic can be found in [4]).

Consider a problem as described in Section 2.3. Denote $\mathcal{A}$ the set of training data, i.e.

$$\mathcal{A} := \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}.$$

Consider splitting $\mathcal{A}$ into k "roughly equal-sized" non-empty subsets $A_1, \dots, A_k$ such that

$$\bigcup_{i=1}^{k} A_i = \mathcal{A} \wedge A_i \cap A_j = \emptyset;\ i \neq j \wedge \max_{i,j\in\{1,\dots,k\}} \{|\mathrm{card}\,(A_i) - \mathrm{card}\,(A_j)|\} \leq 1.$$

We define an indexing function κ as

$$\kappa : \{1, \dots, n\} \to \{1, \dots, k\}, \\ i \mapsto j \mid \mathbf{x}_i \in A_j.$$

Denote by $\hat{f}^j(x)$ the fitted model using data from the set: $\mathcal{A} \setminus A_j$. The *cross-validation* estimate of prediction error is defined as

$$CV\big(\hat{f}\big) := \frac{1}{n}\sum_{i=1}^{n} L\big(y_i, \hat{f}^{\kappa(i)}(\mathbf{x}_i)\big),$$

where $L\big(y, \hat{f}(\mathbf{x})\big)$ is the loss function measuring errors between the observed value y and the predicted value $\hat{f}(\mathbf{x})$. Typical choices for the loss function are *squared error*

$$L\big(y, \hat{f}(\mathbf{x})\big) := \big(y - \hat{f}(\mathbf{x})\big)^2,$$

and an *absolute error*

$$L\big(y, \hat{f}(\mathbf{x})\big) := \big|y - \hat{f}(\mathbf{x})\big|.$$

The case $k = n$ is called the *leave-one-out cross-validation*. In this case κ is an identity function (i.e. $\kappa(i) = i$), and for the i-th observation, the fit is computed using all the data samples except the i-th.

Consider a set of models $f(\mathbf{x}, \alpha)$ indexed by a tuning parameter α. Denote by $\hat{f}^p(\mathbf{x}, \alpha)$ the α-th model fit on sample $\mathcal{A} \setminus A_p$. Define the *cross-validation estimate* of prediction error for this set of models as

$$CV\big(\hat{f}, \alpha\big) := \frac{1}{n}\sum_{i=1}^{n} L\big(y_i, \hat{f}^{\kappa(i)}(\mathbf{x}_i, \alpha)\big). \tag{2.11}$$

Our goal is to find parameter $\widehat{\alpha}$ minimizing (2.11). Our final chosen model is then $f\big(\mathbf{x}, \widehat{\alpha}\big)$, which is fitted to all the data.

2.6. Confidence Intervals for the Expected Value

Consider a problem as described in Section 2.2. Consider a model of the form (2.1). Assume that $\varepsilon_1, \dots, \varepsilon_n$ are independent. Then also $Y_1, \dots, Y_n$, each Y_i being a function of ε_i, are independent. Defining an indexing function τ as

$$\begin{aligned} \tau \colon \mathcal{D} &\to \{1, \dots, m\}, \\ \mathbf{x} &\mapsto j \mid \mathbf{x} \in R_j, \end{aligned} \tag{2.12}$$

we can write (2.1) as

$$Y_i = f(\mathbf{x}_i) + \varepsilon_i = \beta_{\tau(\mathbf{x}_i)} + \varepsilon_i. \tag{2.13}$$

Theorem 2.13. *For $\widehat{\beta}_j$ (where $j = 1, \dots, m$) as defined in (2.4), it follows that*

$$\mathrm{E}\left(\widehat{\beta}_j\right) = \beta_j, \tag{2.14}$$

$$\operatorname{var}\left(\widehat{\beta}_j\right) = \frac{\sigma^2}{\mathcal{C}_j}. \tag{2.15}$$

Proof. Firstly we need to compute the expected value and variance of Y_i (where $i = 1, \dots, n$)

$$\mathrm{E}(Y_i) \overset{(2.13)}{=} \mathrm{E}\left(\beta_{\tau(\mathbf{x}_i)} + \varepsilon_i\right) = \mathrm{E}\left(\beta_{\tau(\mathbf{x}_i)}\right) + \mathrm{E}(\varepsilon_i) \overset{\mathrm{E}(\varepsilon_i)=0}{=} \beta_{\tau(\mathbf{x}_i)}, \tag{2.16}$$

$$\operatorname{var}(Y_i) \overset{(2.13)}{=} \operatorname{var}\left(\beta_{\tau(\mathbf{x}_i)} + \varepsilon_i\right) = \operatorname{var}(\varepsilon_i) \overset{\operatorname{var}(\varepsilon_i)=\sigma^2}{=} \sigma^2. \tag{2.17}$$

Using (2.4), both the desired equalities can be computed directly

$$\begin{aligned} \mathrm{E}\left(\widehat{\beta}_j\right) &= \mathrm{E}\left(\frac{1}{\mathcal{C}_j} \sum_{\{i \mid \mathbf{x}_i \in R_j\}} Y_i\right) = \frac{1}{\mathcal{C}_j} \sum_{\{i \mid \mathbf{x}_i \in R_j\}} \mathrm{E}(Y_i) \overset{(2.16)}{=} \frac{1}{\mathcal{C}_j} \sum_{\{i \mid \mathbf{x}_i \in R_j\}} \beta_{\tau(\mathbf{x}_i)} \\ &\overset{(2.12)}{=} \frac{1}{\mathcal{C}_j} \sum_{\{i \mid \mathbf{x}_i \in R_j\}} \beta_j = \frac{1}{\mathcal{C}_j} \mathcal{C}_j \beta_j = \beta_j, \\ \operatorname{var}\left(\widehat{\beta}_j\right) &= \operatorname{var}\left(\frac{1}{\mathcal{C}_j} \sum_{\{i \mid \mathbf{x}_i \in R_j\}} Y_i\right) = \frac{1}{\mathcal{C}_j^2} \operatorname{var}\left(\sum_{\{i \mid \mathbf{x}_i \in R_j\}} Y_i\right) \\ &\overset{Y_1, \dots, Y_n \text{ are independent}}{=} \frac{1}{\mathcal{C}_j^2} \sum_{\{i \mid \mathbf{x}_i \in R_j\}} \operatorname{var}(Y_i) \overset{(2.17)}{=} \frac{1}{\mathcal{C}_j^2} \sum_{\{i \mid \mathbf{x}_i \in R_j\}} \sigma^2 \\ &= \frac{1}{\mathcal{C}_j^2} \mathcal{C}_j \sigma^2 = \frac{\sigma^2}{\mathcal{C}_j}. \end{aligned}$$

□

Remark. According to (2.14), $\widehat{\beta}_j$ is an *unbiased* estimator of β_j.

Remark. Denote the residual sum of squares as

$$S_e := \sum_{i=1}^{n} \left(Y_i - \widehat{\beta}_{\tau(\mathbf{x}_i)}\right)^2 = \sum_{j=1}^{m} \sum_{\{i \mid \mathbf{x}_i \in R_j\}} \left(Y_i - \widehat{\beta}_j\right)^2. \tag{2.18}$$

Remark. Further on, define $\mathbf{1} := (1, \ldots, 1)^T$, $\mathbf{0} := (0, \ldots, 0)^T$ and denote $\mathbf{I}$ as the identity matrix. If the size of $\mathbf{I}$ ($\mathbf{1}$ or $\mathbf{0}$) is not clear from the context, we will use $\mathbf{I}_q$ ($\mathbf{1}_q$ or $\mathbf{0}_q$), where the lower index q indicates the matrix (vector) of size q.

Denote $\mathbf{Y}_j := \left(Y_{j_1}, \ldots, Y_{j\mathcal{C}_j}\right)^T$ the vector of all Y_i's, for which $\mathbf{x}_i \in R_j$. From (2.16), (2.17) and independence of Y_i's, it follows that

$$\mathrm{E}\left(\mathbf{Y}_j\right) = \beta_j \mathbf{1}, \tag{2.19}$$

$$\operatorname{var}\left(\mathbf{Y}_j\right) = \sigma^2 \mathbf{I}. \tag{2.20}$$

Using (2.4), it is easy to see that

$$\widehat{\beta}_j = \frac{1}{\mathcal{C}_j} \mathbf{1}^T \mathbf{Y}_j. \tag{2.21}$$

Remark. Notice that

$$\sum_{j=1}^{m} \mathcal{C}_j = n. \tag{2.22}$$

Lemma 2.14. *Define $\mathbf{M}_j$ as*

$$\mathbf{M}_j := \mathbf{I}_{\mathcal{C}_j} - \frac{1}{\mathcal{C}_j} \mathbf{1}_{\mathcal{C}_j} \mathbf{1}_{\mathcal{C}_j}^T. \tag{2.23}$$

Then $\mathbf{M}_j$ is symmetric, idempotent and it follows that

$$\mathbf{1}_{\mathcal{C}_j}^T \mathbf{M}_j = \mathbf{0}^T. \tag{2.24}$$

Proof. The proof of symmetry is trivial. Idempotence follows from

$$\begin{aligned} \mathbf{M}_j \mathbf{M}_j &= \mathbf{I}_{\mathcal{C}_j} - \frac{2}{\mathcal{C}_j} \mathbf{1}_{\mathcal{C}_j} \mathbf{1}_{\mathcal{C}_j}^T + \frac{1}{\mathcal{C}_j^2} \mathbf{1}_{\mathcal{C}_j} \mathbf{1}_{\mathcal{C}_j}^T \mathbf{1}_{\mathcal{C}_j} \mathbf{1}_{\mathcal{C}_j}^T \\ &\overset{\mathbf{1}_{\mathcal{C}_j}^T \mathbf{1}_{\mathcal{C}_j} = \mathcal{C}_j}{=} \mathbf{I}_{\mathcal{C}_j} - \frac{2}{\mathcal{C}_j} \mathbf{1}_{\mathcal{C}_j} \mathbf{1}_{\mathcal{C}_j}^T + \frac{1}{\mathcal{C}_j^2} \mathbf{1}_{\mathcal{C}_j} \mathcal{C}_j \mathbf{1}_{\mathcal{C}_j}^T = \mathbf{M}_j. \end{aligned}$$

Equation (2.24) can be obtained simply by direct computation. Indeed

$$\mathbf{1}_{\mathcal{C}_j}^T \mathbf{M}_j = \mathbf{1}_{\mathcal{C}_j}^T - \frac{1}{\mathcal{C}_j} \mathbf{1}_{\mathcal{C}_j}^T \mathbf{1}_{\mathcal{C}_j} \mathbf{1}_{\mathcal{C}_j}^T \overset{\mathbf{1}_{\mathcal{C}_j}^T \mathbf{1}_{\mathcal{C}_j} = \mathcal{C}_j}{=} \mathbf{1}_{\mathcal{C}_j}^T - \frac{1}{\mathcal{C}_j} \mathcal{C}_j \mathbf{1}_{\mathcal{C}_j}^T = \mathbf{0}^T.$$

□

Theorem 2.15. *Define random variable s^2 as*

$$s^2 := \frac{S_e}{n - m}, \tag{2.25}$$

where S_e was defined in (2.18). Then s^2 is an unbiased estimate of σ^2, i.e.

$$\mathrm{E}\left(s^2\right) = \sigma^2. \tag{2.26}$$

Proof. Firstly we compute $\operatorname{E}(S_e)$ as follows

$$\begin{aligned}
\operatorname{E}(S_e) &\overset{(2.18)}{=} \operatorname{E}\left(\sum_{j=1}^{m}\sum_{\{i|\mathbf{x}_i\in R_j\}}\left(Y_i-\widehat{\beta}_j\right)^2\right)\\
&\overset{(2.21)}{=} \operatorname{E}\left(\sum_{j=1}^{m}\sum_{\{i|\mathbf{x}_i\in R_j\}}\left(Y_i-\frac{1}{\mathcal{C}_j}\mathbf{1}^T\mathbf{Y}_j\right)^2\right)\\
&= \sum_{j=1}^{m}\operatorname{E}\left(\left(\mathbf{Y}_j-\frac{1}{\mathcal{C}_j}\mathbf{1}\mathbf{1}^T\mathbf{Y}_j\right)^T\left(\mathbf{Y}_j-\frac{1}{\mathcal{C}_j}\mathbf{1}\mathbf{1}^T\mathbf{Y}_j\right)\right)\\
&\overset{(2.23)}{=} \sum_{j=1}^{m}\operatorname{E}\left(\mathbf{Y}_j^T\mathbf{M}_j^T\mathbf{M}_j\mathbf{Y}_j\right) \overset{\text{Lemma 2.14}}{=} \sum_{j=1}^{m}\operatorname{E}\left(\mathbf{Y}_j^T\mathbf{M}_j\mathbf{Y}_j\right)\\
&\overset{\text{Theorem 4.18 in [1]}}{=} \sum_{j=1}^{m}\Big(\operatorname{Tr}\left(\mathbf{M}_j\operatorname{var}\left(\mathbf{Y}_j\right)\right)+\left(\operatorname{E}\left(\mathbf{Y}_j\right)\right)^T\mathbf{M}_j\operatorname{E}\left(\mathbf{Y}_j\right)\Big)\\
&\overset{(2.19)\,\&\,(2.20)}{=} \sum_{j=1}^{m}\Big(\operatorname{Tr}\left(\mathbf{M}_j\sigma^2\mathbf{I}\right)+\beta_j\mathbf{1}^T\mathbf{M}_j\beta_j\mathbf{1}\Big)\\
&\overset{(2.24)}{=} \sum_{j=1}^{m}\sigma^2\operatorname{Tr}\left(\mathbf{M}_j\right) \overset{(2.23)}{=} \sigma^2\sum_{j=1}^{m}\left(\operatorname{Tr}\left(\mathbf{I}_{\mathcal{C}_j}-\frac{1}{\mathcal{C}_j}\mathbf{1}_{\mathcal{C}_j}\mathbf{1}_{\mathcal{C}_j}^T\right)\right)\\
&= \sigma^2\sum_{j=1}^{m}\left(\mathcal{C}_j-1\right) \overset{(2.22)}{=} \sigma^2\left(n-m\right),
\end{aligned}\tag{2.27}$$

Dividing (2.27) by $(n-m)$, (2.26) is obtained. □

Further on in this section, assume that for $i=1,\ldots,n$, ε_i are normally distributed (i.e., $\varepsilon_i\sim N\left(0,\sigma^2\right)$, where the expected value and variance of ε_i was defined in Section 2.2). Then also Y_i are normally distributed, for $i=1,\ldots,n$.

Lemma 2.16. *Matrix $\mathbf{M}_j$ (see (2.23)) is positive semi-definite.*

Proof. Multiplying $\mathbf{M}_j$ by vector $\mathbf{c}$ (of length $\mathcal{C}_j$) from left and right, we obtain

$$\begin{aligned}
\mathbf{c}^T\mathbf{M}_j\mathbf{c} &\overset{(2.23)}{=} \mathbf{c}^T\mathbf{c}-\frac{1}{\mathcal{C}_j}\mathbf{c}^T\mathbf{1}_{\mathcal{C}_j}\mathbf{1}_{\mathcal{C}_j}^T\mathbf{c}\\
&\overset{\mathbf{1}_{\mathcal{C}_j}^T\mathbf{1}_{\mathcal{C}_j}=\mathcal{C}_j}{=} \frac{1}{\mathcal{C}_j}\left(\mathbf{c}^T\mathbf{c}\cdot\mathbf{1}_{\mathcal{C}_j}^T\mathbf{1}_{\mathcal{C}_j}-\left(\mathbf{c}^T\mathbf{1}_{\mathcal{C}_j}\right)^2\right)\geq 0,
\end{aligned}$$

where the last inequality is a well-known *Cauchy-Schwarz* inequality for $\mathbb{R}^{\mathcal{C}_j}$

$$\left|\langle\mathbf{u},\mathbf{v}\rangle\right|^2\leq\langle\mathbf{u},\mathbf{u}\rangle\langle\mathbf{v},\mathbf{v}\rangle;\quad\forall\,\mathbf{u},\mathbf{v}\in\mathbb{R}^{\mathcal{C}_j},$$

where $\langle\mathbf{u},\mathbf{v}\rangle$ denotes the inner product of vectors $\mathbf{u},\mathbf{v}$ (in this case, $\mathbf{u}=\mathbf{c}$ and $\mathbf{v}=\mathbf{1}_{\mathcal{C}_j}$). □

Theorem 2.17. *Consider s^2 as defined in (2.25). Assume that $\sigma^2 > 0$ and $\mathcal{C}_j \geq 2$, for $j = 1, \ldots, m$. Then*

$$\frac{(n-m)\, s^2}{\sigma^2} \sim \chi^2_{n-m} \, .$$

Proof. Define random variable Z_i as

$$Z_i := \frac{Y_i - \beta_{\tau(\mathbf{x}_i)}}{\sigma}. \tag{2.28}$$

It is easy to see that

$$\frac{1}{\mathcal{C}_j} \sum_{\{i|\mathbf{x}_i \in R_j\}} Z_i = \frac{1}{\mathcal{C}_j} \sum_{\{i|\mathbf{x}_i \in R_j\}} \frac{(Y_i - \beta_j)}{\sigma} \overset{(2.4)}{=} \frac{\widehat{\beta}_j - \beta_j}{\sigma} \, . \tag{2.29}$$

Denote $\mathbf{Z}_j := \left(Z_{j_1}, \ldots, Z_{j_{\mathcal{C}_j}}\right)^T$ the vector of all Z_i's, for which $\mathbf{x}_i \in R_j$. From (2.16), (2.17) and Z_i being a linear combination of Y_i, which is normally distributed, it follows that $Z_i \sim N\,(0,1)$ (for $i = 1, \ldots, n$). Finally, since Y_i's are independent, Z_i's are also independent, thus $\mathbf{Z}_j \sim N\left(\mathbf{0}_{\mathcal{C}_j}, \mathbf{I}_{\mathcal{C}_j}\right)$. Furthermore, not only Z_i's are independent, but $\mathbf{Z}_p$ and $\mathbf{Z}_q$ are also independent for $p \neq q$. Then

$$\begin{aligned}
\frac{(n-m)\, s^2}{\sigma^2} &\overset{(2.25)\,\&\,(2.18)}{=} \frac{1}{\sigma^2} \sum_{j=1}^{m} \sum_{\{i|\mathbf{x}_i \in R_j\}} \left(Y_i - \widehat{\beta}_j\right)^2 \\
&= \sum_{j=1}^{m} \sum_{\{i|\mathbf{x}_i \in R_j\}} \left(\frac{(Y_i - \beta_j) - (\widehat{\beta}_j - \beta_j)}{\sigma}\right)^2 \\
&\overset{(2.28)\,\&\,(2.29)}{=} \sum_{j=1}^{m} \sum_{\{i|\mathbf{x}_i \in R_j\}} \left(Z_i - \frac{1}{\mathcal{C}_j} \sum_{\{k|\mathbf{x}_k \in R_j\}} Z_k\right)^2 \\
&= \sum_{j=1}^{m} \left(\mathbf{Z}_j - \frac{1}{\mathcal{C}_j} \mathbf{1}\mathbf{1}^T \mathbf{Z}_j\right)^T \left(\mathbf{Z}_j - \frac{1}{\mathcal{C}_j} \mathbf{1}\mathbf{1}^T \mathbf{Z}_j\right) \\
&\overset{\text{Lemma 2.14}}{=} \sum_{j=1}^{m} \mathbf{Z}_j^T \mathbf{M}_j \mathbf{Z}_j .
\end{aligned} \tag{2.30}$$

From Lemma 2.14 and Lemma 2.16, it follows that M_j is symmetric, idempotent, non-zero and positive semi-definite, hence by Theorem 4.16 in [1] we obtain

$$\mathbf{Z}_j^T \mathbf{M}_j \mathbf{Z}_j \sim \chi^2_{\mathrm{Tr}(\mathbf{M}_j \mathbf{I})},$$

which yields

$$\mathbf{Z}_j^T \mathbf{M}_j \mathbf{Z}_j \sim \chi^2_{\mathcal{C}_j - 1}. \tag{2.31}$$

Finally, using Theorem 4.13 in [1], independence of $\mathbf{Z}_i$'s, (2.22), (2.30) and (2.31), we obtain

$$\frac{(n-m)\, s^2}{\sigma^2} = \sum_{j=1}^{m} \mathbf{Z}_j^T \mathbf{M}_j \mathbf{Z}_j \sim \chi^2_{n-m}.$$

□

Theorem 2.18. *Assume that $\sigma^2 > 0$ and $\mathcal{C}_j \geq 2$, for $j = 1, \ldots, m$. Then $\widehat{\beta}_j$ and s^2 are independent.*

Proof. Using (2.30), we can write s^2 as

$$s^2 = \frac{\sigma^2}{n-m} \sum_{j=1}^{m} \mathbf{Z}_j^T \mathbf{M}_j \mathbf{Z}_j \overset{(2.28)}{=} \sum_{j=1}^{m} \left(\mathbf{Y}_j - \beta_j \mathbf{1}\right)^T \frac{1}{n-m} \mathbf{M}_j \left(\mathbf{Y}_j - \beta_j \mathbf{1}\right).$$

From (2.19), (2.20) and normality of Y_i's it follows that $\mathbf{Y}_j \sim N\left(\beta_j \mathbf{1}, \sigma^2 \mathbf{I}\right)$. Denote $\mathbf{B} := \frac{1}{\mathcal{C}_j} \mathbf{1}_{\mathcal{C}_j}^T$. Then

$$\mathbf{B}\, \mathrm{var}\left(\mathbf{Y}_j\right) \left(\frac{1}{n-m} \mathbf{M}_j\right) = \frac{\sigma^2}{\mathcal{C}_j \left(n-m\right)} \mathbf{1}^T \mathbf{I} \mathbf{M}_j \overset{(2.24)}{=} \mathbf{0}^T. \tag{2.32}$$

Since $\frac{1}{n-m} \mathbf{M}_j$ is positive semi-definite (see Lemma 2.16) and using (2.32), we obtain from Theorem 4.19 in [1] that $\left(\mathbf{Y}_j - \beta_j \mathbf{1}\right)^T \frac{1}{n-m} \mathbf{M}_j \left(\mathbf{Y}_j - \beta_j \mathbf{1}\right)$ and $\widehat{\beta}_j = \mathbf{B}\mathbf{Y}_j$ are independent. Moreover, $\left(\mathbf{Y}_i - \beta_i \mathbf{1}\right)^T \frac{1}{n-m} \mathbf{M}_i \left(\mathbf{Y}_i - \beta_i \mathbf{1}\right)$ and $\widehat{\beta}_j$ are independent for $i \neq j$, since Y_i's are independent. Finally, since s^2 is a sum of m independent random variables, where each of them is independent with $\widehat{\beta}_j$, then s^2 and $\widehat{\beta}_j$ are also independent. □

Theorem 2.19. *Consider $\tilde{x} \in \mathcal{D}$. Then*

$$\frac{\widehat{\beta}_{\tau(\tilde{x})} - \beta_{\tau(\tilde{x})}}{s} \sqrt{\mathcal{C}_{\tau(\tilde{x})}} \sim t_{n-m},$$

where t_{n-m} is a Student's t-distribution with $n-m$ degrees of freedom.

Proof. It follows from Theorem 2.13 and $\widehat{\beta}_j$ being normally distributed, that

$$\widehat{\beta}_{\tau(\tilde{x})} \sim N\left(\beta_{\tau(\tilde{x})}, \frac{\sigma^2}{\mathcal{C}_{\tau(\tilde{x})}}\right),$$

thus

$$\frac{\widehat{\beta}_{\tau(\tilde{x})} - \beta_{\tau(\tilde{x})}}{\sigma} \sqrt{\mathcal{C}_{\tau(\tilde{x})}} \sim N\left(0,1\right). \tag{2.33}$$

Using Theorem 4.22 in [1], (2.33), Theorem 2.17 and Theorem 2.18, we obtain

$$\frac{\frac{\widehat{\beta}_{\tau(\tilde{\mathbf{x}})} - \beta_{\tau(\tilde{\mathbf{x}})}}{\sigma} \sqrt{\mathcal{C}_{\tau(\tilde{\mathbf{x}})}}}{\sqrt{\frac{(n-m)s^2}{\sigma^2}}} \sqrt{n-m} = \frac{\widehat{\beta}_{\tau(\tilde{\mathbf{x}})} - \beta_{\tau(\tilde{\mathbf{x}})}}{s} \sqrt{\mathcal{C}_{\tau(\tilde{\mathbf{x}})}} \sim t_{n-m}.$$

□

Remark. Using Theorem 2.19, we can construct the $(1-\alpha)$-confidence interval for $\mathrm{E}\left(\tilde{Y}\right)$, where $\tilde{Y}$ is an independent future observation associated with $\tilde{\mathbf{x}}$

$$\left(\widehat{\beta}_{\tau(\tilde{\mathbf{x}})} - t_{n-m}\left(\alpha\right) \frac{s}{\sqrt{\mathcal{C}_{\tau(\tilde{\mathbf{x}})}}}; \widehat{\beta}_{\tau(\tilde{\mathbf{x}})} + t_{n-m}\left(\alpha\right) \frac{s}{\sqrt{\mathcal{C}_{\tau(\tilde{\mathbf{x}})}}}\right),$$

where for a random variable $T \sim t_p$, value $t_p(\alpha)$ is defined as $\left(1-\frac{\alpha}{2}\right)$-quantile, i.e.

$$P\{|T| \geq t_p(\alpha)\} = \alpha. \tag{2.34}$$

Remark. In Theorem 2.17 and Theorem 2.18, we assumed that $\mathcal{C}_j \geq 2$, which is quite reasonable, since we do not want to grow *trees* with *leaves* containing only one observation. Such *trees* would be probably overfitted and we would need to use e.g. *k-fold cross-validation* (see Section 2.5) in order to obtain an optimal *subtree*.

2.7. Maximum Likelihood Estimation for Gamma Distribution of the Response Variable

In Section 2.3, for a given partition of $\mathcal{D}$, the values of $\beta_1, \ldots, \beta_m$ were estimated by minimizing the residual sum of squares (see (2.2)). If the response variable has gamma distribution, we might use another approach, maximum likelihood estimation. Maximum likelihood is described in detail in [2] in the context of *generalized linear models*. The crucial difference between least squares and maximum likelihood is that instead of minimizing a loss function, we estimate the values of unknown parameters in such way, that amongst all the possible considered models, the probability of making such observations as we have done, is the greatest.

Let $Y_i \sim \Gamma(\alpha, \beta_i)$, for $i = 1, \ldots, n$, where, $\beta_1, \ldots, \beta_n$ are the parameters of interest and $\alpha > 0$ is regarded as a nuisance parameter. Consider k predictors and denote $\mathbf{x}_i := (x_{i1}, \ldots, x_{ik})^T \in \mathcal{D}$, for $i = 1, \ldots, n$. Consider a partition of $\mathcal{D}$ into m regions $R_1, \ldots, R_m$. We are not interested in parameters $\beta_1, \ldots, \beta_n$ directly, but instead we estimate parameters $\mu_1, \ldots, \mu_m$, where for $j = 1, \ldots, m$ it follows that

$$\mathrm{E}(Y_i) = \mu_j; \quad \forall i \mid \mathbf{x}_i \in R_j. \tag{2.35}$$

According to [1], for $Y_i \sim \Gamma(\alpha, \beta_i)$ it follows that

$$\mathrm{E}(Y_i) = \frac{\alpha}{\beta_i}. \tag{2.36}$$

Combining (2.35) and (2.36), we obtain

$$\beta_i = \frac{\alpha}{\mu_j}; \quad \forall i \mid \mathbf{x}_i \in R_j;\ j = 1, \ldots, m. \tag{2.37}$$

Using this notation, our model of interest (compare with (2.1)) is now of the form

$$Y_i = \sum_{j=1}^{m} \mu_j \cdot \chi_{R_j}(\mathbf{x}_i) + \varepsilon_i;\ i = 1, \ldots, n,$$

The probability density function $f_i := f_i(y_i; \beta_i)$ of Y_i (see [1]) is of the form

$$f_i = \exp\left(-y_i\beta_i + \alpha \ln \beta_i + (\alpha - 1)\ln y_i - \ln \Gamma(\alpha)\right), \tag{2.38}$$

where $\Gamma(\alpha)$ is the gamma function.

Remark. Do not confuse the notation f_i with the notation used in Section 2.3. In this subsection, f_i denotes the probability density function of Y_i.

Denote $\mathbf{y} = (y_1, \ldots, y_n)^T$ and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_n)^T$. Assuming that $Y_1, \ldots, Y_n$ are independent, we can write the joint log-likelihood function as

$$\begin{aligned}
\ell := \ell(\mathbf{y}; \boldsymbol{\mu}) &= \ln f(\mathbf{y}; \boldsymbol{\mu}) \overset{Y_1,\ldots,Y_n \text{ are independent}}{=} \ln\left(\prod_{i=1}^{n} f_i\right) = \sum_{i=1}^{n} \ln f_i \\
&\overset{(2.38)}{=} \sum_{i=1}^{n} \ln \exp\big(-y_i\beta_i + \alpha \ln \beta_i + (\alpha-1)\ln y_i - \ln\Gamma(\alpha)\big) \\
&= \sum_{i=1}^{n} \big(-y_i\beta_i + \alpha \ln \beta_i + (\alpha-1)\ln y_i - \ln\Gamma(\alpha)\big) \qquad (2.39) \\
&= -n\ln\Gamma(\alpha) + \sum_{j=1}^{m} \sum_{\{i|\mathbf{x}_i \in R_j\}} (-y_i\beta_i + \alpha\ln\beta_i + (\alpha-1)\ln y_i) \\
&\overset{(2.37)}{=} -n\ln\Gamma(\alpha) + \sum_{j=1}^{m} \sum_{\{i|\mathbf{x}_i \in R_j\}} \left(-y_i\frac{\alpha}{\mu_j} + \alpha\ln\frac{\alpha}{\mu_j} + (\alpha-1)\ln y_i\right).
\end{aligned}$$

Our goal is to find the values of $\mu_1, \ldots, \mu_m$ maximizing the expression (2.39). Differentiating both sides of (2.39) with respect to μ_u and setting it to 0, we obtain

$$\begin{aligned}
\frac{\partial \ell}{\partial \mu_u} &= \sum_{\{i|\mathbf{x}_i \in R_u\}} \frac{\partial}{\partial \mu_u}\left(-y_i\frac{\alpha}{\mu_u} + \alpha\ln\frac{\alpha}{\mu_u} + (\alpha-1)\ln y_i\right) \\
&= \sum_{\{i|\mathbf{x}_i \in R_u\}} \left(y_i\frac{\alpha}{\mu_u^2} + \alpha\frac{\mu_u}{\alpha}\left(-\frac{\alpha}{\mu_u^2}\right)\right) = \frac{\alpha}{\mu_u^2}\sum_{\{i|\mathbf{x}_i \in R_u\}} (y_i - \mu_u) \qquad (2.40) \\
&= \frac{\alpha}{\mu_u^2}\left(\sum_{\{i|\mathbf{x}_i \in R_u\}} y_i - \mathcal{C}_u\mu_u\right) = 0,
\end{aligned}$$

where $\mathcal{C}_u$ has the same meaning as in Section 2.3. Assuming that $\{i \mid \mathbf{x}_i \in R_u\}$ is non-empty, we obtain a unique solution of (2.40)

$$\widehat{\mu}_u = \frac{1}{\mathcal{C}_u} \sum_{\{i|\mathbf{x}_i \in R_u\}} y_i\,. \qquad (2.41)$$

Similarly as in Section 2.3.1, it is an easy exercise to verify that (2.41) maximizes (2.39), by verifying that the Hessian matrix of (2.39) is negative definite. Details of these computations can be found in [7].

Notice that (2.41) is formally the same as (2.4). For a given partition of $\mathcal{D}$ and for a future independent observation $\tilde{Y}$ (associated with $\tilde{\mathbf{x}}$), the predicted value, $\mu_j \mid \tilde{\mathbf{x}} \in R_j$, is the same as it was for the least squares estimator derived in Section 2.3.

The difference between the least squares minimization approach and maximum likelihood estimation is how the *split predictor* X_j and *split point* s are obtained. Instead of solving (2.6), different optimization problem is considered

$$\max_{j;s}\left(\max_{\mu_1,\ldots,\mu_m} \ell(\mathbf{y}; \boldsymbol{\mu})\right), \qquad (2.42)$$

where the meaning of j and s was explained in Section 2.3.2 and $\boldsymbol{\mu} := (\mu_1, \dots, \mu_m)$. For every pair (j, s), the inner maximization of (2.42) is solved by (2.41). Similarly as in Section 2.3.2, the outer maximization can be simply computed by iterating through all the predictors and possible values of s and choosing the pair with the greatest value of (2.39) evaluated at the point $\widehat{\boldsymbol{\mu}} := (\widehat{\mu}_1, \dots, \widehat{\mu}_n)^T$ computed according to (2.41). If we assume $\alpha > 0$ to be fixed, it is obvious that maximizing the value of (2.39) evaluated at the point $\widehat{\boldsymbol{\mu}}$ for different partitions $\{R_1(j_q, s_q); R_2(j_q, s_q)\}$ (for q being from some finite index set) is the same as maximizing the expression

$$\begin{aligned}
&\sum_{j=1}^{m} \sum_{\{i|\mathbf{x}_i \in R_j\}} \left(-y_i \frac{\alpha}{\mu_j} + \alpha \ln \frac{\alpha}{\mu_j} + (\alpha - 1) \ln y_i \right) \Bigg|_{\boldsymbol{\mu} = \widehat{\boldsymbol{\mu}}} \\
&\qquad = (\alpha - 1) \sum_{j=1}^{m} \sum_{\{i|\mathbf{x}_i \in R_j\}} \ln y_i + n\alpha \ln \alpha \\
&\qquad\quad - \alpha \sum_{j=1}^{m} \left(\frac{1}{\widehat{\mu}_j} \sum_{\{i|\mathbf{x}_i \in R_j\}} y_i + \mathcal{C}_j \ln \widehat{\mu}_j \right) \\
&\qquad \overset{(2.41)\ \&\ (2.22)}{=} (\alpha - 1) \sum_{i=1}^{n} \ln y_i + n\alpha \ln \alpha - n\alpha - \alpha \sum_{j=1}^{m} \mathcal{C}_j \ln \widehat{\mu}_j \,.
\end{aligned} \tag{2.43}$$

We can see that the first three terms of (2.43) do not depend on the partition of $\mathcal{D}$, hence for fixed $\alpha > 0$, the outer maximization of (2.42) is solved by computing and comparing the values of

$$\ell^* := -\sum_{j=1}^{m} \mathcal{C}_j \ln \widehat{\mu}_j \tag{2.44}$$

for all the pairs (j, s) and selecting the pair with the greatest value of (2.44).

Remark. The outer maximization of (2.42) can be replaced by minimization of so called *deviance*, which provides exactly the same results. Details of this approach can be found in [7].

3. Random Forests

Regression trees as described in Section 2 have some disadvantages, e.g. instability or the chance of overfitting the data easily. These issues can be overcome by *random forests*, which improve the predictions in terms of variance, by simply growing multiple *regression trees*, each of them on a slightly different dataset (as we will see in this chapter). Firstly, we will briefly introduce the basic concepts of bootstrap and bagging, which will be important for growing *random forests*.

3.1. Bootstrapping and Bagging

This section provides only a brief introduction into the bootstrapping and bagging. More detailed information regarding this topic can be found in [4].

Consider a set of training data $\mathcal{A} := \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$. Denote $S(\mathcal{A})$ some quantity computed from data $\mathcal{A}$ (e.g. a prediction at some input point based on some model). The basic idea of bootstrapping is to randomly draw datasets with replacement from the set of training data, each sample the same size as the original one. This is done B times, producing B bootstrapped datasets. Then the quantity $S(\cdot)$ is computed from each of the B bootstrapped datasets. According to [4], using bootstrap sampling, we can compute the estimation $\widehat{\text{var}}(S(\mathcal{A}))$ of the variance of $S(\mathcal{A})$ as

$$\widehat{\text{var}}(S(\mathcal{A})) = \frac{1}{B-1} \sum_{b=1}^{B} \left(S\left(\mathcal{A}^b\right) - \bar{S}\right)^2, \tag{3.1}$$

where $\mathcal{A}^b$ denotes the b-th bootstrapped dataset from $\mathcal{A}$ and $\bar{S}$ is defined as

$$\bar{S} := \frac{1}{B} \sum_{b=1}^{B} S\left(\mathcal{A}^b\right).$$

Now we will show how to use bootstrap to improve the prediction itself. Suppose we fit a model $\hat{f}$ to our training data $\mathcal{A}$ and obtain a prediction $\hat{f}(\tilde{\mathbf{x}})$ at input $\tilde{\mathbf{x}}$. Bagging averages this prediction over B bootstrapped samples. For b-th bootstrapped sample $\mathcal{A}^b$, we fit our model to obtain prediction $\hat{f}^b(\tilde{\mathbf{x}})$. The bagging estimate is defined as

$$\hat{f}_{bag}(\tilde{\mathbf{x}}) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^b(\tilde{\mathbf{x}}).$$

When bagging is applied on *regression trees*, each bootstrapped *tree* might contain different *splits* and might have a different number of *leaves*. The bagged estimate is the average prediction at $\tilde{\mathbf{x}}$ over all the B *trees*.

Bagging can significantly reduce the variance of unstable procedures like *regression trees* (see [4]). Applying bagging on *regression trees*, any simple structure is lost. Hence interpretability is lost as a result of improving the stability of the model. Instability of a *regression tree* means that a small change in the training data can result in a very different series of *splits*, hence providing quite different results.

The key difference between bootstrapping and bagging is that bootstrapping is a random sampling with replacement, and bagging is performing the bootstrap multiple times and training an estimator for each of the bootstrapped data and then aggregating the predictions to make a final prediction. So bootstrapping is a sampling technique, whereas bagging, also known as bootstrap aggregation, is a machine learning ensemble technique designed to improve accuracy of statistical models used for prediction and classification.

3.2. Introduction to Random Forests

In Section 3.1, basic idea of bagging was introduced, where we fit a *regression tree* many times to bootstrapped samples of the training data and average the results. *Random forests* technique is a modification of bagging, that builds a large collection of *trees*, while reducing correlation amongst them at the same time.

An important feature of *trees* obtained by bagging is that they are identically distributed (i.d.), since each *tree* is built on a data sample randomly drawn with replacement from the same population. As a result, the expectation and variance of each *tree* is the same, so also the bias of bagged *trees* (i.e., squared distance between predicted and real values) is unchanged in comparison to the individual *trees*. Hence the only improvement can be through variance reduction of bagged *trees* when compared to individual *trees*.

Theorem 3.1. *Let $X_1, \dots, X_n$ be i.d. random variables (i.e.,* $\operatorname{var}(X_i) = \sigma^2$*, for $i = 1, \dots, n$). Assume that* $\operatorname{cor}(X_i, X_j) = \rho \geq 0$*, for $i \neq j$ and $i, j = 1, \dots, n$. Denote $\bar{X}_n$ the average of $\mathbf{X} := (X_1, \dots, X_n)^T$, i.e.*

$$\bar{X}_n := \frac{1}{n}\sum_{i=1}^{n} X_i = \frac{1}{n}\mathbf{1}^T\mathbf{X}. \tag{3.2}$$

Then variance of $\bar{X}_n$ is of the form

$$\operatorname{var}\left(\bar{X}_n\right) = \rho\sigma^2 + \frac{1-\rho}{n}\sigma^2. \tag{3.3}$$

Proof. The variance-covariance matrix of $\mathbf{X}$ is of the form

$$\operatorname{var}(\mathbf{X}) = \begin{pmatrix} \sigma^2 & \rho\sigma^2 & \dots & \rho\sigma^2 \\ \rho\sigma^2 & \sigma^2 & \dots & \rho\sigma^2 \\ \vdots & & \ddots & \\ \rho\sigma^2 & \rho\sigma^2 & \dots & \sigma^2 \end{pmatrix} = \rho\sigma^2\mathbf{1}\mathbf{1}^T + \sigma^2(1-\rho)\mathbf{I}. \tag{3.4}$$

Using basic properties of variance, we obtain

$$\begin{aligned} \operatorname{var}\left(\bar{X}_n\right) &\overset{(3.2)}{=} \operatorname{var}\left(\frac{1}{n}\mathbf{1}^T\mathbf{X}\right) = \frac{1}{n^2}\operatorname{var}\left(\mathbf{1}^T\mathbf{X}\right) \\ &\overset{\operatorname{var}(\mathbf{AX})=\mathbf{A}\operatorname{var}(\mathbf{X})\mathbf{A}^T}{=} \frac{1}{n^2}\mathbf{1}^T\operatorname{var}(\mathbf{X})\mathbf{1} \\ &\overset{(3.4)}{=} \frac{\sigma^2}{n^2}\mathbf{1}^T\left(\rho\mathbf{1}\mathbf{1}^T + (1-\rho)\mathbf{I}\right)\mathbf{1} \\ &= \frac{\sigma^2}{n^2}\left(\rho\mathbf{1}^T\mathbf{1}\mathbf{1}^T\mathbf{1} + (1-\rho)\mathbf{1}^T\mathbf{1}\right) \\ &\overset{\mathbf{1}^T\mathbf{1}=n}{=} \frac{\sigma^2}{n^2}\left(n^2\rho + n(1-\rho)\right) = \rho\sigma^2 + \frac{1-\rho}{n}\sigma^2. \end{aligned}$$

□

Remark. Notice that if X_i's are independent (i.e., $\rho = 0$) in Theorem 3.1, we obtain $\operatorname{var}\left(\bar{X}_n\right) = \frac{1}{n}\sigma^2$, which is a well-known property of variance of a mean of independent, identically distributed (i.i.d.) random variables.

Remark. An important result obtained from Theorem 3.1 is

$$\lim_{n\to\infty}\operatorname{var}\left(\bar{X}_n\right) \overset{(3.3)}{=} \rho\sigma^2 \leq \sigma^2 = \operatorname{var}(X_i)\,;\ i \in \mathbb{N},$$

since $0 \leq \rho \leq 1$.

Knowing that *regression trees* obtained by bagging are i.d. and assuming that correlation between the *trees* is constant and less than 1, we can decrease the variance of a single *regression tree* model by growing a sufficiently large number of *trees*. The basic idea of *random forests* is to improve the variance reduction obtained by bagging by reducing the correlation between *trees* without increasing their variance too much. This is achieved by selecting a random subset of $m \leq k$ predictors before each *split* when growing a *tree*.

Algorithm 3.2. Consider a set of training data $\mathcal{A} := \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i = (x_{i1}, \ldots, x_{ik})^T$, for $i = 1 \ldots, n$. Each of the k predictors is either continuous or categorical. The following algorithm can be used to grow a *random forest*.

1. For $b = 1$ to B:
   - (a) Draw a bootstrap data sample $\mathcal{A}^b$ of size n from original data $\mathcal{A}$.
   - (b) Grow a *regression tree* T_b on bootstrapped data $\mathcal{A}^b$, by recursively repeating the following steps for each *leaf* of the *tree*, until the minimum *node size* is reached for all the *leaves* of the current *tree*:
     - (i) Select a random subset of $m \leq k$ predictors from k original predictors.
     - (ii) Select the best *split predictor* and *split point* (see Section 2.3 and Section 2.7, respectively) from the selected m predictors.
     - (iii) Split the *node* into two *nodes* only if the resulting *nodes* satisfy the condition of minimum *node size*, and perform Steps 1(b)(i) –1(b)(iii) on both of the resulting *nodes*.
   - (c) If needed, perform pruning of the *tree* T_b (see Section 2.4) and use the *k-fold cross-validation* (see Section 2.5) to select the best *subtree* T_b', and set $T_b := T_b'$.
2. Output the ensemble of *regression trees* $\{T_b\}_{b=1}^{B}$.
3. Prediction at a point $\tilde{\mathbf{x}}$ is computed as

$$\bar{T}(\tilde{\mathbf{x}}) := \frac{1}{B} \sum_{b=1}^{B} T_b(\tilde{\mathbf{x}}),$$

   where $T_b(\tilde{\mathbf{x}})$ is a prediction at $\tilde{\mathbf{x}}$ obtained from b-th *tree*.

According to [4], recommended minimum *node size* is 5, and the recommended value for m in Step 1(b)(i) is $\lfloor \frac{k}{3} \rfloor$, where $\lfloor x \rfloor$ is the floor function defined as

$$\lfloor x \rfloor := \max\{m \in \mathbb{Z} \mid m \leq x\}.$$

In practice, the best values for both these parameters depend on the particular problem and can be considered as tuning parameters.

When growing a *regression tree*, only the most relevant predictors are selected for *splits*, whereas predictors that do not affect Y_i's very much are not selected. When the number of predictors k is large, but the number of relevant predictors is relatively very small when compared to k, *random forests* are not likely to perform good for small values of m, since there is small chance of a relevant predictor being selected at each *split*. Before each *split*, the probability of a relevant predictor being selected follows a hypergeometric distribution.

3.3. Confidence Intervals for the Expected Value

Consider a random sample $\mathcal{A}$ from an unknown distribution F. Consider a parameter of interest $\theta = t(F)$ and its estimate $\hat{\theta} = s(\mathcal{A})$. For $b = 1, \ldots, B$, consider a bootstrap dataset $\mathcal{A}^b$ from $\mathcal{A}$ and define a bootstrap replication of $\hat{\theta}$ as $\hat{\theta}^b := s\left(\mathcal{A}^b\right)$. Then according to [3], the bootstrap estimate $\hat{s}$ of the *standard error* of a *statistic* $\hat{\theta}$ can be computed as

$$\hat{s} = \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}\left(\hat{\theta}^b - \bar{\hat{\theta}}\right)^2}, \tag{3.5}$$

where $\bar{\hat{\theta}}$ is defined as

$$\bar{\hat{\theta}} := \frac{1}{B}\sum_{b=1}^{B}\hat{\theta}^b.$$

Remark. Notice the similarity between (3.1) and (3.5). Indeed, the *standard error* of a *statistic* is the estimation of the square root of variance of its sampling distribution.

Now, consider an independent future observation $\tilde{Y}$ associated with $\tilde{\mathbf{x}}$. The value $T_b(\tilde{\mathbf{x}})$ (defined in Step 3 of Algorithm 3.2) can be considered as a bootstrap replication of $\mathrm{E}\left(\tilde{Y}\right)$, and from (3.5), the estimate $\hat{s}$ of the *standard error* of $\mathrm{E}\left(\tilde{Y}\right)$ can be computed as

$$\hat{s} = \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}\left(T_b(\tilde{\mathbf{x}}) - \bar{T}(\tilde{\mathbf{x}})\right)^2},$$

where $\bar{T}(\tilde{\mathbf{x}})$ was defined also in Step 3.

Using bootstrap, we can obtain confidence intervals without making any assumptions about data. For each of the B bootstrapped data samples we compute

$$Z_b := \frac{T_b(\tilde{\mathbf{x}}) - \bar{T}(\tilde{\mathbf{x}})}{s_b},$$

where s_b is the estimated *standard error* for the b-th *tree* and can be computed as (also compare with (2.15))

$$s_b = \sqrt{\frac{s^2}{\mathcal{C}_{\tau(\tilde{\mathbf{x}})}}},$$

where s^2 was defined in (2.25). Consider a random variable Z

$$Z = \frac{\bar{T}(\tilde{\mathbf{x}}) - \mathrm{E}\left(\tilde{Y}\right)}{\hat{s}}.$$

According to [3], the $(1-\alpha)$-quantile of Z can be estimated by the value $\hat{t}\left(\alpha\right)$, which satisfies the following condition

$$\frac{\operatorname{card}\left(\left\{b \mid Z_b \leq \hat{t}\left(\alpha\right)\right\}\right)}{B} = \alpha. \tag{3.6}$$

If $B\alpha$ is not an integer, then e.g. interpolation can be used to compute $\hat{t}\left(\alpha\right)$. Finally, using (3.6), the bootstrap-t $(1-\alpha)$-confidence interval for $\mathrm{E}\left(\tilde{Y}\right)$ can be computed as

$$\left(\bar{T}\left(\tilde{\mathbf{x}}\right) - \hat{s}\,\hat{t}\left(1-\frac{\alpha}{2}\right); \bar{T}\left(\tilde{\mathbf{x}}\right) - \hat{s}\,\hat{t}\left(\frac{\alpha}{2}\right)\right).$$

Another approach to compute confidence intervals is using the empirical distribution function. Define the empirical $(1-\alpha)$-quantile $\hat{t}_e\left(\alpha\right)$ as the value satisfying

$$\frac{\operatorname{card}\left(\left\{b \mid T_b\left(\tilde{\mathbf{x}}\right) \leq \hat{t}_e\left(\alpha\right)\right\}\right)}{B} = \alpha. \tag{3.7}$$

From (3.7), the empirical $(1-\alpha)$-confidence interval for $\mathrm{E}\left(\tilde{Y}\right)$ can be computed simply as

$$\left(\hat{t}_e\left(\frac{\alpha}{2}\right); \hat{t}_e\left(1-\frac{\alpha}{2}\right)\right).$$

Remark. Notice a slight difference in notations of different quantiles defined in this article. In (2.34), the value $t_p\left(\alpha\right)$ denotes $\left(1-\frac{\alpha}{2}\right)$-quantile, whilst in (3.6) and (3.7), values $F_{m,n}\left(\alpha\right)$, $\hat{t}\left(\alpha\right)$ and $\hat{t}_e\left(\alpha\right)$ denote $(1-\alpha)$-quantiles. This is because of the symmetry of *Student's t*-distribution. Since reader might be used to different notation, we'd rather emphasize the notation used in this article.

4. Conclusion

Section 2 provides an introduction to the concept of *regression trees*. Two principles, least squares and maximum likelihood estimation, of growing the *trees* are described and confidence intervals for the expected value were also derived. Introduction to *regression trees* can be found in [4] and more detailed information can be found in [8], according to which Section 2.4 was written. Proofs of theorems in Section 2.4 can also be found in [8], but were slightly modified (especially the proof of Theorem 2.10), since the author was not completely satisfied by the proofs provided in [8]. Since we could not find literature covering the topic of confidence intervals from Section 2.6 to the desired extent, the results in this section were derived by the author, following the derivation of confidence intervals for the expected value for *linear regression* from [1], since the process was quite similar. Also, for the same reason, the maximum likelihood estimation with the assumption of gamma distribution of the response variable of regression trees from Section 2.7 was derived by the author, inspired by the process of derivation of *generalized linear models* described in [2].

Section 3 deals with *random forests*. Although this chapter might seem to be short when compared to the previous one, the opposite is true. Most of the required apparatus for *random forests* has been already derived in Section 2, since *random forests* use *regression trees* together with bootstrapping and bagging to reduce the variance of single *regression trees* and provide even more accurate predictions. Confidence intervals for the expected value were also derived for *random forests*. The introduction to *random forests* (Section 3.1 and Section 3.2 can be found in [4]). Confidence intervals from Section 3.3 were application of confidence intervals for bootstrap described in [3].

Application of described statistical models (i.e. *regression trees* and *random forests*) on real data can be found in [7], as well as comparison of those two models with *linear regression* and *generalized linear models*. Due to skewness of the distribution of our data, we decided to derive a maximum likelihood estimation for regression trees (see Section 2.7), assuming a response variable with gamma distribution. This gamma distribution reflects skewness of the distribution of our data in [7], which was also reflected in the final results, since maximum likelihood method for *regression trees* provided better results when compared to ordinary least squares (see Section 2.3) (and also the best results amongst all the studied statistical models).

Reference

[1] J. Anděl: *Základy matematické statistiky*, MatfyzPress, Praha, 2011.

[2] A. J. Dobson, A. G. Barnett: *An Introduction to Generalized Linear Models*, Chapman and Hall/CRC, New York, 2008.

[3] B. Efron, R. Tibshirani: *An Introduction to the Bootstrap*, Chapman and Hall/CRC, New York, 1993.

[4] T. Hastie, R. Tibshirani, J. Friedman: *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, New York, 2016.

[5] P. Hebák, J. Hustopecký I. Malá: *Vícerozměrné statistické metody [2]* Informatorium, Praha, 2005.

[6] P. McCullagh, J. A. Nelder: *Generalized Linear Models*, Chapman and Hall/CRC, New York, 1989.

[7] D. Oberta: *Statistical Models for Prediction of Project Duration*, Bachelor's Thesis; Faculty of Mechanical Engineering, Brno University of Technology, Brno, 2023.

[8] B. D. Ripley: *Pattern Recognition and Neural Networks*, Cambridge University Press, Cambridge, 1996.

Dušan Oberta, Ústav matematiky, Fakulta strojního inženýrství, Vysoké učení technické v Brně, Technická 2, 61669 Brno, Česká republika,

e-mail: `228584@vutbr.cz`