Ensemble Learning

Songpan Yang

School of Mathematics Sun Yat-sen University

Dec, 3, 2020

Table of Contents

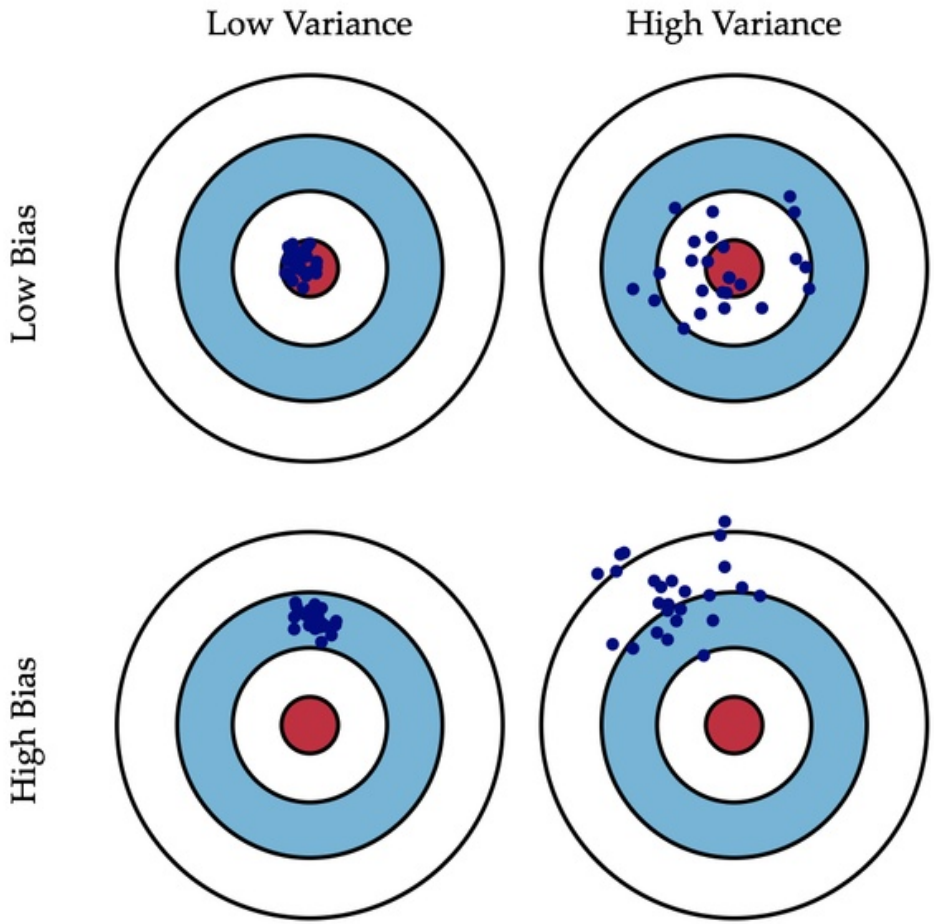
Table of Contents

Why do we study Ensemble Learning?

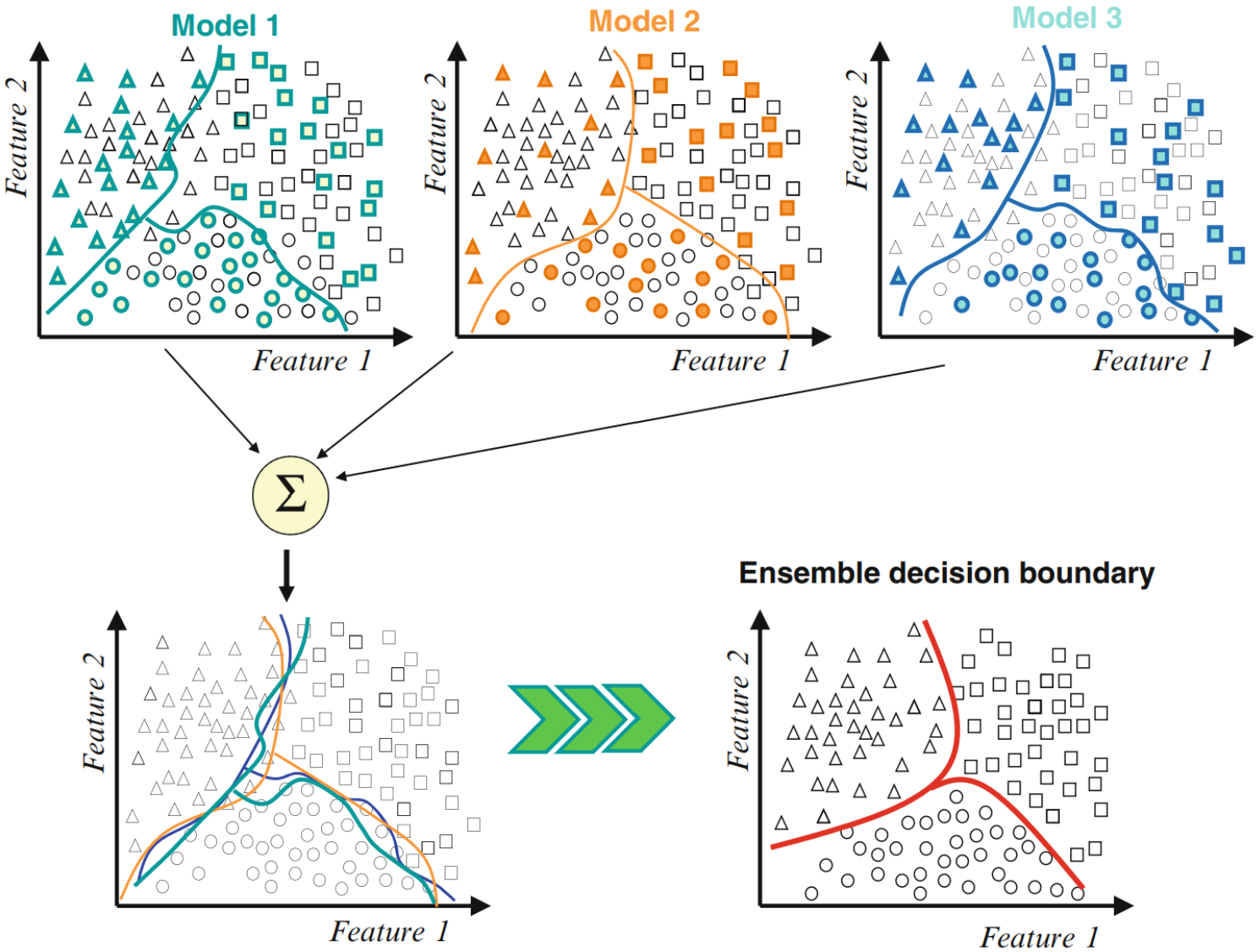
- We often consult others before making a decision
- An expert (or oracle) VS no expert
- The accuracy of each decision (classifier) has a nonzero variability.
- Two components of the any classification error: bias and variance
- The goal of ensemble systems:
  1. create several classifiers with relatively fixed (or similar) bias
  2. combining their outputs, to reduce the variance

Bias-variance tradeoff

- The true model (An expert or oracle): $y = f_{true}(x)$
- Suppose we have a model (classifier): $y = f(x) + \epsilon$
- $E\left(y - f(x)\right) = \sigma_\epsilon^2 + \text{Bias}^2\left(f\right) + \text{Var}\left(f\right)$



An example


Model 1
Model 2
Model 3
Feature 2
Feature 1
Σ
Ensemble decision boundary

The process of Ensemble Learning

1. Data sampling/selection
2. Training Member Classifiers: bagging, boosting, stacking
3. Combining Ensemble Members

Data sampling/selection

- Making different errors on any given sample
- Diversity in ensembles.
- The most common approach:
  using different subsets of the training data
  1. bootstrapped replicas of the training data
  2. sampling from a distribution that favors previously misclassified samples
  3. use different subsets of the available features to train each classifier
- Other less common approaches:
  1. using different parameters of the base classifier
  2. using different base classifiers as the ensemble members

Popular Ensemble-Based Algorithms

- Three popular algorithms: Bagging, Boosting, Stacking
- Denote the classifier f_i, the weight of classifier w_i, the correlation coefficient of each classifier ρ
- The Full model be the linear combination of basic classifiers of Bagging and Boosting

$$E(F) = E\left(\sum_i^m w_i f_i\right) = \sum_i^m w_i E\left(f_i\right) = w\sum_i^m E\left(f_i\right)$$

$$\begin{aligned} Var(F) &= Var\left(\sum_i^m w_i f_i\right) = Cor\left(\sum_i^m w_i f_i, \sum_i^m w_i f_i\right) \\ &= \sum_i^m w_i^2 Var(f_i) + \sum_i^m \sum_{j\neq i}^m 2\rho w_i w_j \sqrt{Var(f_i)Var(f_j)} \\ &= m^2 w^2 \sigma^2 \rho + m w^2 \sigma^2 (1-\rho) \end{aligned}$$

bagging and boosting

- For bagging:

$$E(F) = w \sum_{i}^{m} E\left(f_i\right) = \frac{1}{m} m\mu$$

$$Var(F) = \sigma^2 \rho + \frac{\sigma^2 (1-\rho)}{m}$$

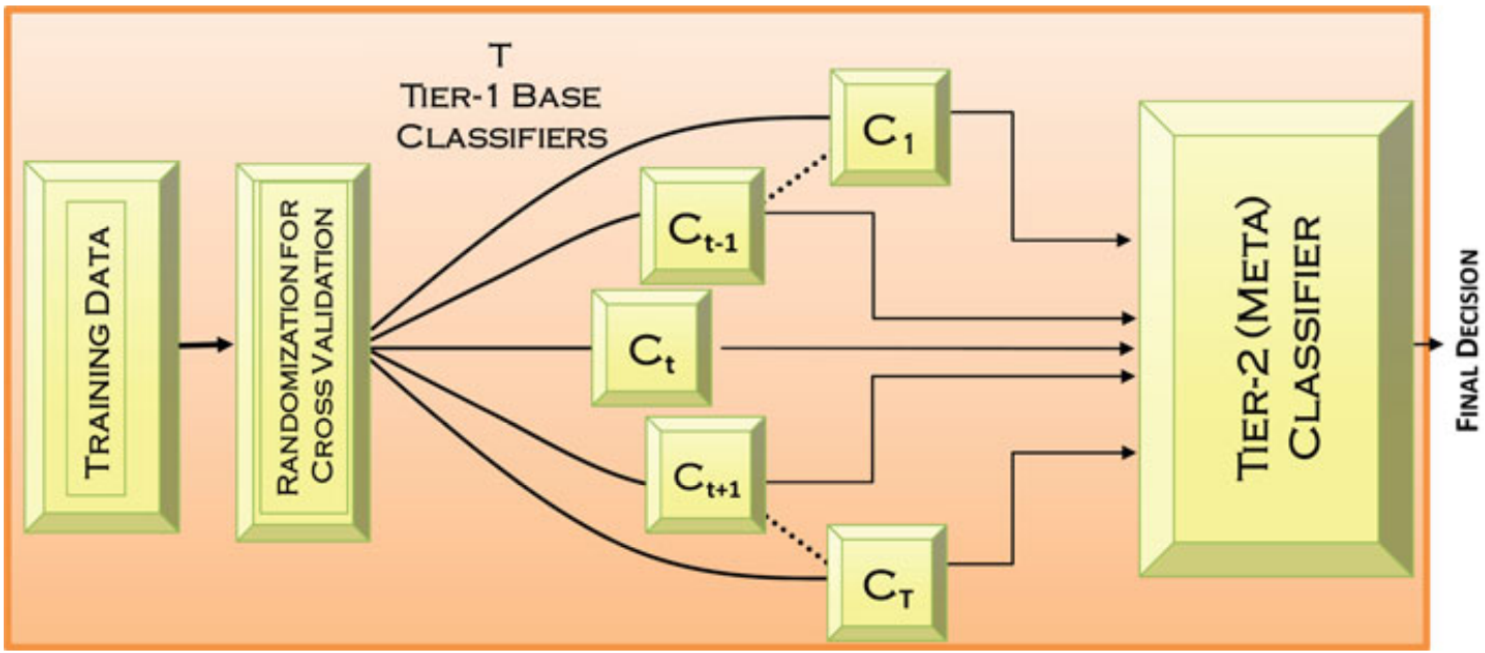
- For boosting:

$$E(F) = w \sum_{i}^{m} E\left(f_i\right)$$

$$Var(F) = m^2 w^2 \sigma^2$$

Stacking

- A separate classifier, trained on the outputs of the ensemble members
- Suppose we first train T classifiers base on a cross-validation partition of the training data
- The outputs of these classifiers on their pseudo-training blocks constitute the training data



Combining Ensemble Members

- Combine the individual classifiers: depends on the type of classifiers
- Combining Class Labels:

$$d_{t,c} \in \{0,1\}, t = 1, \ldots, T, c = 1, \ldots, C$$

1. Majority voting: unanimous voting, simple majority, plurality voting

$$\sum_{t=1}^{T} d_{t,c^*} = \max_c \sum_{t=1}^{T} d_{t,c}$$

2. Weighted Majority Voting:

$$\sum_{t=1}^{T} w_t d_{t,c^*} = \max_c \sum_{t=1}^{T} w_t d_{t,c}$$

3. Borda Count: rank order the classifier outputs

Combining Ensemble Members

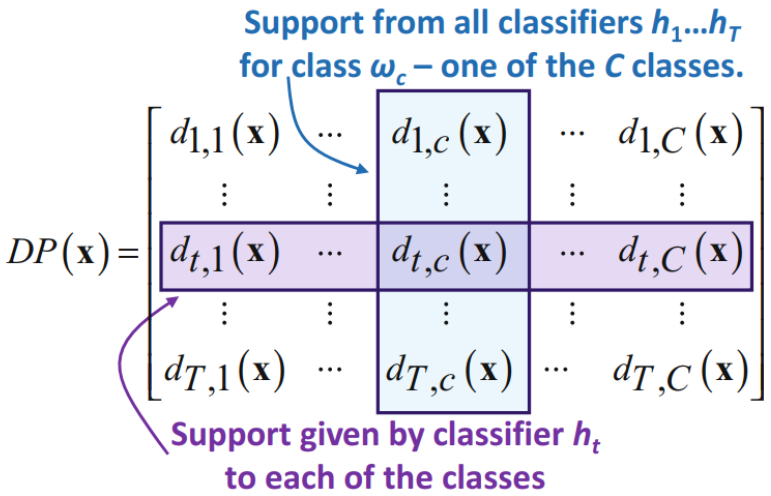
- How do we assign the weights?
- A plausible and commonly used strategy is to use the performance of a classifier on a separate validation (or even training) dataset, as an estimate of that classifier's generalization performance (AdaBoost)

Combining Continuous Outputs

- An estimate of the posterior probability for that class

$$P(\omega_c \mid \boldsymbol{x}) \approx \tilde{g}_c(\boldsymbol{x}) = \frac{e^{g_c(x)}}{\sum_{i=1}^{C} e^{g_i(x)}} \Rightarrow \sum_{i=1}^{C} \tilde{g}_i(\boldsymbol{x}) = 1$$

- Kuncheva's decision profile matrix $DP(x)$, let $d_{t,c} \in [0,1]$ be the $P(\omega_c \mid \boldsymbol{x})$ of the t classifier



Combining Continuous Outputs

- Algebraic Combiners

$$\mu_c(\boldsymbol{x}) = F\left[d_{1,c}(\boldsymbol{x}), \ldots, d_{T,C}(\boldsymbol{x})\right]$$

- F is one of the following combination functions

1. mean rules: $\mu_c(x) = \frac{1}{T}\sum_{t=1}^{T} d_{t,c}(x)$
2. weighted Average: $\mu_c(\boldsymbol{x}) = \frac{1}{T}\sum_{t=1}^{T} w_t d_{t,c}(\boldsymbol{x})$
3. Trimmed mean: Sometimes classifiers may erroneously give unusually low or high support to a particular class
4. Minimum/Maximum/Median Rule:

$$\mu_c(x) = \min_{t=1,\ldots,T}\left\{d_{t,c}(x)\right\}$$
$$\mu_c(x) = \max_{t=1,\ldots,T}\left\{d_{t,c}(x)\right\}$$
$$\mu_c(x) = \text{median}_{t=1,\ldots,T}\left\{d_{t,c}(x)\right\}$$

Combining Continuous Outputs

5. Product Rule: $\mu_c(\boldsymbol{x}) = \frac{1}{T} \prod_{t=1}^{T} d_{t,c}(\boldsymbol{x})$
6. Generalized Mean: $\mu_c(\boldsymbol{x}) = \left(\frac{1}{T} \sum_{t=1}^{T} (d_{t,c}(\boldsymbol{x}))^{\alpha} \right)^{1/\alpha}$
7. Decision Template: $DT_c = 1/N_c \sum_{X_c \in \omega_c} DP(X_c)$

$$\mu_c(x) = S\left(DP(x), DT_c\right), c = 1, \ldots, C$$

7. S is usually a squared Euclidean distance

$$\mu_c(\boldsymbol{x}) = 1 - \frac{1}{T \times C} \sum_{t=1}^{T} \sum_{i=1}^{C} (DT_c(t,i) - d_{t,i}(\boldsymbol{x}))^2$$

Table of Contents

Review: Classification and Regression Trees

- Grow a binary tree
- At each node, "split" the data into two "daughter" nodes
- Splits are chosen using a splitting criterion
- For regression the predicted value at a node is the average response variable for all observations in the node
- For classification the predicted class is the most common class in the node (majority vote)

Splitting criterion

- For classification: ID3, C4.5, CART
- For regression: CART
- ID3:

$$H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} \log_2 \frac{|C_k|}{|D|}$$

$$H(D \mid A) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} H(D_i)$$

$$\text{Gain}(D, A) = H(D) - H(D \mid A)$$

Splitting criterion

- C4.5:

$$\text{Gain}_{\text{ratio}}(D, A) = \frac{\text{Gain}(D, A)}{H_A(D)}$$

$$H_A(D) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \log_2 \frac{|D_i|}{|D|}$$

- CART(classification):

$$\begin{aligned}\text{Gini}(D) &= \sum_{k=1}^{K} \frac{|C_k|}{|D|} \left(1 - \frac{|C_k|}{|D|}\right) \\ &= 1 - \sum_{k=1}^{K} \left(\frac{|C_k|}{|D|}\right)^2\end{aligned}$$

Splitting criterion

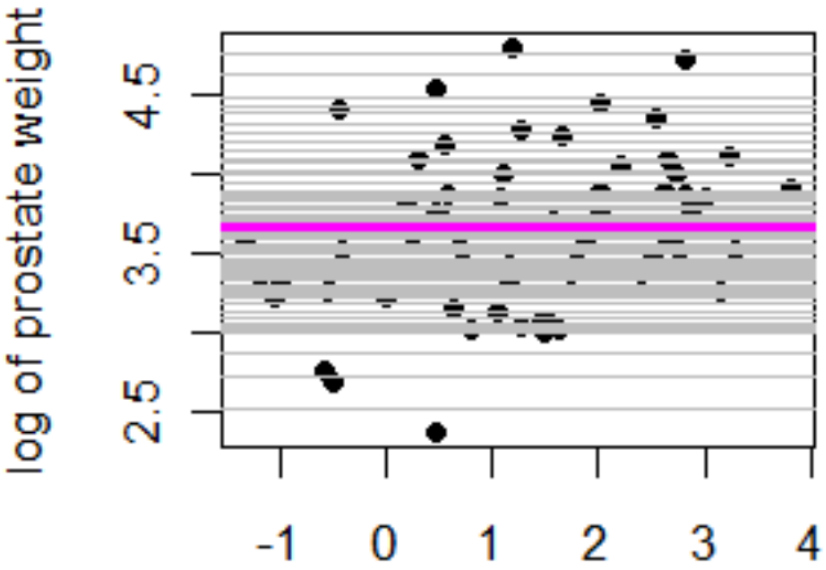
- CART(classification):

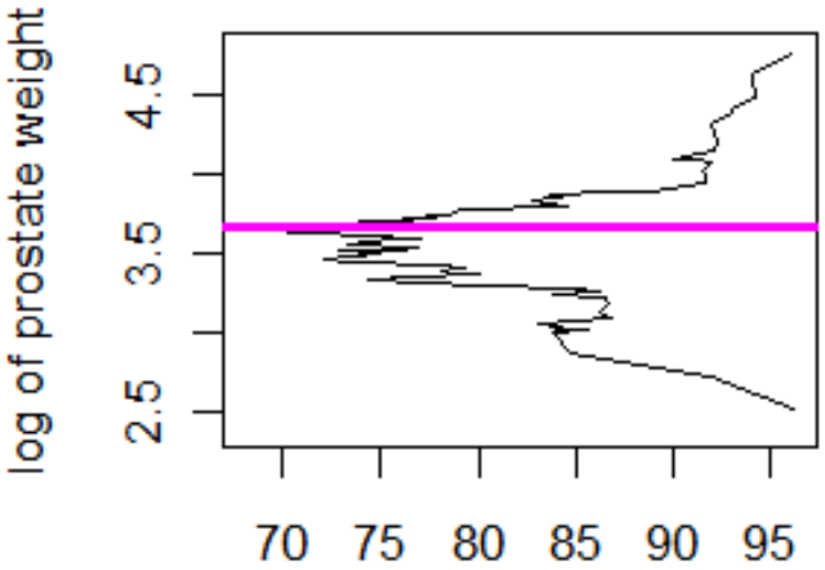
$$\text{Gini}(D \mid A) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} \text{Gini}\left(D_i\right)$$

- CART(regression):

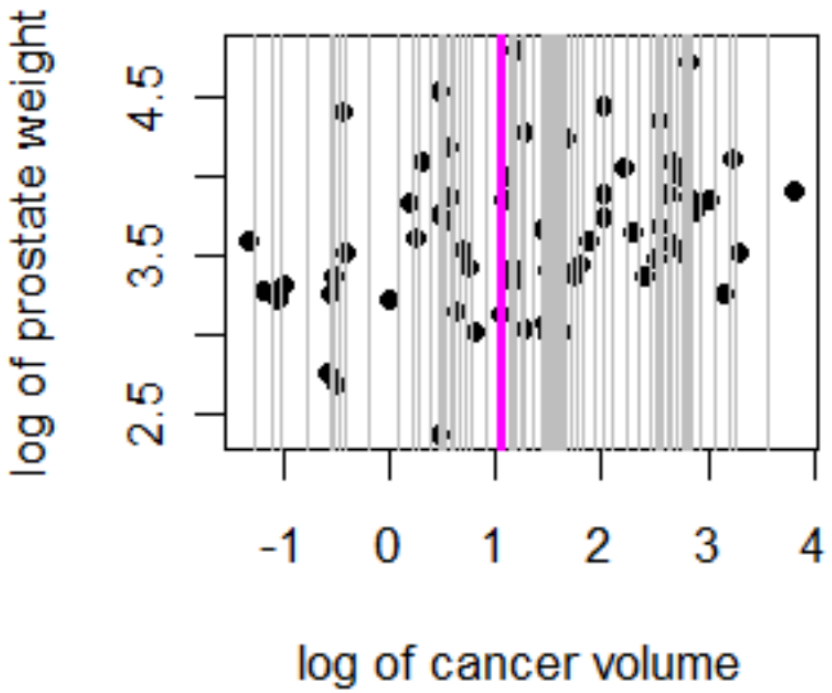
$$\min_{a,s} \left[\min_{c_1} \sum_{x_i \in D_1} \left(y_i - c_1\right)^2 + \min_{c_2} \sum_{x_i \in D_2} \left(y_i - c_2\right)^2 \right]$$

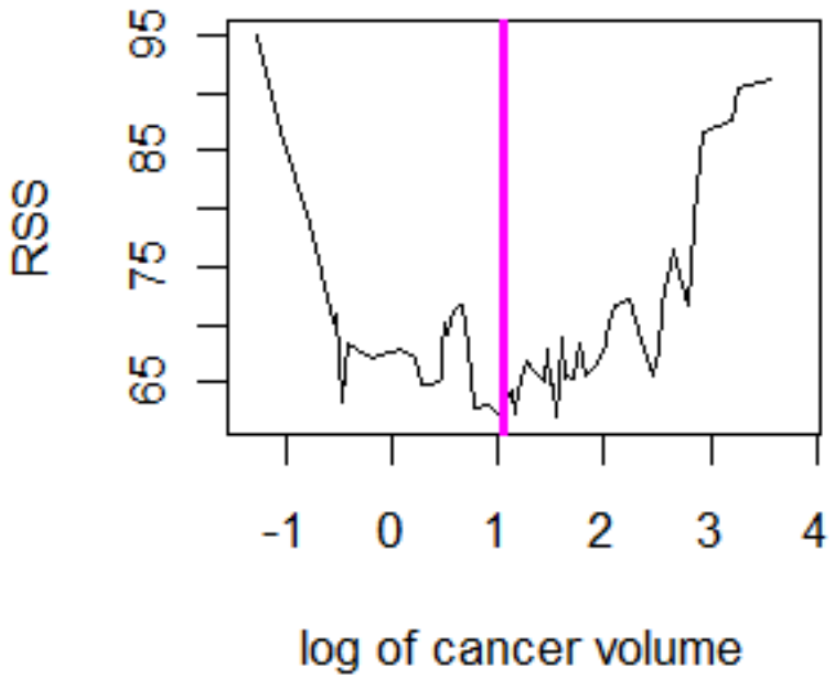
Choosing the best horizontal split



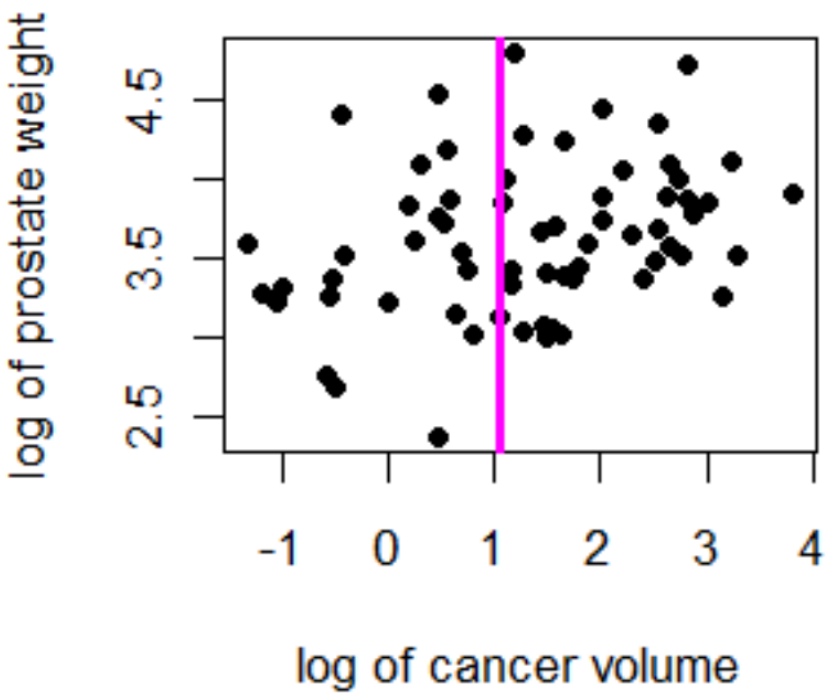


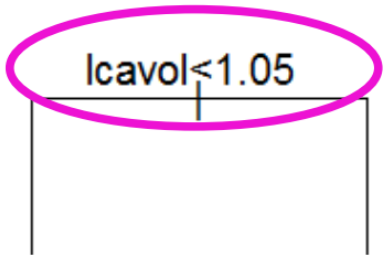
Choosing the best vertical split



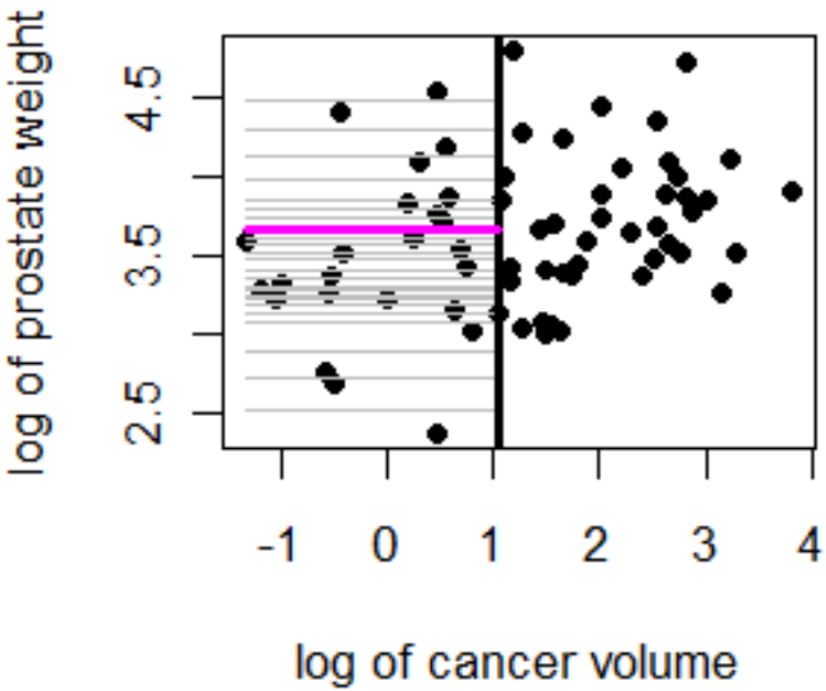


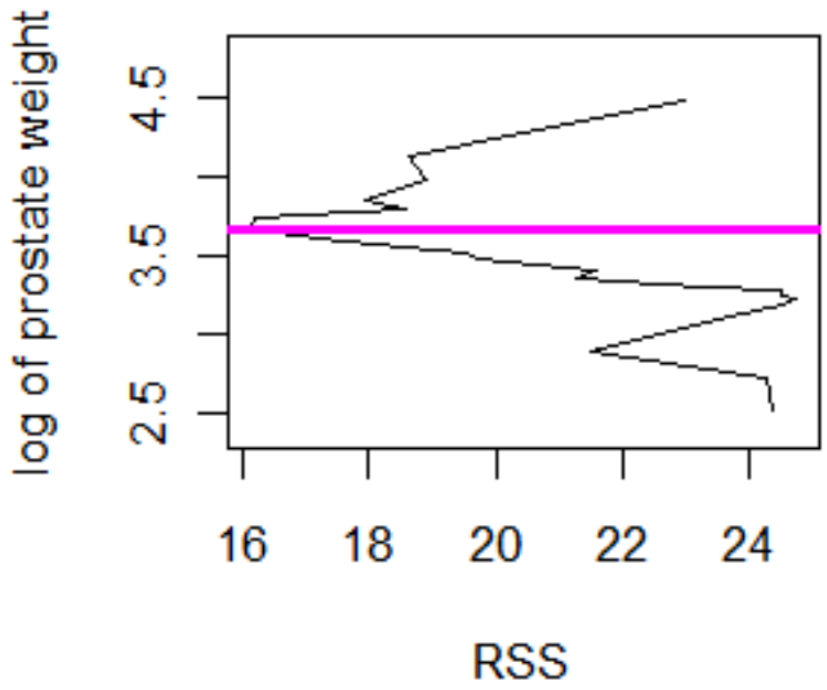
Regression tree (prostate cancer)



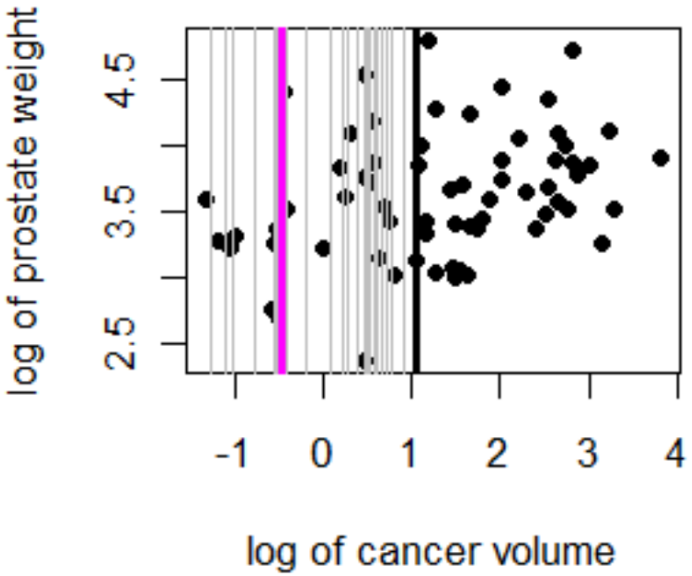


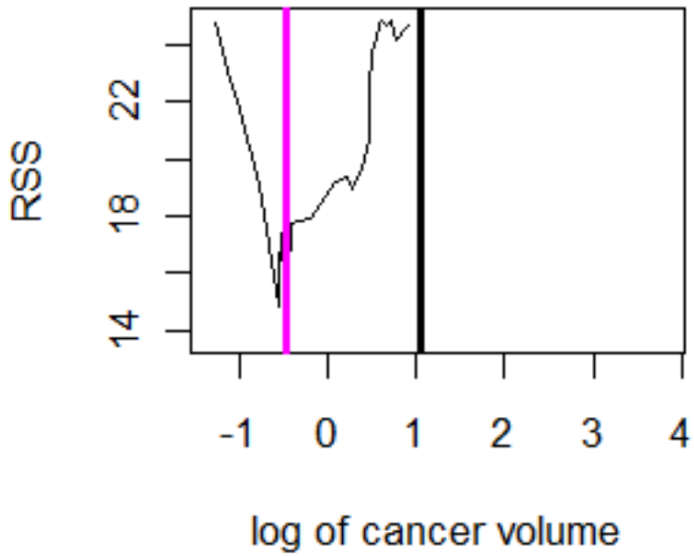
Choosing the best split in the left node





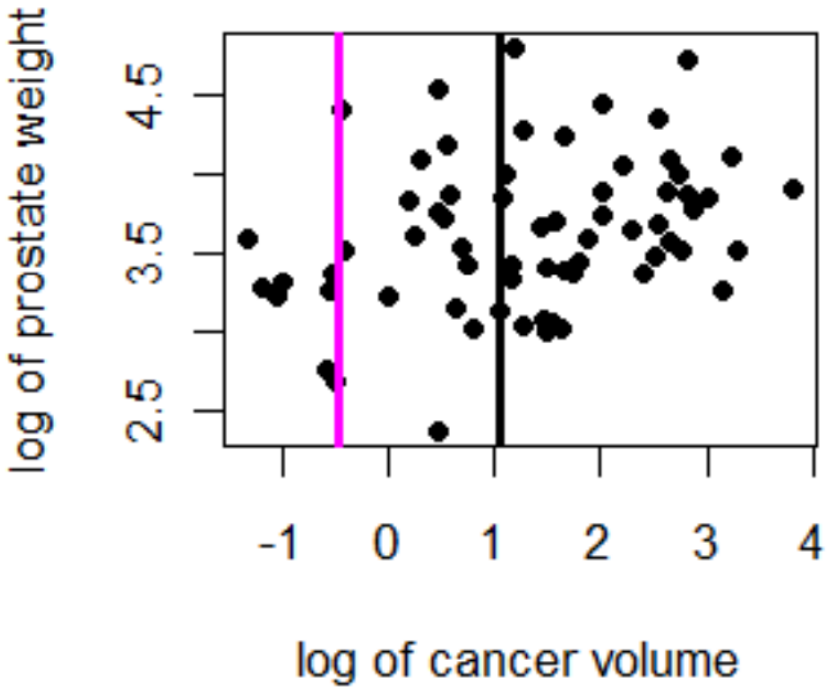
Choosing the best split in the left node

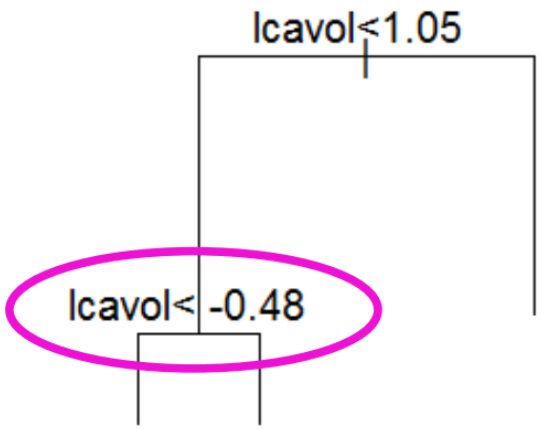




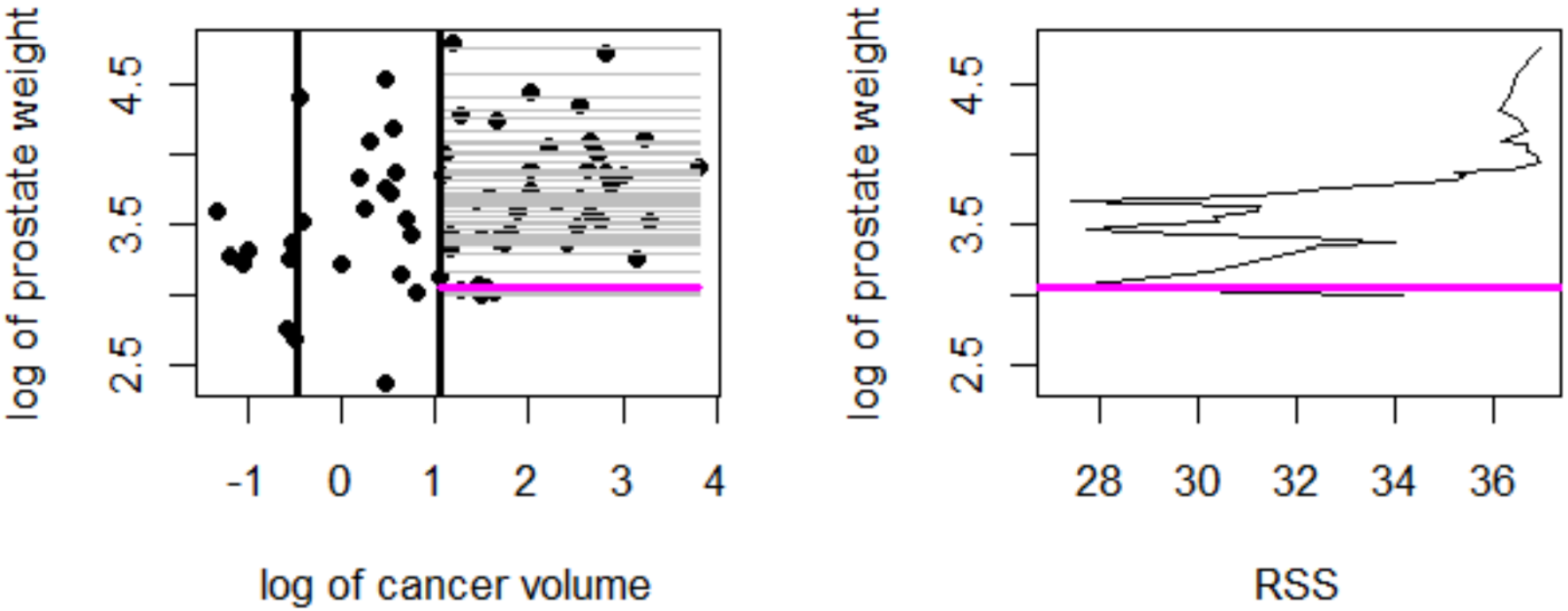
Best vertical split is at -.48 with RSS = 13.61.

Regression tree (prostate cancer)

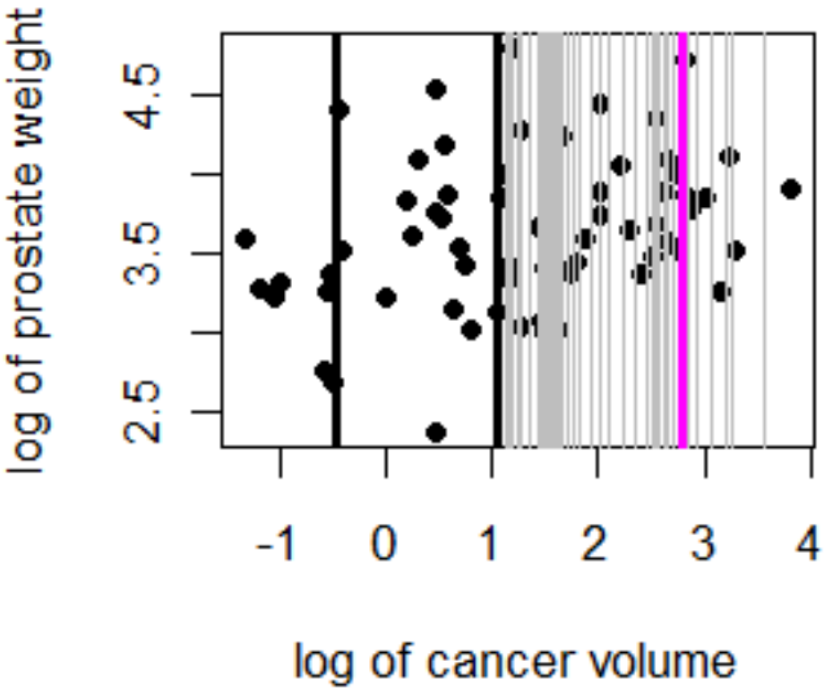


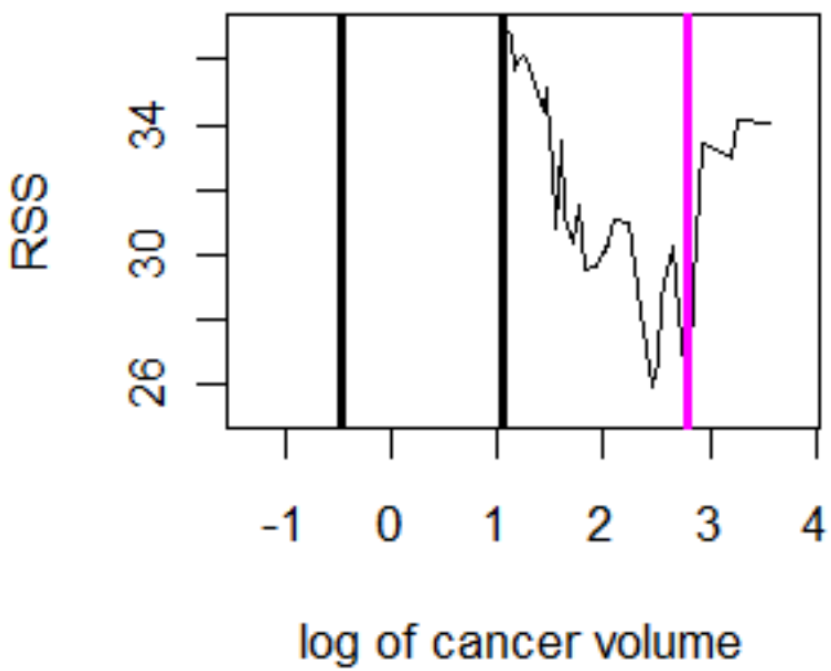


Choosing the best split in the right node

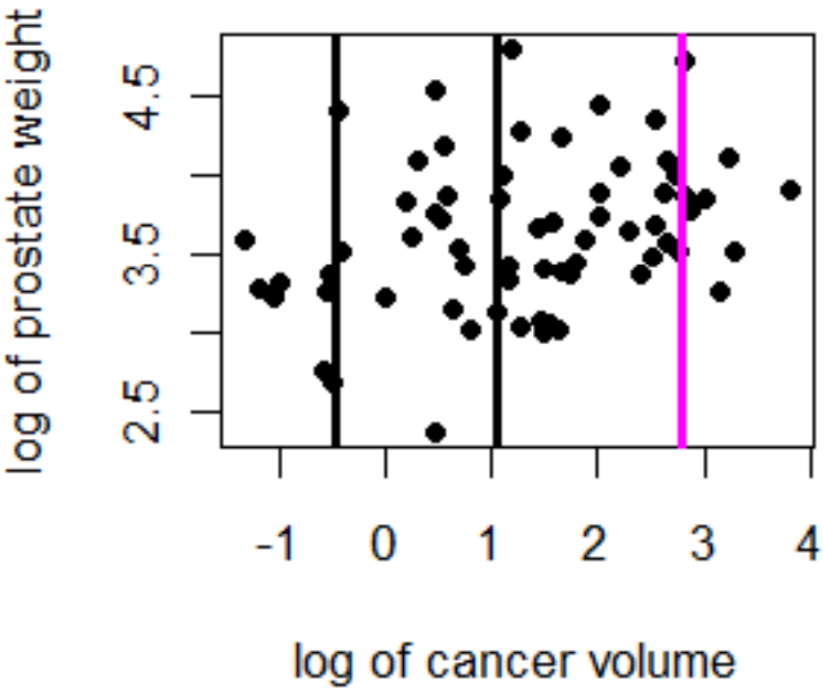


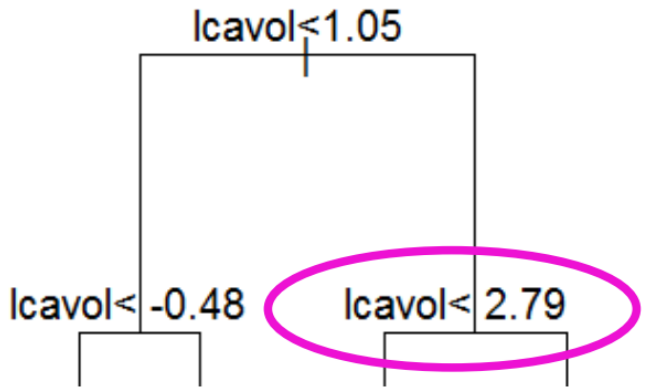
Choosing the best split in the right node



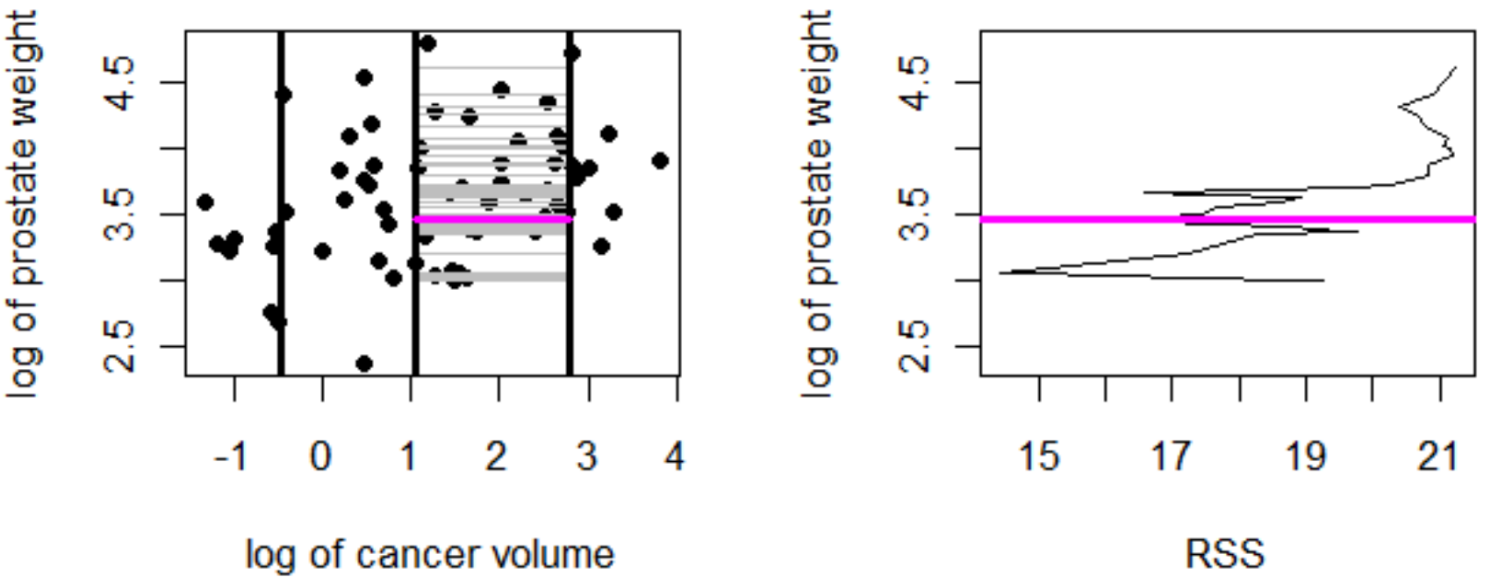


Regression tree (prostate cancer)



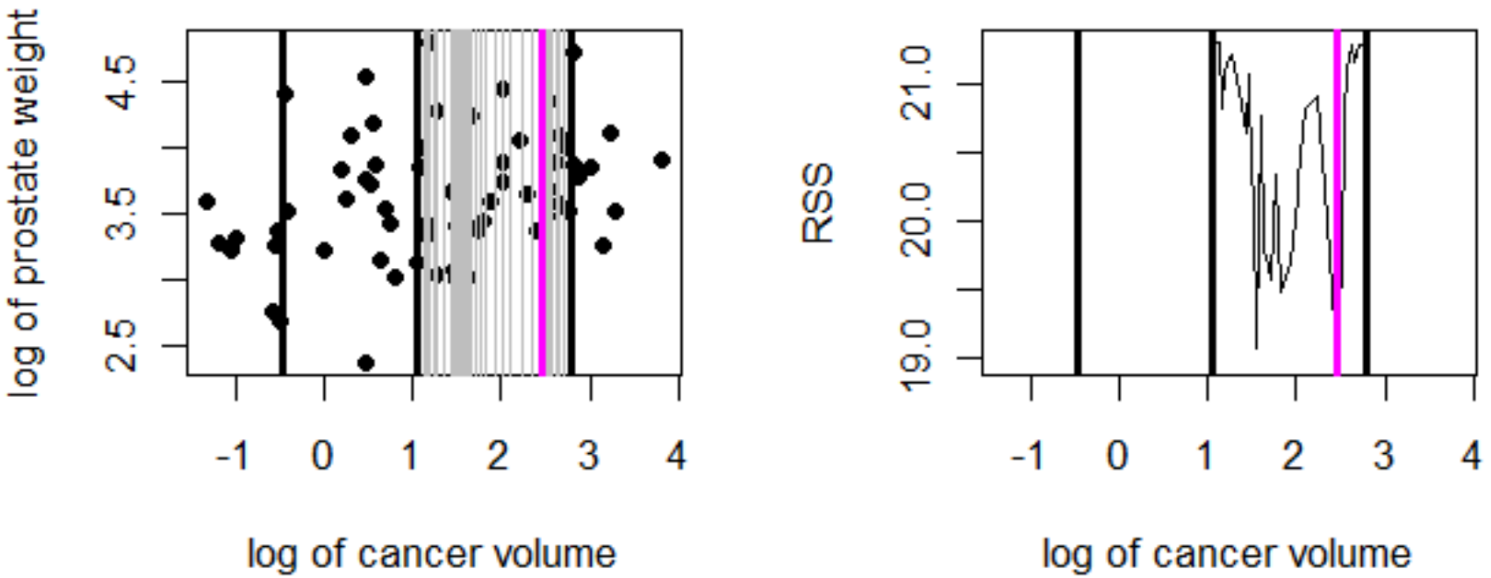


Choosing the best split in the third node



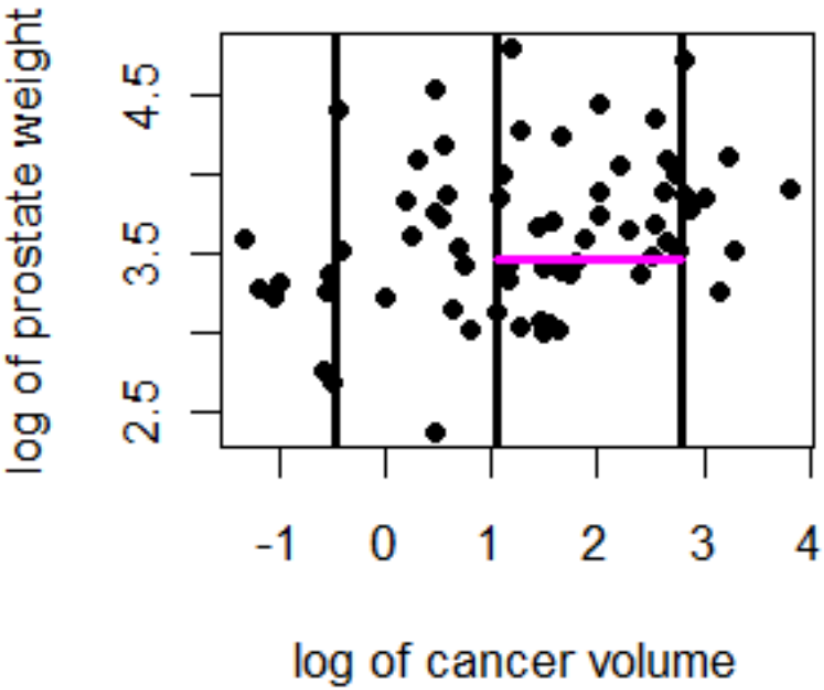
Best horizontal split is at 3.07 with RSS = 14.42, but this is too close to the edge. Use 3.46 with RSS = 16.14.

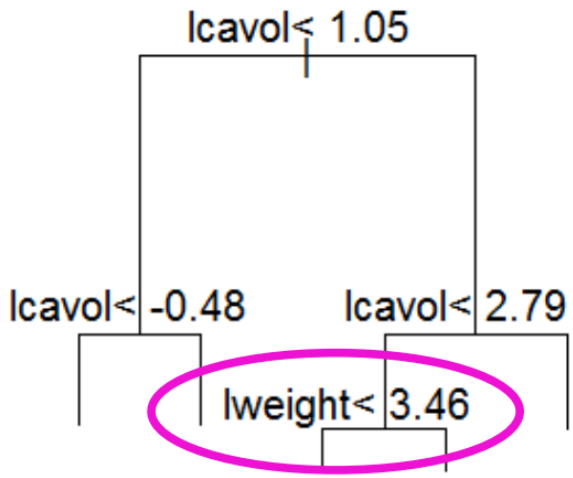
Choosing the best split in the third node



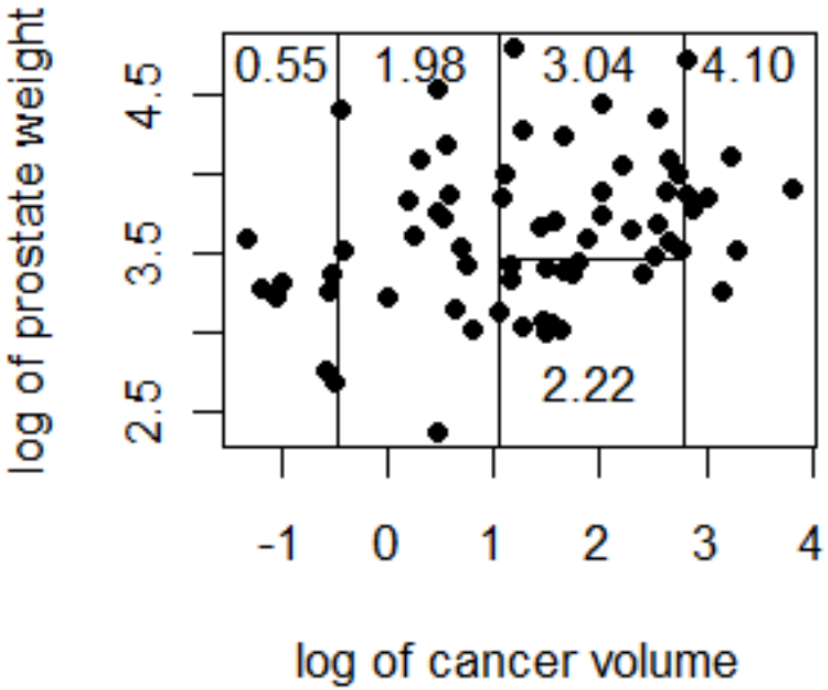
Best vertical split is at 2.46 with RSS = 18.97.

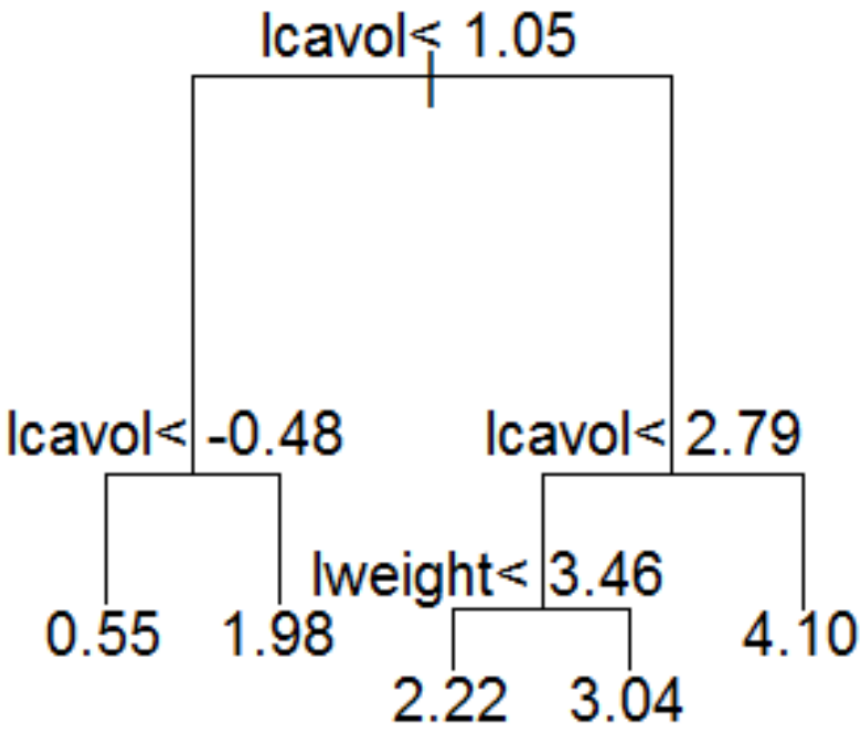
Regression tree (prostate cancer)



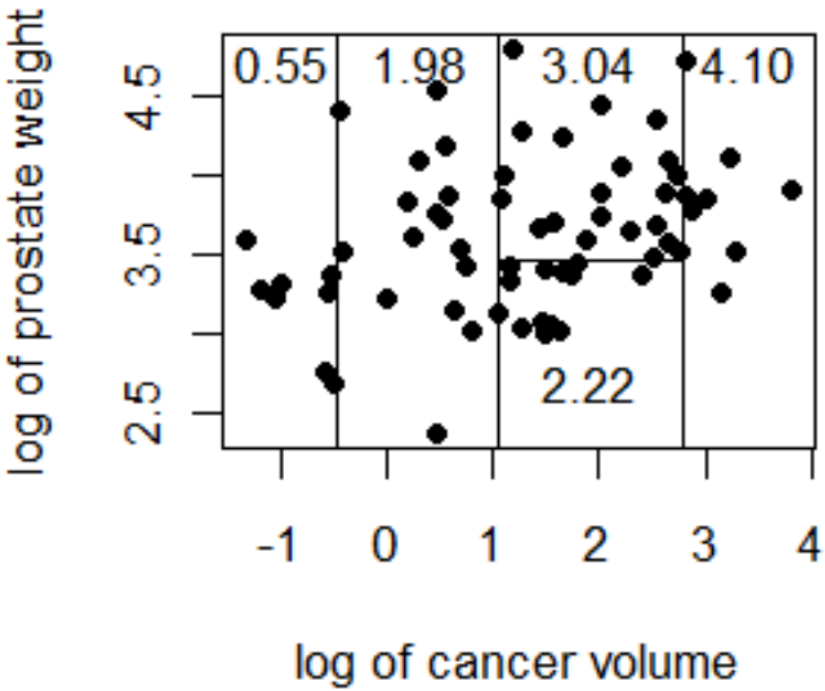


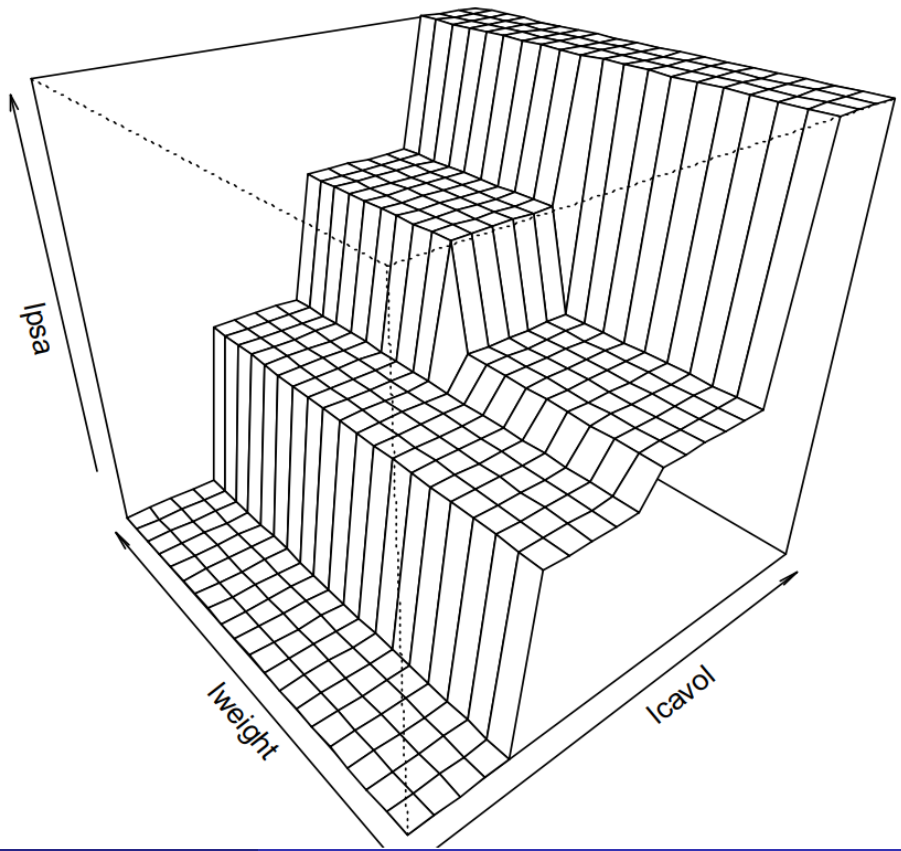
Regression tree (prostate cancer)



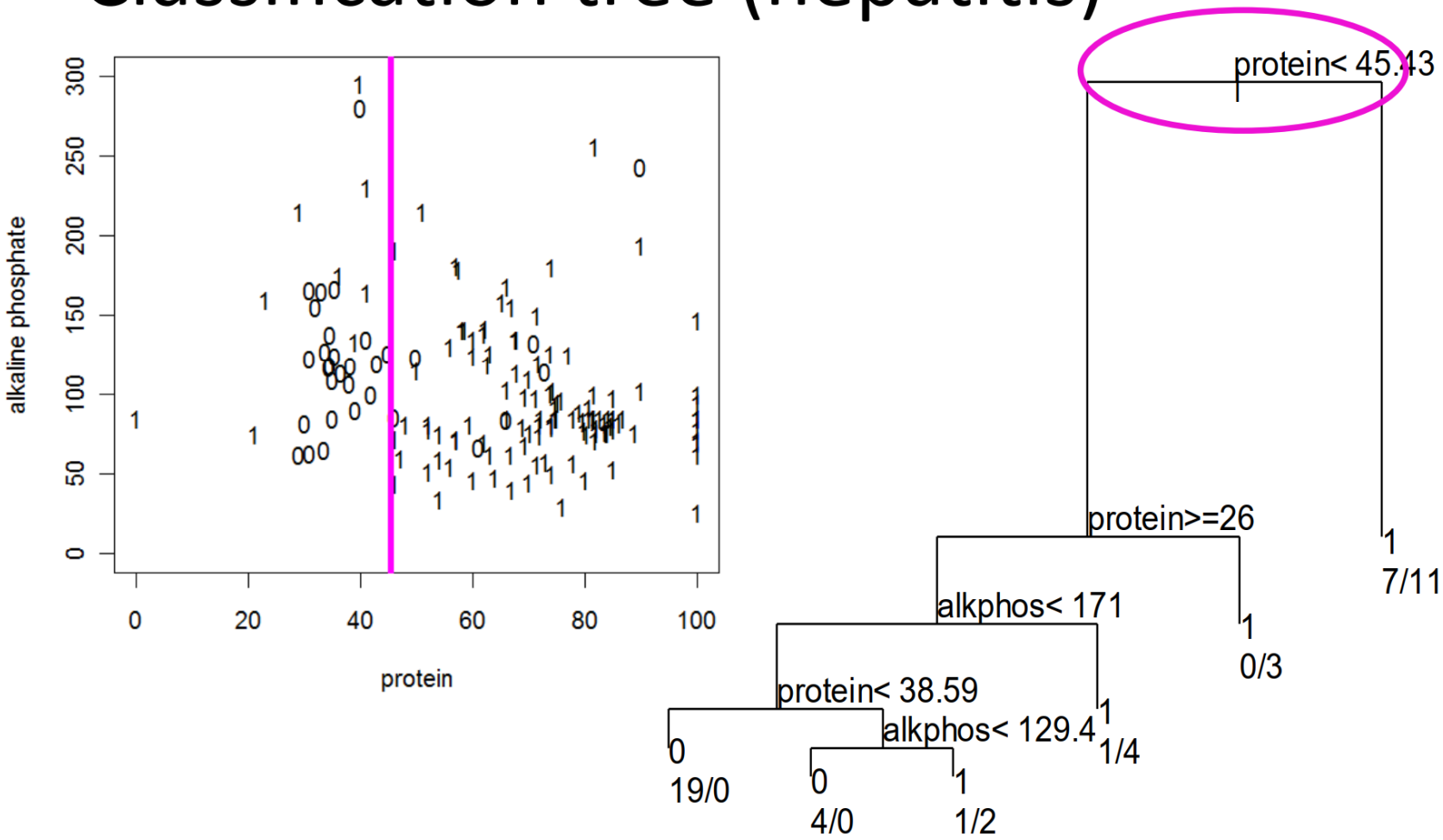


Regression tree (prostate cancer)

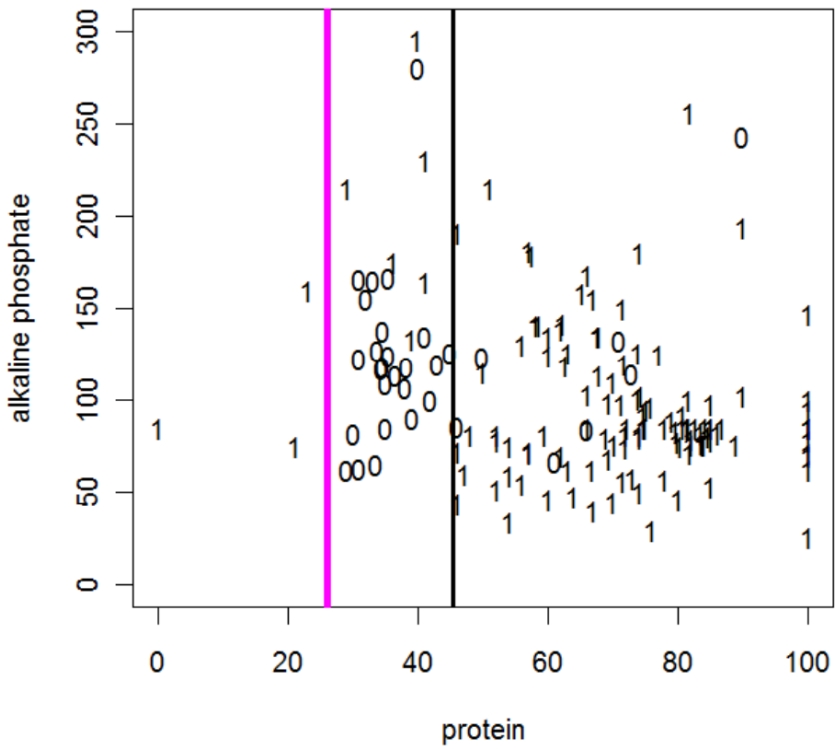


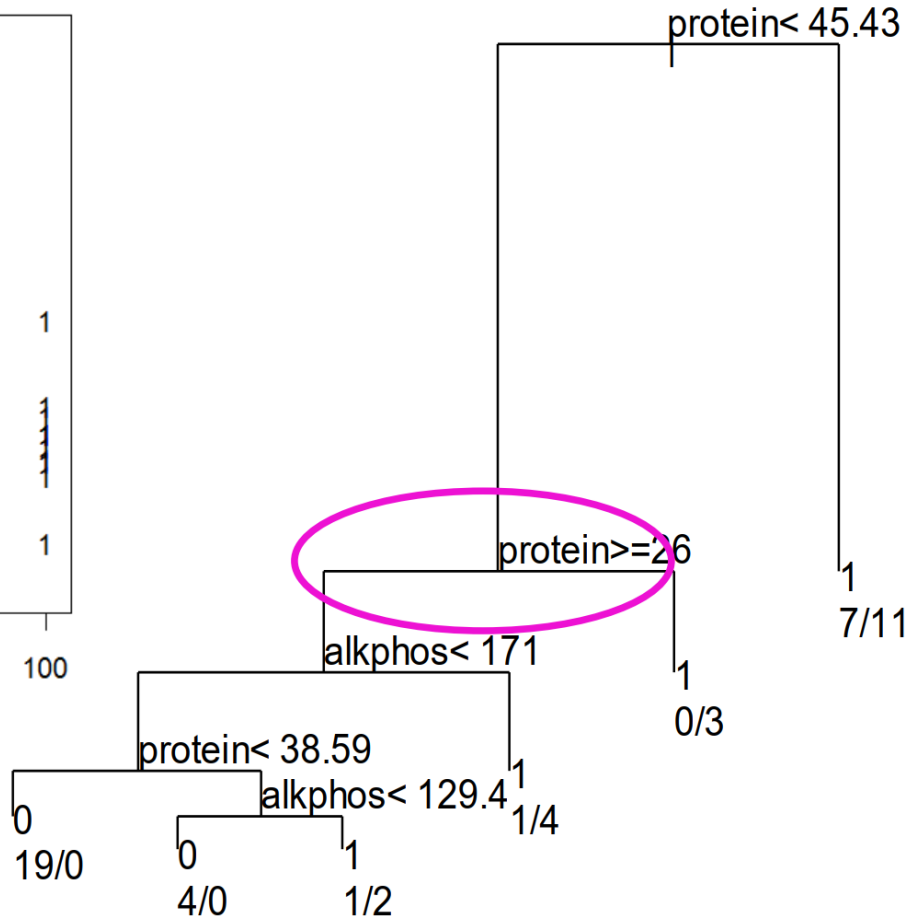


Classification tree (hepatitis)



Classification tree (hepatitis)





The idea of Random Forest

- The essential idea in bagging is to average many noisy but approximately unbiased models
- If B trees are iid with σ^2, then average variance is $\frac{1}{B}\sigma^2$
- If B trees are simply i.d. , the variance of the average is $\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$
- As B increases, the second term disappears, but the first remains
- Random Forest builds a large collection of de-correlated trees, averages trees

Random Forest Algorithm

Algorithm 15.1 *Random Forest for Regression or Classification.*

1. For $b = 1$ to B:
   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size N from the training data.
   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.
      i. Select m variables at random from the p variables.
      ii. Pick the best variable/split-point among the m.
      iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:

Regression: $\hat{f}_{\text{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^B T_b(x)$.

Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\text{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.

The difference of RF and CART

- RF inherit many of the advantages of CART:
  1. Applicable to both regression and classification problems
  2. No formal distributional assumptions (non-parametric)
  3. Can handle highly non-linear interactions and classification boundaries
  4. Handles missing values
- The improvment of RF:
  1. Accuracy
  2. Instability

Details of Random Forest

- Why random forests do not overfit as more trees are added?
- Define the magin function:

$$mg(\mathrm{X}, Y) = \frac{1}{B}\sum_{b=1}^{B} \mathrm{I}(T_b(\mathrm{X}) = Y) - \max_{j \neq Y} \frac{1}{B}\sum_{b=1}^{B} \mathrm{I}(T_b(\mathrm{X}) = j)$$

- The generalization error:

$$PE^* = P_{\mathrm{X},Y}(mg(\mathrm{X}, Y) < 0)$$

- As the number of trees increases, for almost surely converges to

$$P_{\mathrm{X},Y}\left(P_{\Theta}(T(\mathrm{X}, \Theta) = Y) - \max_{j \neq Y} P_{\Theta}(T(\mathrm{X}, \Theta) = j) < 0\right)$$

Details of Random Forest

- The inventors make the following recommendations:

1. For classification, the default value for m is $\lfloor \sqrt{p} \rfloor$ and the minimum node size is one.
2. For regression, the default value for m is $\lfloor p/3 \rfloor$ and the minimum node size is five.

- Out of Bag Samples: the original sample is not chosen in boostraps process
- Suppose we have N samples, the probability of a sample being OOB is:

$$\left(1 - \frac{1}{N}\right)^N \approx e^{-1} \approx 0.368$$

Details of Random Forest

- An oob error estimate is almost identical to that obtained by N-fold crossvalidation
- Denote S_b be the index set of observations in b^{th} tree
- $C_i = \{b_i : i \in S_b\}$ is the trees set using the i^{th} sample
- For classification:

$$\hat{C}_{rf}^{B}(x) = majorityvote\{\hat{C}_b(x)\}_1^B$$

$$Err_{oob} = \frac{1}{N}\sum_{i=1}^{N} I\left(y_i \neq \hat{C}_{rf}^{B}(x_i)\right)$$

$$Err_{cv} = \frac{1}{N}\sum_{i=1}^{N} I\left(y_i \neq F_{cv}^{(-i)}(x_i)\right)$$

Details of Random Forest: OOB estimate

- Out-of bag estimates for strength and correlation
- We have:

$$P_\Theta(T(\mathrm{x},\Theta)=y)-\max_{j\neq \mathrm{y}} P_\Theta(T(\mathrm{x},\Theta)=j)$$

- $Q(\mathrm{x},j)$ is an estimate for $P_\Theta(T(\mathrm{x},\Theta)=j)$

$$Q(\mathrm{x},j)=\sum_k I\left(T\left(\mathrm{x},\Theta_k\right)=j;(y,\mathrm{x})\notin T_{k,B}\right)/\sum_k I\left((y,\mathrm{x})\notin T_{k,B}\right)$$

- The estimate strength is s

$$s=Q(\mathrm{x},y)-\max_{j\neq \mathrm{y}} Q(\mathrm{x},j)$$

Details of Random Forest: OOB estimate

- The estimate correlation is $\bar{\rho}$

$$\bar{\rho} = \text{var}(mr) / \left(E_\Theta sd(\Theta)\right)^2$$

- where

$$\text{var}(mr) = E_{X,Y}\left[P_\Theta(T(x,\Theta) = y) - \max_{j \neq y} P_\Theta(T(x,\Theta) = j)\right]^2 - s^2$$

$$sd(\Theta) = \left[p_1 + p_2 + (p_1 - p_2)^2\right]^{1/2}$$

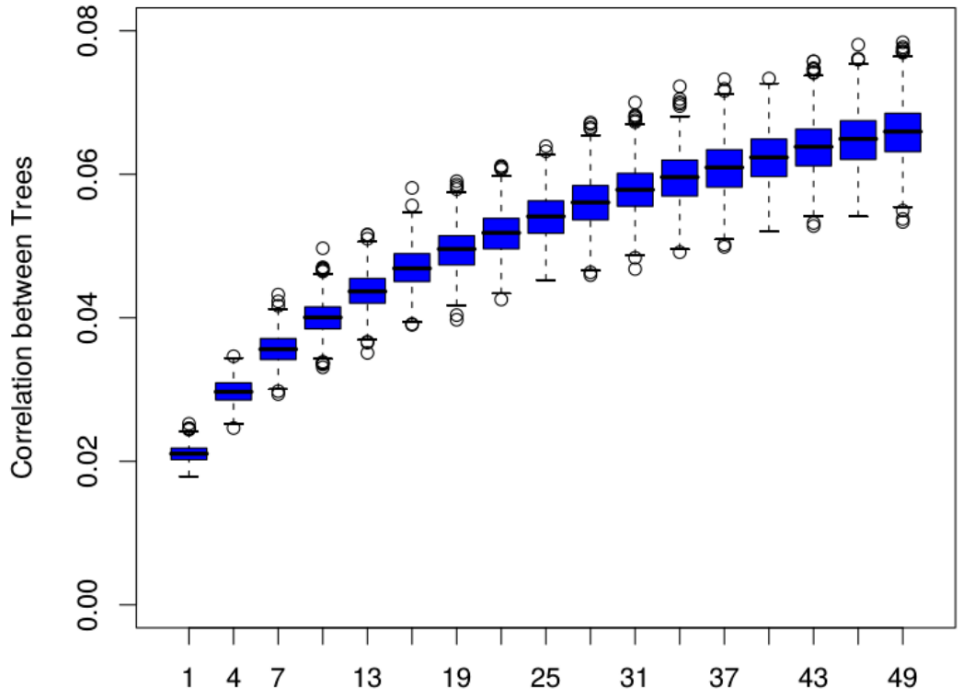
$$p_1 = E_{X,Y}(T(X,\Theta) = Y)$$

$$p_2 = E_{X,Y}(T(X,\Theta) = \hat{j}(X,Y))$$

$$\hat{j}(X,Y) = \arg\max_{j \neq Y} P_\Theta(T(X,\Theta) = j)$$

Details of Random Forest

- $Y = \frac{1}{\sqrt{50}} \sum_{j=1}^{50} X_j + \epsilon$
- pairs of tree predictions at x for different training sets Z are likely to be less similar if they do not use the same splitting variables



Details of Random Forest: variable importance

- Four methods to measure variable importance by Breiman
- Method 1:
  1. randomly permute all values of the m^{th} variable
  2. computing a new internal error rate
  3. The amount by which this new error exceeds the original test set error is defined as the importance of the mth variable
- Method 2:
  1. for the n^{th} sample, calculate the margin
  2. the average lowering of the margin across all cases when the mth variable is randomly permuted as in method 1

Details of Random Forest: variable importance

- Method 3:
  1. for the n^{th} sample, calculate the margin
  2. the count of how many margins are lowered minus the number of margins raised
- Method 4:
  1. the gini criterion

Table of Contents

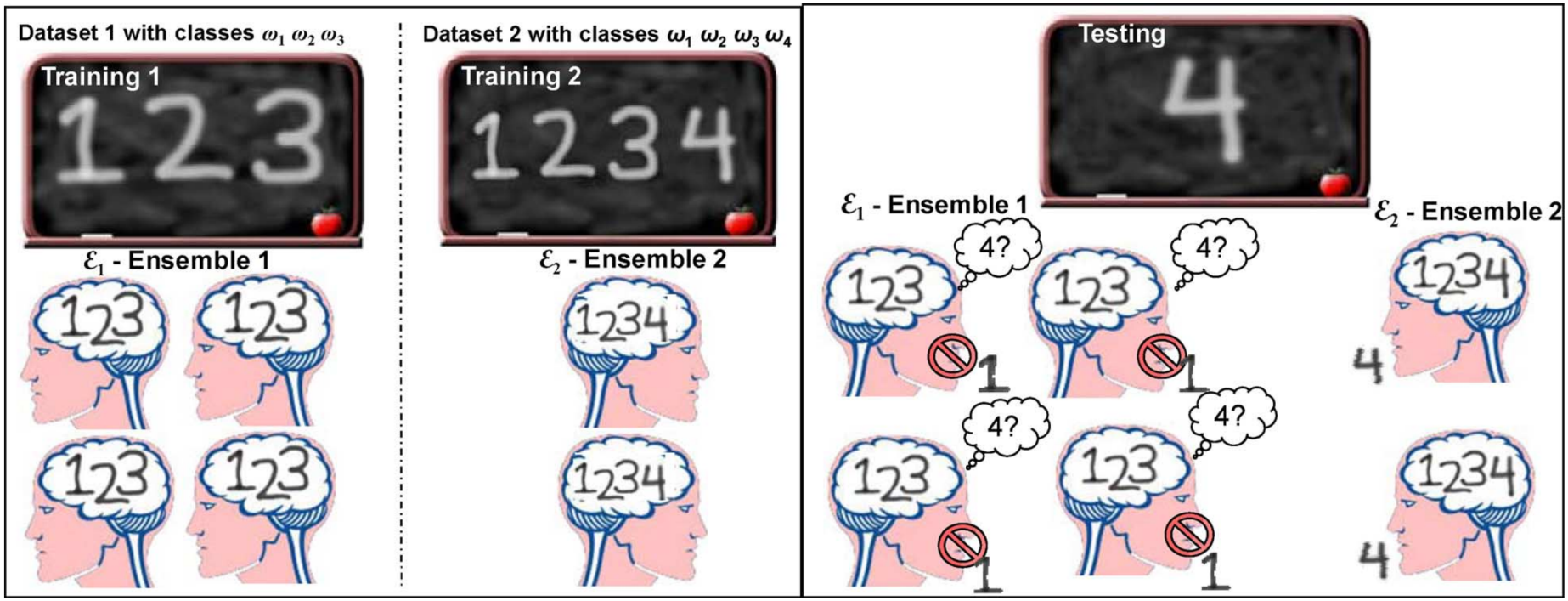
Incremental Learning

- stability–plasticity dilemma
- data often become available in batches over a period of time
- Goal: learn the novel information while retaining as much of the previous knowledge as possible

The idea of Incremental Learning

- discard the existing classifier, and retrain a new one using all data accumulated thus far
- Online learning algorithms: Winston's seminal work, Littlestone's Winnow
  - restricted forms of classifiers
  - Boolean or linear threshold functions,
- train an ensemble of classifiers for each data set
- combine their outputs using an appropriate combination rule
- To generate a diverse set of classifiers

- the weighted majority voting



Incremental Learning

Input: For each dataset $k=1,2,\ldots,K$

- Training data $S_k = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_i, y_i), \cdots, (\mathbf{x}_{m_k}, y_{m_k})\}$, $\mathbf{x}_i \in X; y_i \in Y_k \subset \{\omega_1, \ldots, \omega_C\}$
- Weak learning algorithm **BaseClassifier**.
- Integer T_k, specifying the number of **BaseClassifier**s to create using $\mathcal{D}_k$.

Do for $k=1,2,\ldots,K$

Initialize the instance weights to be uniform when new data arrive $w_1^k(i) = 1/m_k, i = 1, \cdots, m_k$

If $k \neq 1$, **Set** t=0 and **Go to** step 5 of the inner **Do loop** to adjust initialization weights
Do for $t=1,2,\ldots, T_k$:

1. Set $\boldsymbol{D}_t^k = \boldsymbol{w}_t^k \Big/ \sum_{i=1}^{m_k} w_t^k(i)$ so that $\boldsymbol{D}_t^k$ is a distribution.
2. Call **BaseClassifier**, providing it with $TR_t^k \subset S_k$, drawn according to $\boldsymbol{D}_t^k$.
3. Obtain a hypothesis $h_t^k: X \rightarrow Y_k$, and calculate its error $\varepsilon_t^k = \sum_{i:h_t^k(\mathbf{x}_i) \neq y_i} D_t^k(i)$. If $\varepsilon_t^k > ½$, discard h_t^k and go to step 2. Otherwise, compute normalized error $\beta_t^k = \varepsilon_t^k / (1 - \varepsilon_t^k)$.
4. Let $CL_t^k = Y_{k(t)} \subset \{\omega_1, \cdots \omega_C\}$ be the class labels used in training h_t^k for dataset $\mathcal{D}_k$.
5. Call **DW-CAV** to obtain the composite hypothesis H_t^k.
6. Compute the error of the composite hypothesis $E_t^k = \sum_{i:H_t^k(\mathbf{x}_i) \neq y_i} D_t^k(i)$
7. Set $B_t^k = E_t^k/(1-E_t^k)$, and update the weights: $w_{t+1}^k(i) = w_t^k(i) \times \begin{cases} B_t^k, & \text{if } H_t^k(\mathbf{x}_i) = y_i \\ 1, & \text{otherwise} \end{cases}$

End
End
Call **DW-CAV** to obtain the final hypothesis, H_{final}

Incremental Learning

Inputs:

- Instance $\mathbf{x}_i$ to be classified
- All classifiers h_t^k generated thus far
- Normalized error values, β_t^k.
- Class labels, CL_t^k used in training h_t^k.

Initialize classifier voting weights $W_t^k = \log\left(1/\beta_t^k\right)$

Calculate for each $\omega_C \in \left\{\omega_1, ..., \omega_C\right\}$

- Normalization factor $Z_c = \sum_k \sum_{t:c \in CL_t^k} W_t^k$
- Class-specific confidence $P_c(i) = \dfrac{\sum_k \sum_{t:h_t^k(\mathbf{x}_i)=\omega_c} W_t^k}{Z_c}$

If $P_k(i) = P_l(i) = 1, k \neq l$ such that

$\mathcal{E}_k \cap \mathcal{E}_l = \varnothing \rightarrow$ Set $P_k(i) = P_l(i) = 0$

(where $\mathcal{E}_k$ is the set of classifiers that have seen class ω_k)

Update voting weights for instance $\mathbf{x}_i$

$$W_t^k(i) = W_t^k \cdot \prod_{c:\omega_c \notin CL_t^k} \left(1 - P_c(i)\right)$$

Compute final (or current composite) hypothesis

$$H_{final}(\mathbf{x}_i) = \arg\max_{\omega_c \in \Omega} \sum_k \sum_{t:h_t^k(\mathbf{x}_i)=\omega_c} W_t^k(i)$$