W7L1

We end this section with a remark regarding K-fold cross validation.

R Normally K-fold is applied for $K = 5$ or 10. If $K = n$ then the $n$-fold is also called "leave one out cross validation". Note that for large values of $n$ the $n$-fold is not computationally feasible.

LOOC

### Classification Trees

In classification, the main goal is to predict the labels; for instance, whether or not a loan defaults. In addition, we are also interested in proportions defined below:

The proportion of class $k$ observations in region $m$ represented by node $m$:

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} 1_{y_i = k}$$

The algorithm of classification tree is similar to regression tree, but instead of RSS, the following three criteria can be applied:

- Classification error rate: this is the proportions of training observations in node $m$ that do not belong to the most common class. The most common class is given by $k^\star = \underset{1 \le k \le K}{\operatorname{argmax}}(\hat{p}_{mk})$, and so the classification error rate is equal to $1 - \hat{p}_{mk^\star}$
- Gini index; it favors larger partitions and very simple to implement: $\sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$
- Cross-entropy or deviance favors partitions that have small counts but many distinct values:

$$-\sum_{k=1}^{K} \hat{p}_{mk} \ln(\hat{p}_{mk})$$

Both the Gini index and cross-entropy are more sensitive to the purity of a node than classification error rate. These criteria lead to CART models. Like any other machine learning algorithm, CART models have some advantages and disadvantages summarized below.

Advantages of tree models:

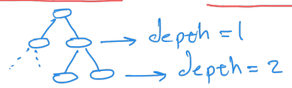
- Simple to understand, interpret, visualize.
- Decision trees implicitly perform variable screening or feature selection.
- These models can handle both numerical and categorical data. They can also handle multi-class classification problems.
- Decision trees require relatively little effort from users for data preparation.

Disadvantages of tree models

- Decision-tree learners can create over-complex trees that do not generalize the data well, i.e. they suffer from overfitting problems.

- Decision trees can be unstable because small variations in the data might result in a completely different tree being generated.
- The algorithm used to obtain optimal tree models cannot guarantee to return the globally optimal decision tree.
- Decision tree learners create biased trees if some classes dominate.

R There are several ways to reduce the overfitting problem of tree models for instance, pruning, enforcing a minimum number of samples in leaf nodes, or by enforcing a maximum depth for the tree.




Decision tree models provide powerful visualizations and interpretability. However, they suffer from overfitting and as a result their prediction power is normally not as good as other supervised learning methods. Ensemble methods could be used improve the forecasting power of decision tree types models, this is a topic that we will explore next.

# 6. Ensemble Machine Learning Methods

W7L2

Some machine learning methods are referred to as weak learner in the sense that their performance is slightly better than random guessing. Tree models are examples of weak learners though we should not forget their interpret-ability power. However, once we join a set of weak learner together, we might be able to improve the forecasting power of the algorithm. In this chapter, we review some frequently used ensemble methods. In particular, we discuss how to improve predictive power of tree models.

In order to understand the concepts, we provide a case study. The dataset to use in this case study is Loan Default Dataset: from "Salford Predictive Modeler®" subcompany of MINITAB:

"MINITAB® and all other trademarks and logos for the Company's products and services are the exclusive property of Minitab, LLC. All other marks referenced remain the property of their respective owners. See minitab.com for more information."

Please visit the following links:

`https://www.salford-systems.com/`

`http://www.minitab.com/en-us/`

Figure 6.1 visualize the dataset using a tree model.

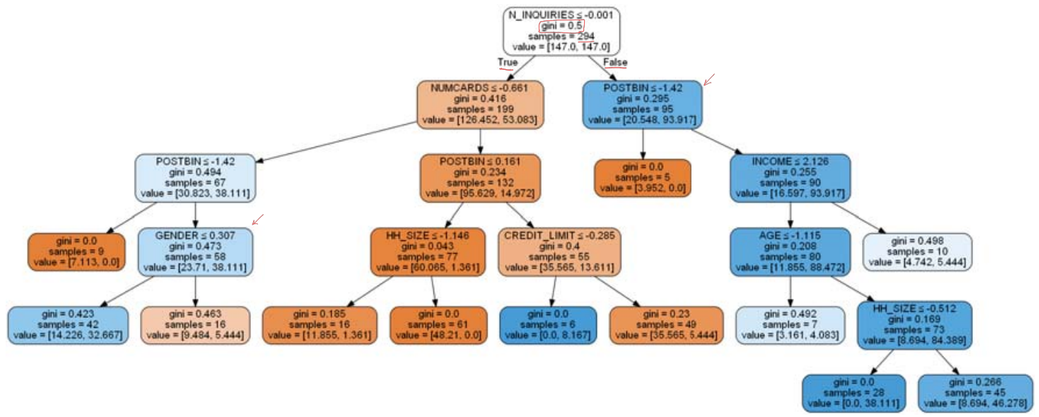



Figure 6.1: Credit Card Loan Dataset from Salford Predictive Modeler

## 6.1 Bootstrap

Bootstrap is a powerful statistical tool to measure uncertainty in statistical analysis. In particular, it is quite helpful in estimating standard errors. The following example (in portfolio analysis) highlights the idea.

■ **Example 6.1** Consider two assets with returns $R_A$ and $R_B$. Suppose that we invest $\alpha$ proportion of our wealth in asset $A$ and $1-\alpha$ in asset B. Find an expression for the optimal $\alpha$ that minimizes the variance of $R$ based on the variances and covariance of the two assets. Assume that we can generate new samples from the original population, discuss the estimation of $\alpha$ and the accuracy of the estimator. ■

$\alpha \longrightarrow A$, $1-\alpha \longrightarrow B$, $\alpha^* = \arg\min_\alpha Var(R)$

$R$ is the return of the investement: $R = \alpha R_A + (1-\alpha) R_B$

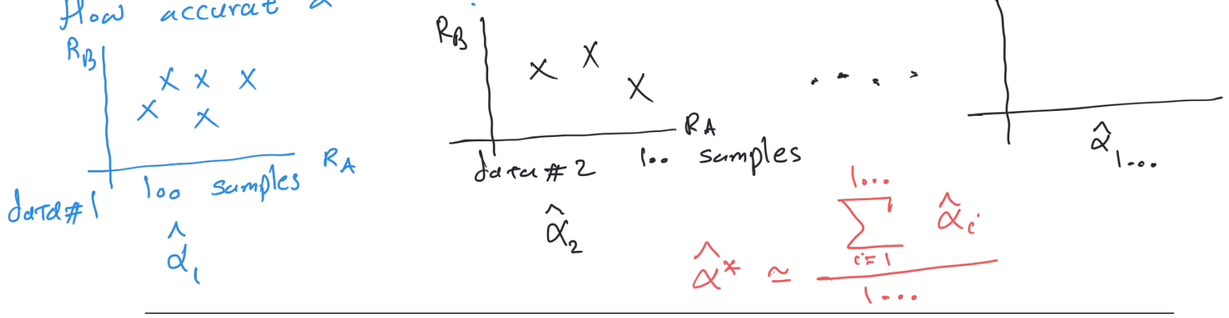
$Var(R) = Var(\alpha R_A + (1-\alpha) R_B)$, $\frac{d(Var(\alpha R_A + (1-\alpha) R_B))}{d\alpha} = 0$

$$\alpha^* = \frac{\sigma_A^2 - \sigma_{AB}}{\sigma_B^2 + \sigma_A^2 - 2\sigma_{AB}}, \quad \sigma_A^2 = Var(R_A), \ \sigma_B^2 = Var(R_B)$$

$\sigma_{AB} = cov(R_A, R_B)$

We can estimate $\alpha^*$ by: $\hat{\alpha}^* = \frac{\hat{\sigma}_A^2 - \hat{\sigma}_{AB}}{\hat{\sigma}_B^2 + \hat{\sigma}_A^2 - 2\hat{\sigma}_{AB}}$

How accurat $\hat{\alpha}^*$ is? And how to measure this?

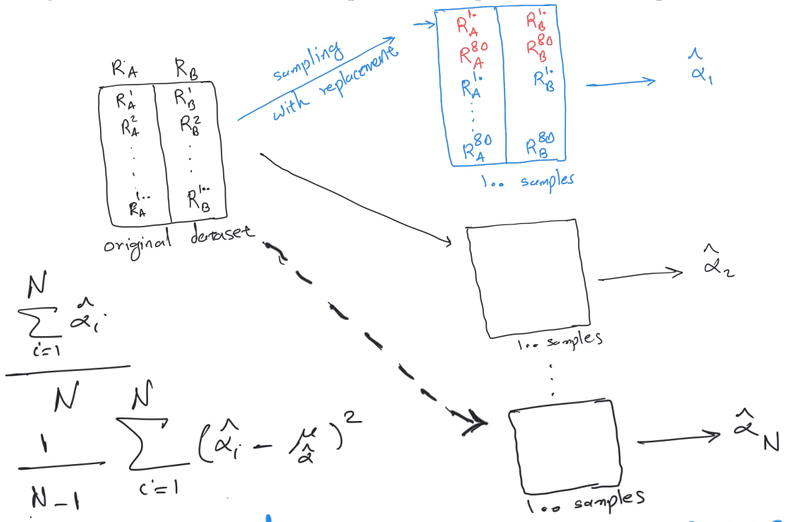


$\{\hat{\alpha}_1, \hat{\alpha}_2, \dots, \hat{\alpha}_{100}\}$ for $\hat{\alpha}^*$

Let say $\hat{\alpha}^* = 0.61$ & $SD(\hat{\alpha}^*) \simeq 0.07$, so roughly speaking, for a dataset, we expect that $\hat{\alpha}^*$ differ from the true $\alpha^*$ by 0.07 on average.



However, in real world problems, we might not have access to such huge (either simulated or real) datasets as assumed in the previous example, and hence the validity of the estimations could be questioned. For instance, this is the case if the assets are from an illiquid market. Remarkably, the Bootstrapping technique can be used in these cases to provide a more reliable estimation. In Bootstrapping, we obtain distinct copies of the dataset by repeatedly sampling (with replacement) from the original dataset. In the context of the previous example, the method is explained below:




$$\mu_{\hat{\alpha}} = \frac{\sum_{i=1}^{N} \hat{\alpha}_i}{N}$$

$$[SD_{\hat{\alpha}}]^2 = \frac{1}{N-1} \sum_{i=1}^{N} (\hat{\alpha}_i - \mu_{\hat{\alpha}})^2$$

Remark. Bootstrap is used in many cases where it is hard or even impossible to estimate the underlying random generator. Bootstrapping can reduce the variance of the estimator which means that we will obtain a more accurate estimator.

Idea. Suppose there $\{Z_i\}_{i \geq 1}$ are i.i.d r.v., $\bar{Z} = \left(\sum_{i=1}^{N} Z_i\right)/N$

$$Var(\bar{Z}) = Var\left(\frac{\sum_{i=1}^{N} Z_i}{N}\right) = \frac{1}{N^2} Var\left(\sum_{i=1}^{N} Z_i\right) = \frac{1}{N^2} \sum_{i=1}^{N} Var(Z_i)$$

$$= \frac{1}{N^2} \cdot N \cdot \sigma^2 = \frac{\sigma^2}{N} \Rightarrow Var(\bar{Z}) = \frac{\sigma^2}{N} \xrightarrow[N \uparrow \infty]{} 0$$

$Var(Z_1) = \sigma^2$

W7L3

## 6.2 Bagging

Bagging (also known as Bootstrap aggregation) is a methodology that uses Bootstrap to lower down the variance. This technique is in particular helpful to reduce the variance of tree-models which can lead to a better forecasting results. Let us explain the idea by a simple example as follows:

$f(x)$ : response or hypothesis function, suppose that we have B uncorrelated dataset for estimation:

$\hat{f}_1(x)$ ↑ dataset 1 $\quad \hat{f}_2(x)$ ↑ dataset 2 $\quad \hat{f}_B(x)$ ↑ dataset B

$$\Rightarrow \hat{f}(x) = \frac{1}{B}\sum_{j=1}^{B} \hat{f}_j(x)$$

Since the training datasets are not correlated then $\hat{f}$ has a lower variance.

However, in reality, we don't have access to multiple training datasets. Alternatively, we can generate B Bootstrapped training dataset from the original one:

$$\hat{f}(x) = \frac{1}{B}\sum_{j=1}^{B} \hat{f}_j^{(b)}(x)$$

$\hat{f}_j^{(b)}(x)$ → Bootstrapped estimator

This is called Bagging.

Let us apply Bagging for tree models:

For regression: $T_1, T_2, \ldots, T_B$, where each tree is obtained using Bootstrapped datasets. These trees have high variance and low bias, i.e. they have overfitting issues

$y_{T_1}$ $\quad y_{T_2}$ $\quad y_{T_B}$

$$\hat{y} = \frac{\sum_{i=1}^{B} y_{T_i}}{B}$$, reduces variance

For classification:

$T_1 \longrightarrow$ a predicted class: $C_1$,

$T_2 \longrightarrow C_2$

$\vdots$

$T_B \longrightarrow C_B$

$C_i \in \{1, 2, \ldots, K\}$

$i = 1, 2, \ldots, B$

Finally, the predicted class is mode by majority vote, i.e. the most frequent class in the predictions.

## 6.3 Appendix: Important Topics in Ensemble Methods

W7L4

### 6.3.1 Out of Bag Error Estimation

Out of Bag (OOB) error estimation is a method to perform error measurement without performing cross validation.

Let us consider bagging tree models: $T_i^{bag}$ for $1 \le i \le B$ is obtained using Bootstrapped data based on the original dataset. In Bootstrapping: the Bootstrapped dataset $\simeq \frac{2}{3}$ of the original dataset, AND $\frac{1}{3}$ of the original dataset (training dataset) is out of bag (OOB).

$T_1^{bag}$ ⋮ $T_B^{bag}$

Cross validation: For an given observation (or feature) $x$, we find those trees $T_i^{bag}$ for which $x$ is OOB, and we know that there are around $\frac{B}{3}$ trees that have not used $x$ in their training. These tree can be used to make a prediction either by averaging (for regression) or majority vote (for classification). Suppose that $E_1, E_2, \ldots, E_B$ are the errors of these predictions. Then we can calculate the MSE or classification errors.

This is a valid measurement of the error, as the response for each case is calculated by a tree that has not used the observation in its training.

### 6.3.2 Variable Importance

Tree models are easy to interpret. However, in bagging method, the results cannot be shown using a single tree, and hence they are not easy to interpret. Hence bagging increases the accuracy at the cost of interpret-ability. How can we come of with a measure of importance for a feature?

Consider bagging of trees:

$T_1^{bag} \longrightarrow D_1^x$

$\vdots$

$T_B^{bag} \longrightarrow D_B^x$

$D_1^x$: feature total decrease in RSS for regression (Gini or entropy for classification) due to splits of $x$

$$\text{feature importance} := \frac{\sum_{j=1}^{B} D_j^x}{B}$$