THE GEORGE WASHINGTON UNIVERSITY

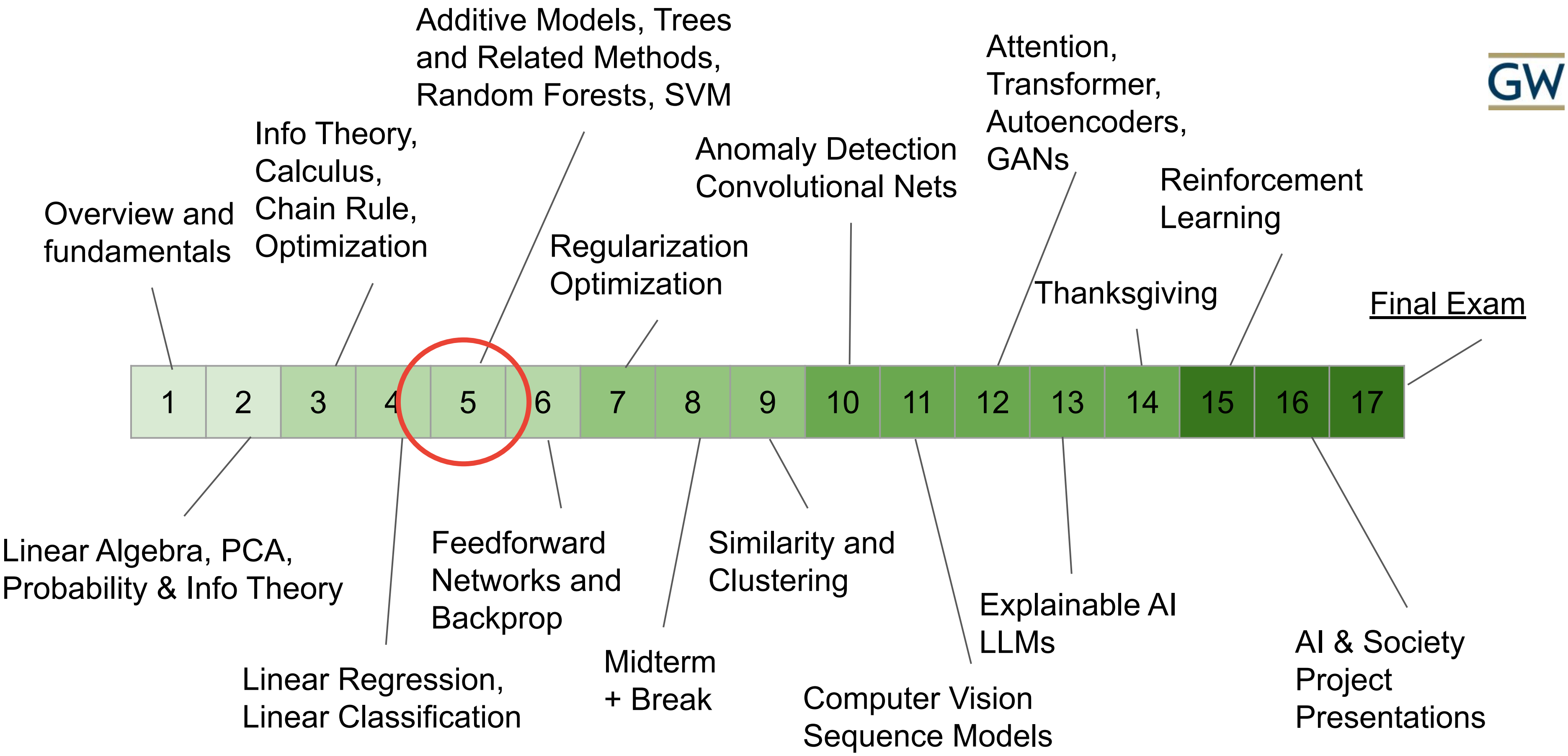
WASHINGTON, DC

CS 4364/6364

Machine Learning

Fall Semester 9/19/2023
Lecture 8.
Decision Trees & Random Forests

John Sipple
jsipple@gwu.edu



Overview and fundamentals

Linear Algebra, PCA, Probability & Info Theory

Info Theory, Calculus, Chain Rule, Optimization

Linear Regression, Linear Classification

Additive Models, Trees and Related Methods, Random Forests, SVM

Feedforward Networks and Backprop

Regularization Optimization

Midterm + Break

Similarity and Clustering

Anomaly Detection Convolutional Nets

Computer Vision Sequence Models

Attention, Transformer, Autoencoders, GANs

Explainable AI LLMs

Thanksgiving

Reinforcement Learning

AI & Society Project Presentations

Final Exam

Topics

Decision Trees

Ensemble Methods

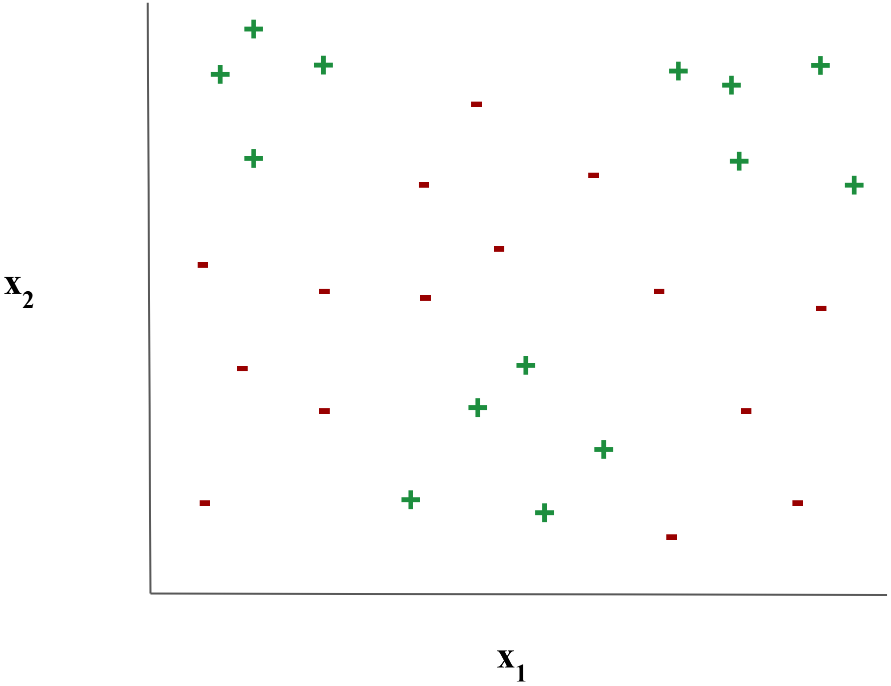
Bagging

Random Forests

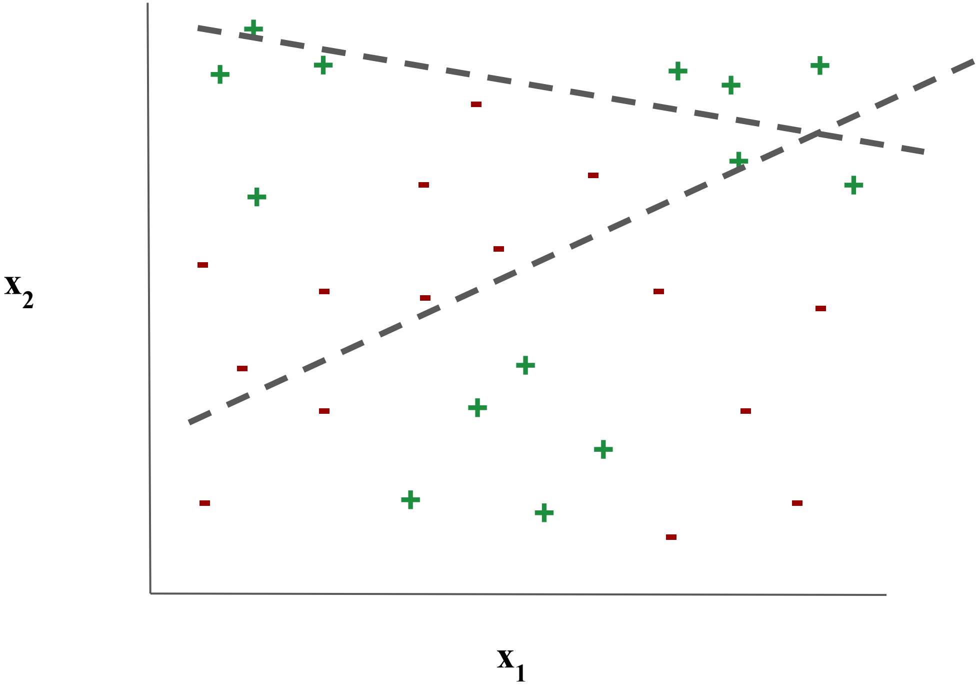
Boosting

Decision Trees

Motivating Example

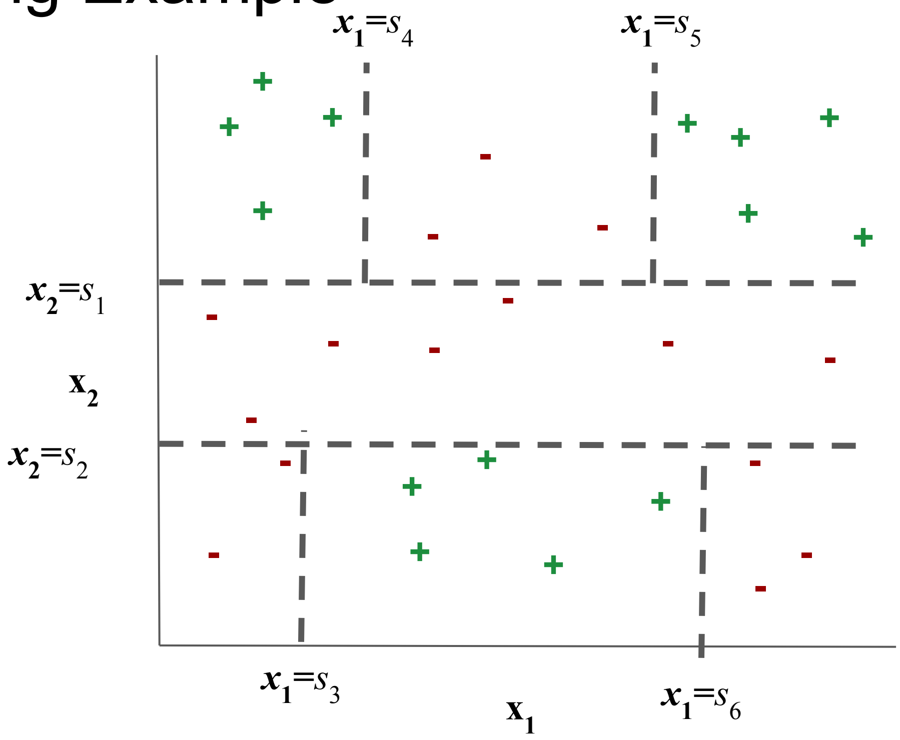


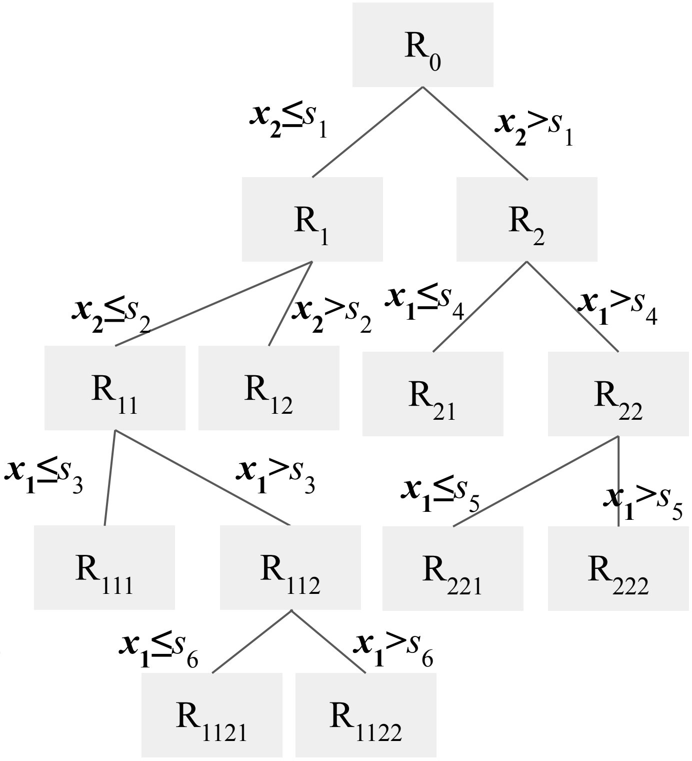
Motivating Example



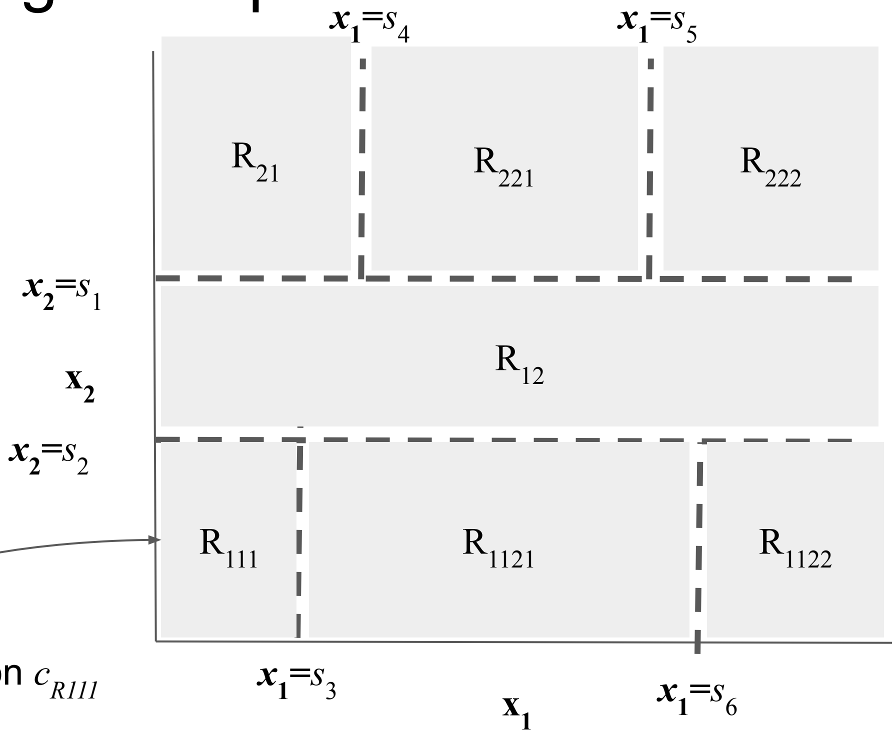
No Logistic Regression Model is an adequate fit to the data

Motivating Example

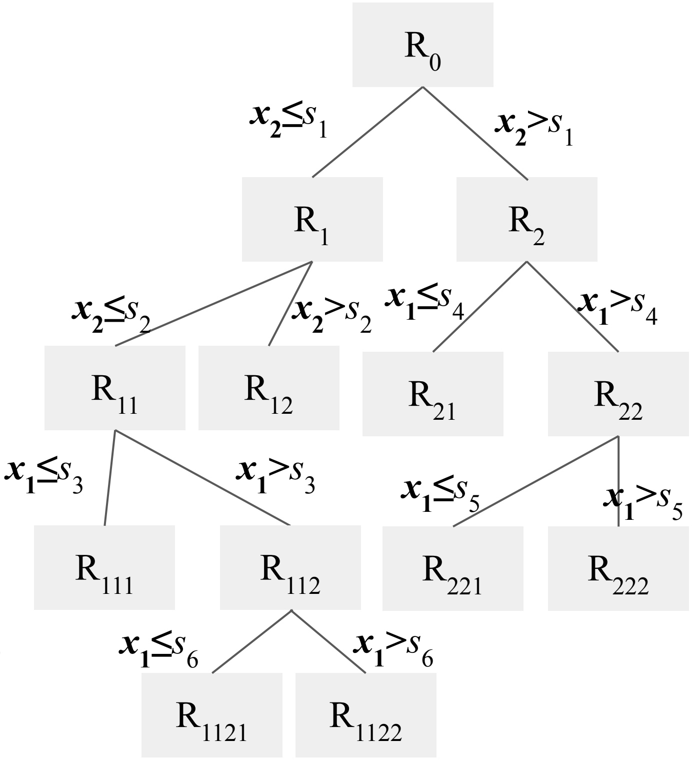




Motivating Example



Region R_{111} with constant prediction c_{R111}



Tree-Based Method: Decision Rule

Top-down, greedy, recursive split of features

Each leaf region R is represented by a constant:

$$\hat{f}(\boldsymbol{x}_i) = \sum_{R_m \in \boldsymbol{R}} c_R I\{\boldsymbol{x}_i \in R_m\}$$

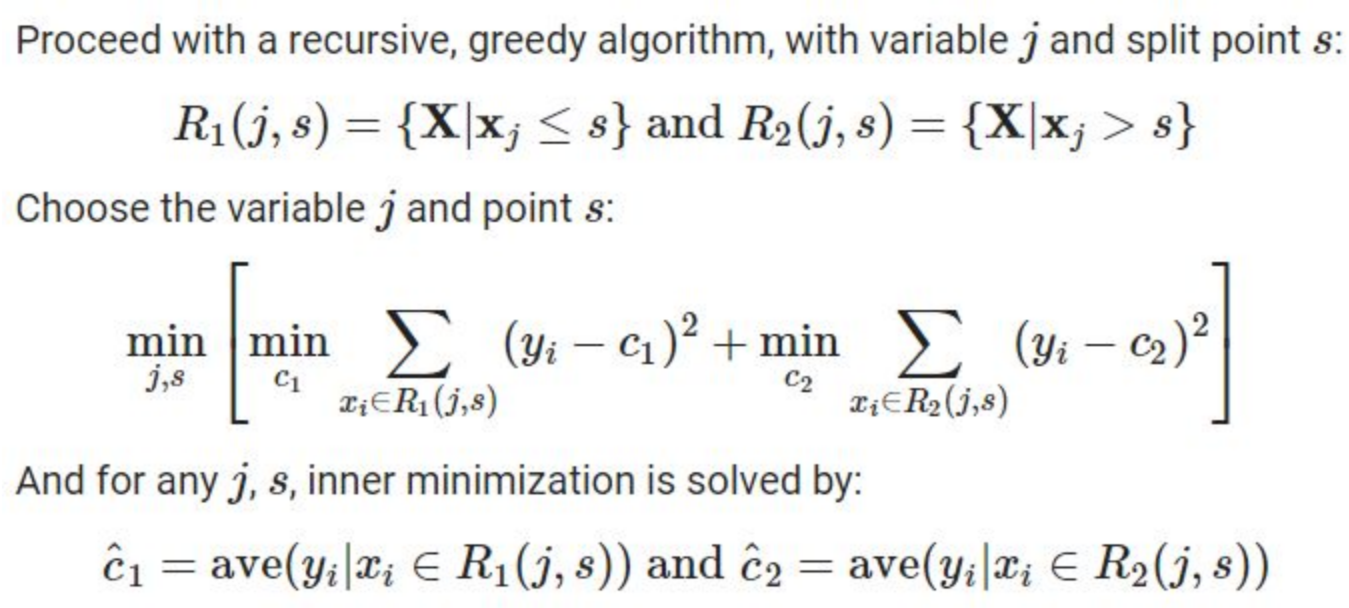
- **Regression**: Average value of each point

$$\hat{c}_R = \text{ave}(y_i | x_i \in R)$$

- **Classification**: Proportion of each class k

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

Growing Regression Trees



Proceed with a recursive, greedy algorithm, with variable j and split point s:

$$R_1(j,s) = \{\mathbf{X}|\mathbf{x}_j \leq s\} \text{ and } R_2(j,s) = \{\mathbf{X}|\mathbf{x}_j > s\}$$

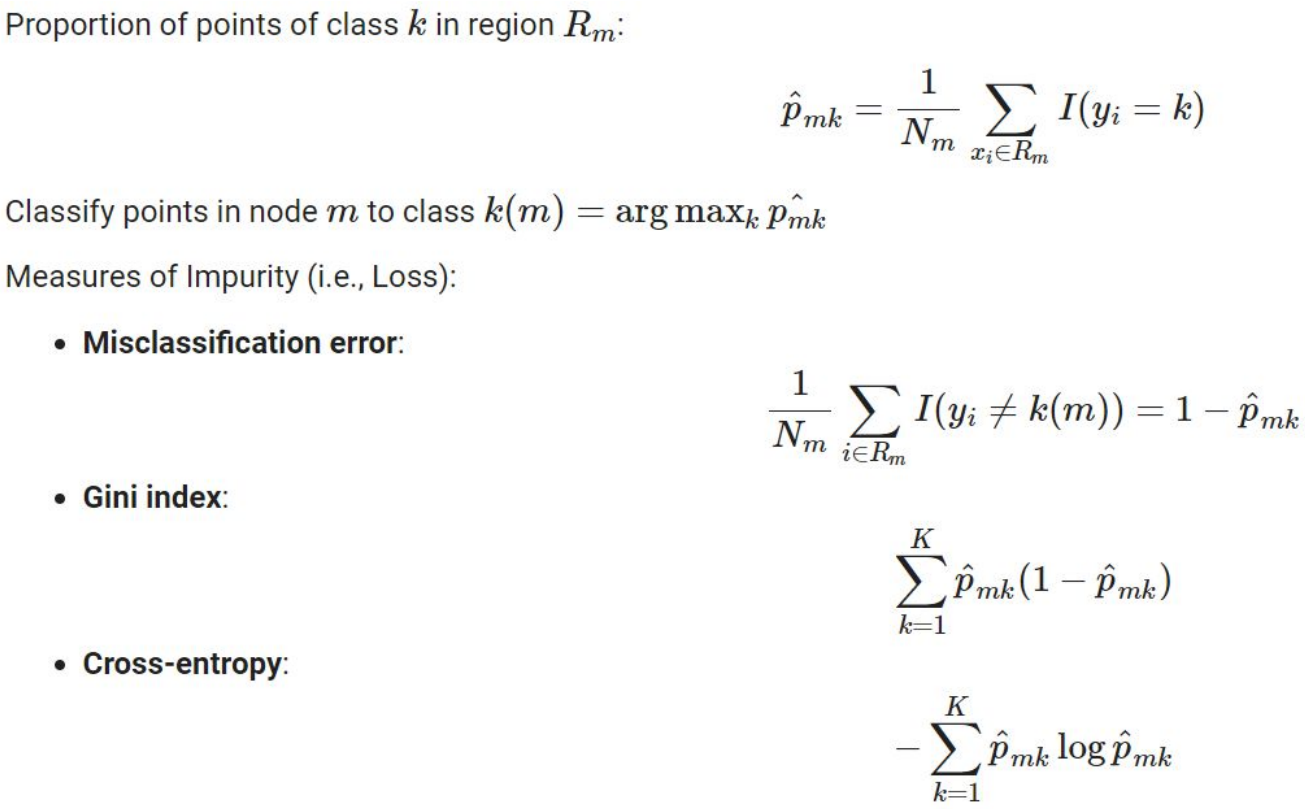
Choose the variable j and point s:

$$\min_{j,s}\left[\min_{c_1}\sum_{x_i \in R_1(j,s)}(y_i - c_1)^2 + \min_{c_2}\sum_{x_i \in R_2(j,s)}(y_i - c_2)^2\right]$$

And for any j, s, inner minimization is solved by:

$$\hat{c}_1 = \text{ave}(y_i|x_i \in R_1(j,s)) \text{ and } \hat{c}_2 = \text{ave}(y_i|x_i \in R_2(j,s))$$

Classification Trees



Proportion of points of class k in region R_m:

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

Classify points in node m to class $k(m) = \arg\max_k \hat{p_{mk}}$

Measures of Impurity (i.e., Loss):

- **Misclassification error**:

$$\frac{1}{N_m} \sum_{i \in R_m} I(y_i \neq k(m)) = 1 - \hat{p}_{mk}$$

- **Gini index**:

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

- **Cross-entropy**:

$$-\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

Comparing Loss Functions

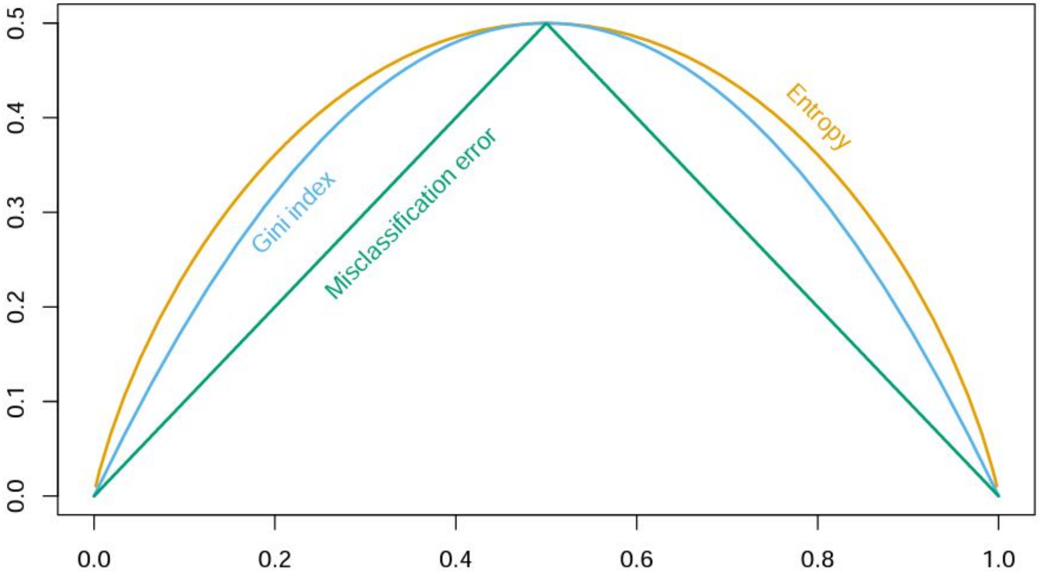


FIGURE 9.3. *Node impurity measures for two-class classification, as a function of the proportion p in class 2. Cross-entropy has been scaled to pass through* $(0.5, 0.5)$.

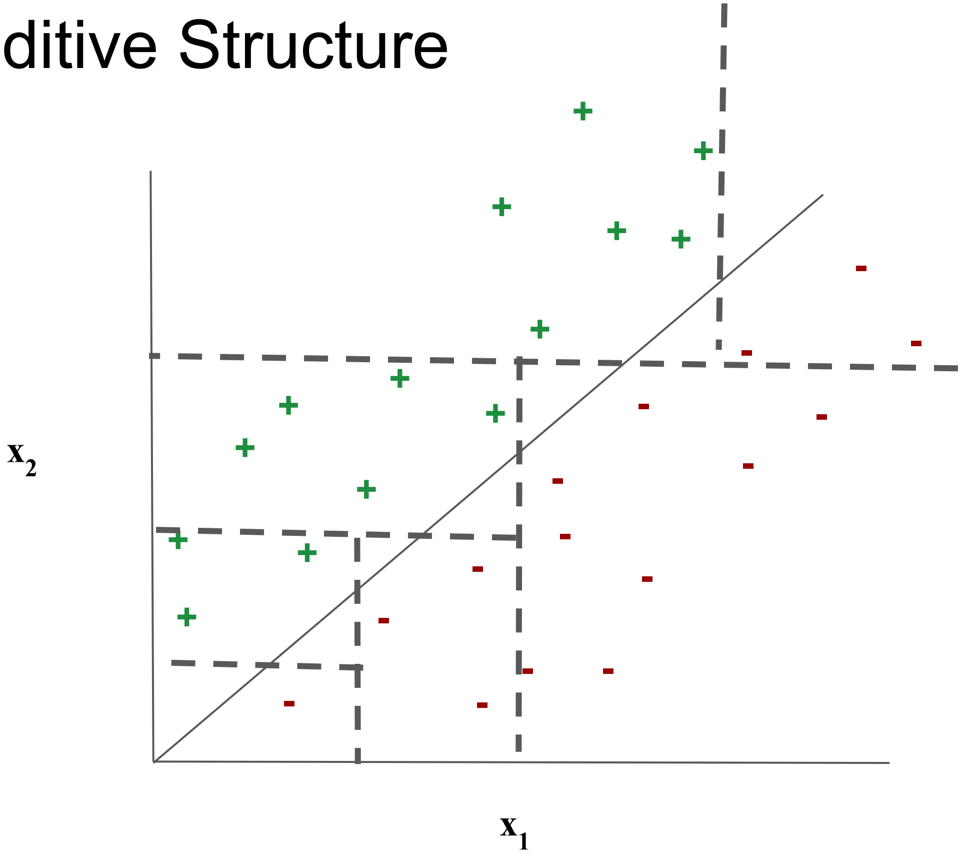
Regularization of Decision Trees

1. Min leaf size
2. Max Depth
3. Max Number of Nodes
4. Min decrease in loss (might stop too early)
5. Pruning:
   - Grow the whole tree
   - Iteratively collapse nodes that don’t increase loss
   - Use validation set

Runtime Complexity

Given	Test Time	Train Time
N Points	$\mathrm{O}(D)$ - depth	Each point belongs to one split in $\mathrm{O}(D)$ nodes
F features	$D < \log_2 N$	Cost of each point-split is proportionate to $\mathrm{O}(F)$
D Depth		Total Cost $\mathrm{O}(NFD)$

No Additive Structure



Pros and Cons of Decision Trees

+ Easy to explain (interpretable)

+ Categorical Variables

+ Fast

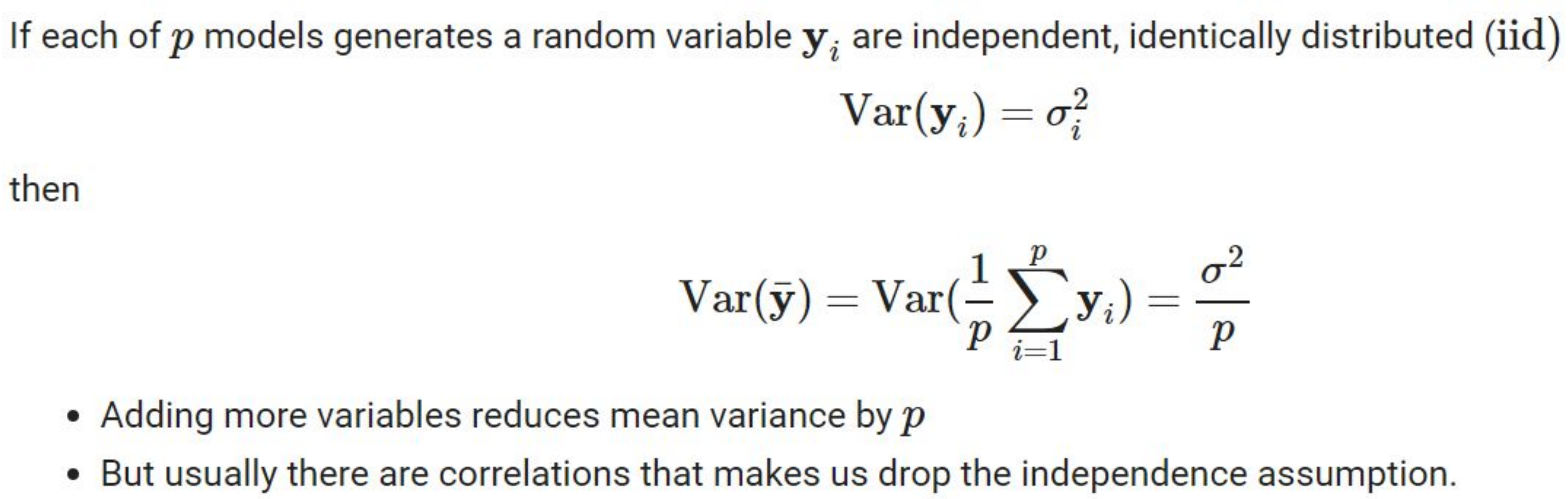
- High Variance (prone to overfit)

- Bad at additive structure

- Low predictive accuracy

Ensemble Methods

Ensemble methods



If each of p models generates a random variable $\mathbf{y}_i$ are independent, identically distributed (iid)

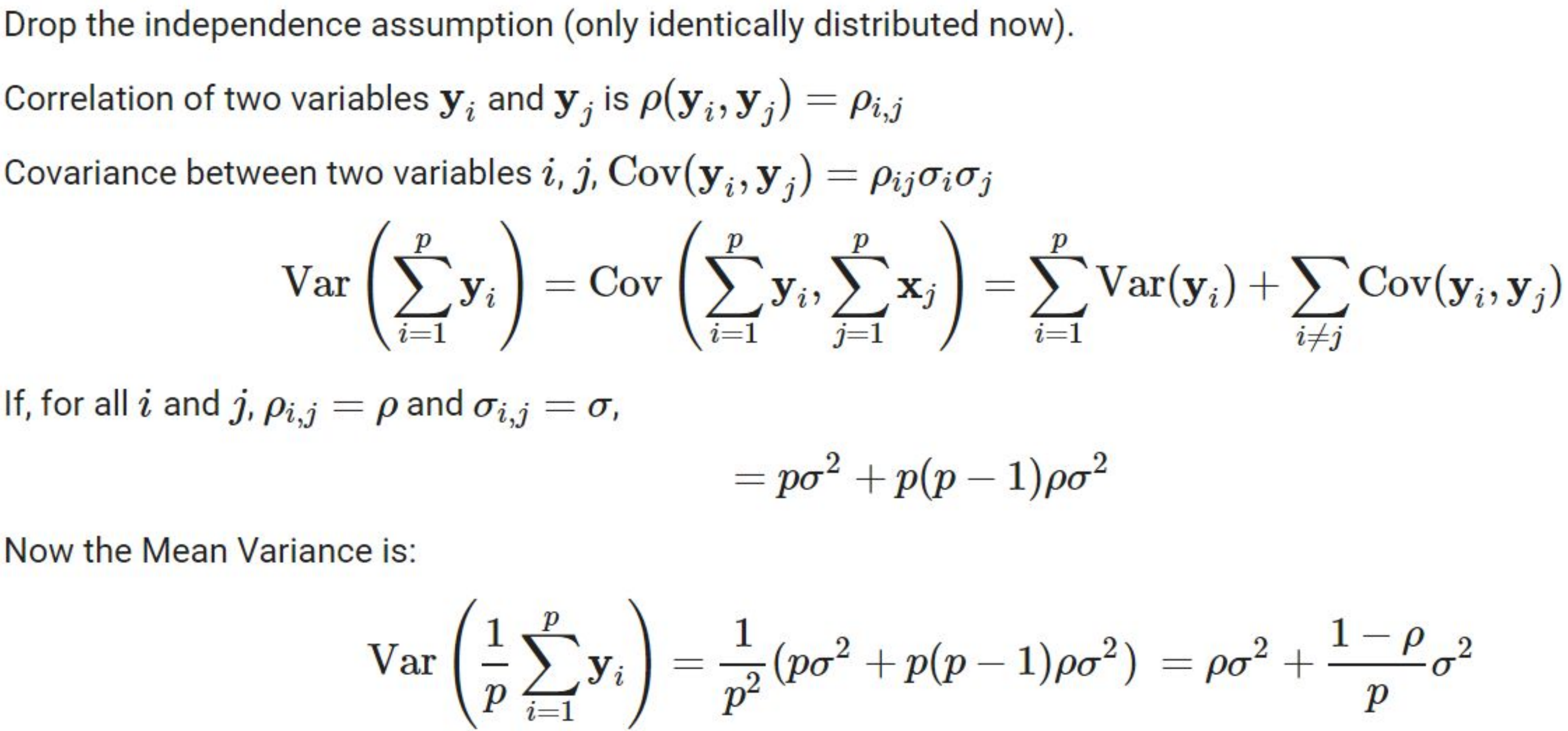
$$\mathrm{Var}(\mathbf{y}_i) = \sigma_i^2$$

then

$$\mathrm{Var}(\bar{\mathbf{y}}) = \mathrm{Var}(\frac{1}{p}\sum_{i=1}^{p}\mathbf{y}_i) = \frac{\sigma^2}{p}$$

- Adding more variables reduces mean variance by p
- But usually there are correlations that makes us drop the independence assumption.

Ensemble Methods



Drop the independence assumption (only identically distributed now).

Correlation of two variables $\mathbf{y}_i$ and $\mathbf{y}_j$ is $\rho(\mathbf{y}_i, \mathbf{y}_j) = \rho_{i,j}$

Covariance between two variables i, j, $\mathrm{Cov}(\mathbf{y}_i, \mathbf{y}_j) = \rho_{ij}\sigma_i\sigma_j$

$$\mathrm{Var}\left(\sum_{i=1}^{p}\mathbf{y}_i\right) = \mathrm{Cov}\left(\sum_{i=1}^{p}\mathbf{y}_i, \sum_{j=1}^{p}\mathbf{x}_j\right) = \sum_{i=1}^{p}\mathrm{Var}(\mathbf{y}_i) + \sum_{i\neq j}\mathrm{Cov}(\mathbf{y}_i, \mathbf{y}_j)$$

If, for all i and j, $\rho_{i,j} = \rho$ and $\sigma_{i,j} = \sigma$,

$$= p\sigma^2 + p(p-1)\rho\sigma^2$$

Now the Mean Variance is:

$$\mathrm{Var}\left(\frac{1}{p}\sum_{i=1}^{p}\mathbf{y}_i\right) = \frac{1}{p^2}(p\sigma^2 + p(p-1)\rho\sigma^2) \;= \rho\sigma^2 + \frac{1-\rho}{p}\sigma^2$$

Ensemble Methods

Two approaches for reducing variance error in models:

$$\mathrm{Var}\left(\frac{1}{p}\sum_{i=1}^{p}\mathbf{y}_i\right) = \rho\sigma^2 + \frac{1-\rho}{p}\sigma^2$$

“*Decorrelate*” multiple models, reducing ρ

Increase the number of models, which increases p

Ways to Ensemble

1. Different Algorithms
2. Different Training Sets
3. Bagging (Approximate Different Training Sets)
   - Random Forest
4. Boosting (We’ll skip)
   - Adaboost
   - XGBoost

Bootstrap Methods

Bagging - Bootstrap Aggregation

Have a true population P

Training Set samples from P, $S \sim P$

Assume population is the training sample $P = S$

Bootstrap samples $Z_1, Z_2, \ldots Z_M \sim S$ (with replacement)

Train separate models G_m on Bootstrap sample Z_m, then average:

$$G(\boldsymbol{x}_i) = \frac{\sum_{m=1}^{M} G_m(\boldsymbol{x})}{M}$$

Ensemble Methods

Two approaches for reducing variance error in models:

$$\mathrm{Var}\left(\frac{1}{p}\sum_{i=1}^{p}\mathbf{y}_i\right) = \rho\sigma^2 + \frac{1-\rho}{p}\sigma^2$$

With bootstrapped samples, you’re increasing bias

“*Decorrelate*” multiple models, reducing ρ

Increase the number of models, which increases p

Random Forest

Random Forest = Decision Trees + Bagging

DT have a high variance, low bias

Good fit for bagging

To further decorrelate (decrease ρ), at each split consider only a fraction of the features:

- Prevents all models from making the same splits - and decorrelates individual trees

Random Forest Algorithm

Algorithm 15.1 Random Forest for Regression or Classification

```
**Train**($M$, $n_{\text{min}}$, $loss$ = {cross-entropy or gini index}):
1.  For $m$ = 1 to $M$:
    a.  Draw bootstrap sample $Z$ of size $N$ from the training data.
    b.  Grow tree $T_m$ from $Z$, by recursively until minimum node size, $n_{min}$, is reached
          i.  Randomly select $p'$ variables from $p$ variables
         ii.  Pick the best split point $s$ among the $p'$ variables using loss function
        iii.  Split node into 2 daughter nodes
2.  Return ensemble of $M$ trees $\{T_m\}$
```

Random Forest Algorithm

Algorithm 15.1 Random Forest for Regression or Classification

Predict($\boldsymbol{x}$):

Regression:

$$\hat{f}_{rf}^{(M)}(\boldsymbol{x}) = \frac{1}{M}\sum_{m=1}^{M} T_m(\boldsymbol{x})$$

Classification:

$$\hat{C}_{rf}^{(M)}(\boldsymbol{x}) = \text{majority vote}\{\hat{C}_m(\boldsymbol{x})\}_1^M$$

Readings

Hastie 12.1-12.3 Support Vector Machines