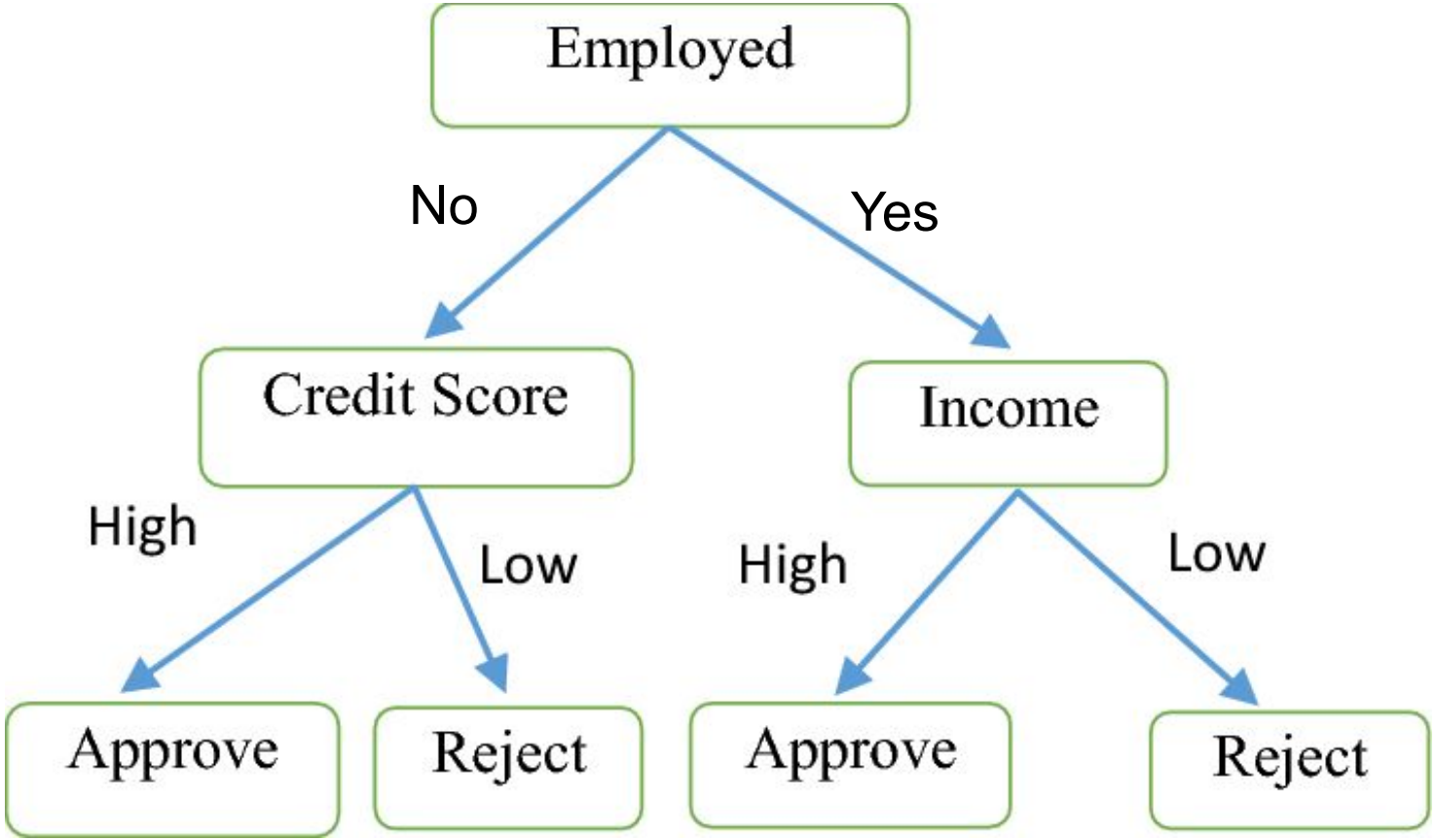
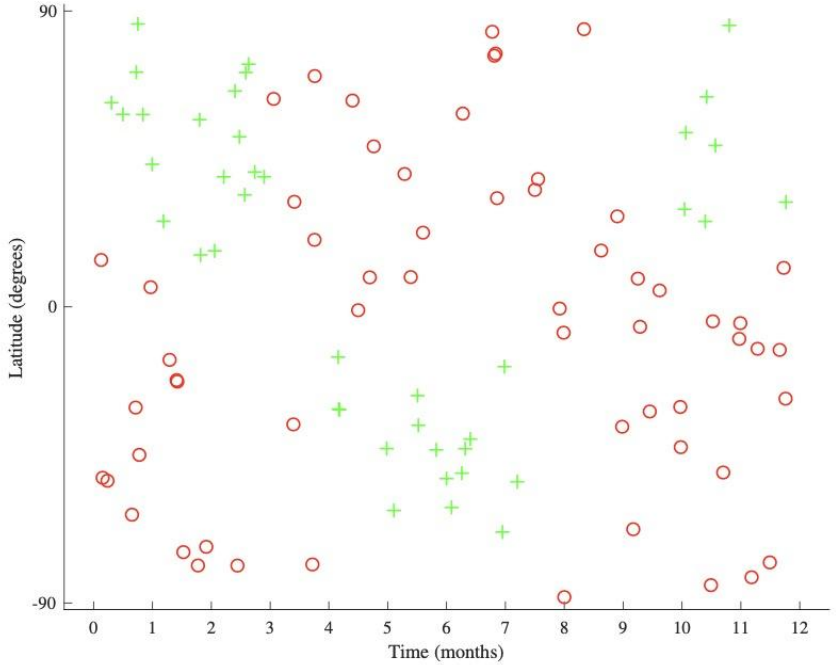
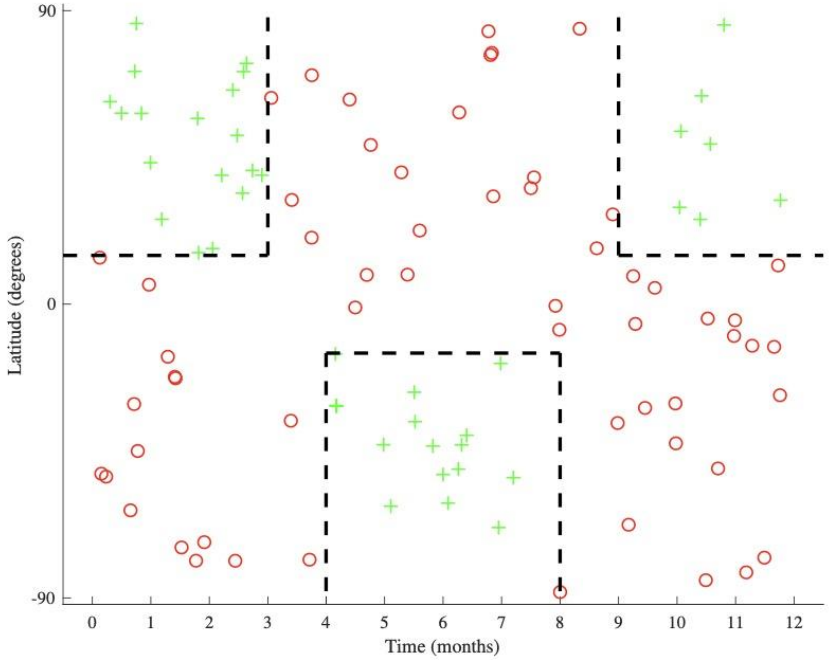
Decision Tree

Vishnuram Ayyavu Vijayakumar


Employed
No
Yes
Credit Score
Income
High
Low
High
Low
Approve
Reject
Approve
Reject

Is it possible to ski?

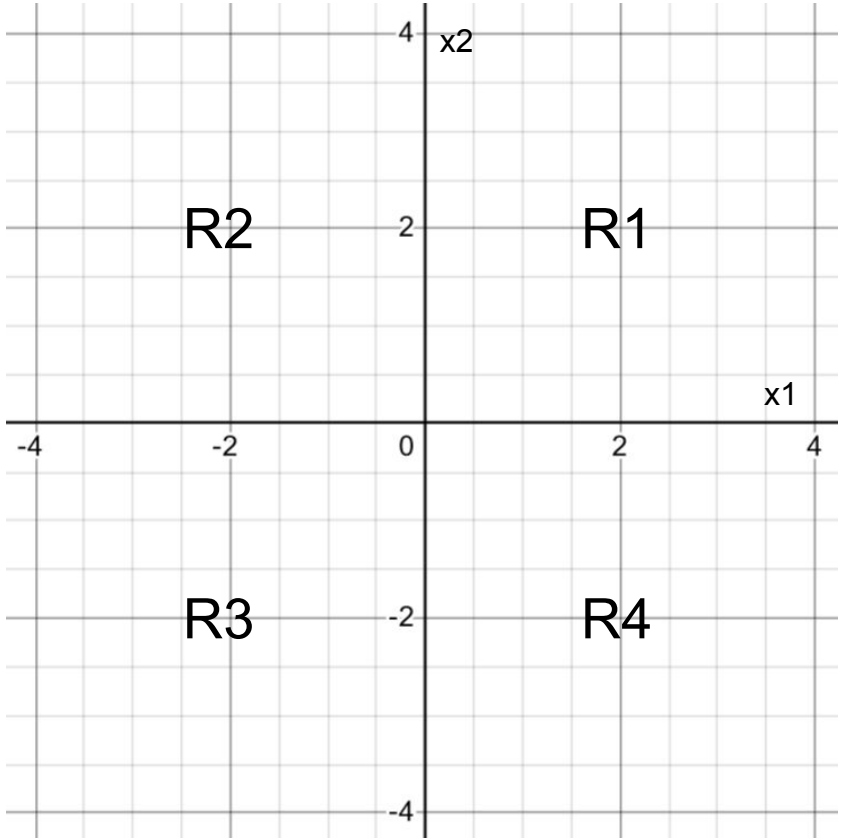




Partition the input domain into disjoint regions R_i such that the union of these regions recovers the entire input domain.

$$\mathcal{X} = \bigcup_{i=0}^{n} R_i$$

$$\text{s.t.} \qquad R_i \cap R_j = \emptyset \ \text{ for } \ i \neq j$$

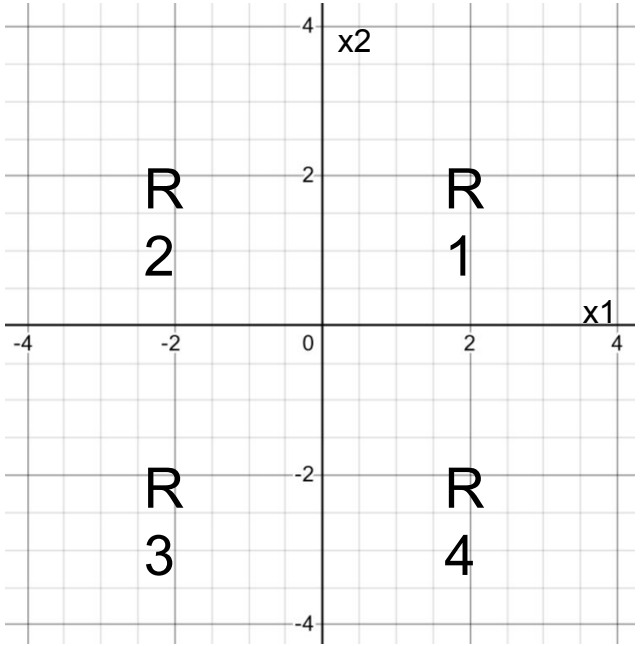


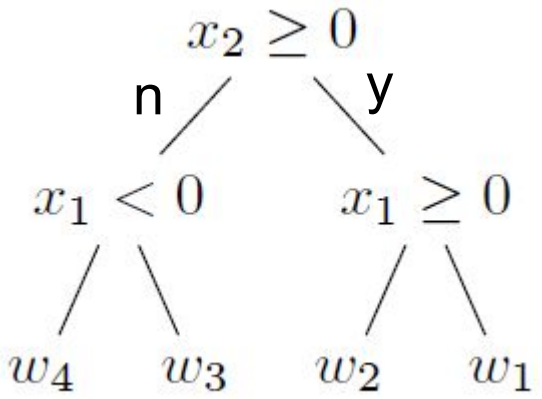
Input domain: $\mathbb{R}^2$

$$R_1 = \{x : x_1 \geq 0, x_2 \geq 0\},$$
$$R_2 = \{x : x_1 < 0, x_2 \geq 0\},$$
$$R_3 = \{x : x_1 < 0, x_2 < 0\},$$
$$R_4 = \{x : x_1 \geq 0, x_2 < 0\},$$

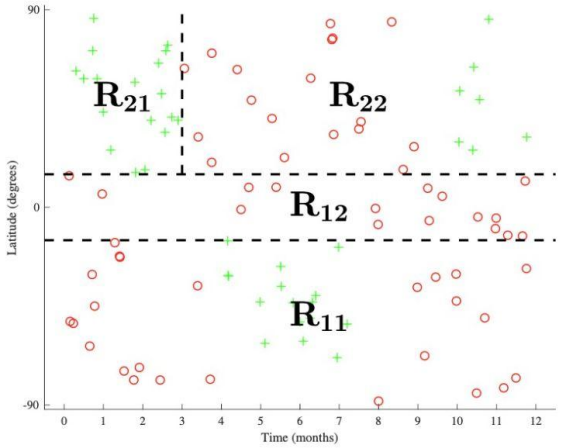
$$f(x) = \begin{cases} w_1 & \text{if } x \in R_1, \\ w_2 & \text{if } x \in R_2, \\ \vdots & \vdots \\ w_k & \text{if } x \in R_k. \end{cases}$$

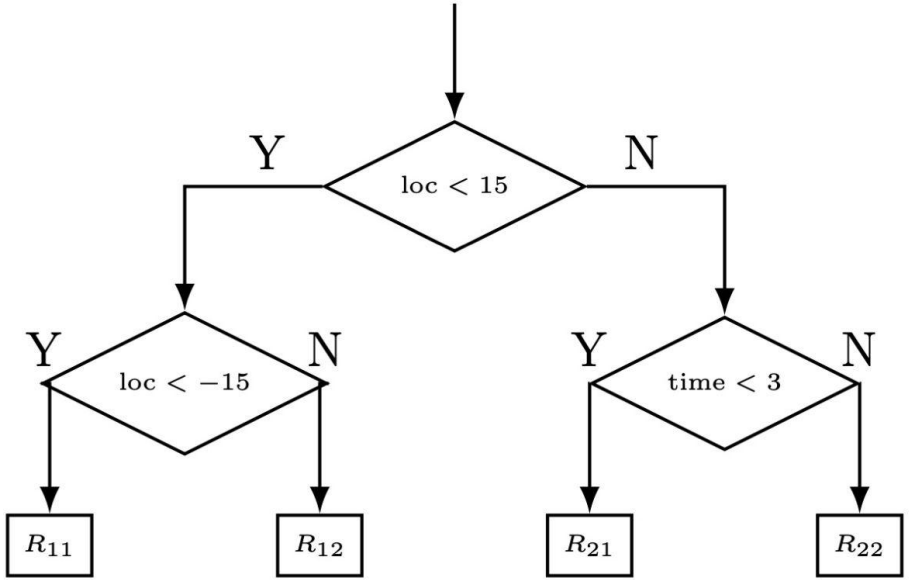
Mapping represented as trees





Mapping represented as trees

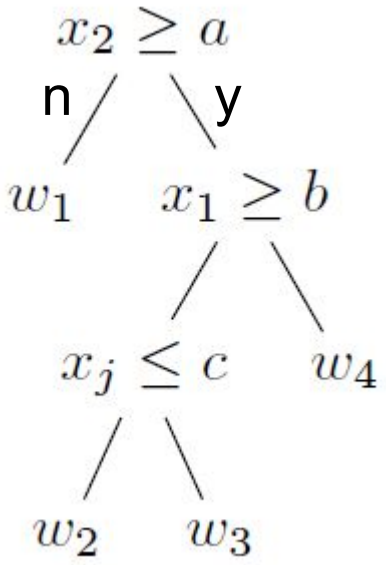




Exercise: Draw out the resulting regions in R^2

Given:

- d = 2
- a,b,c ∈ R
- j ∈ {1, 2}



To fit the model

$$\mathcal{L}(f) = \sum_{i=1}^{n} \ell\big(y^{(i)}, f(x^{(i)})\big) = \sum_{j=1}^{k} \sum_{x^{(i)} \in R_j} \ell(y^{(i)}, w_j).$$

Greedy approach to fit the model: Maximize the Gain Function

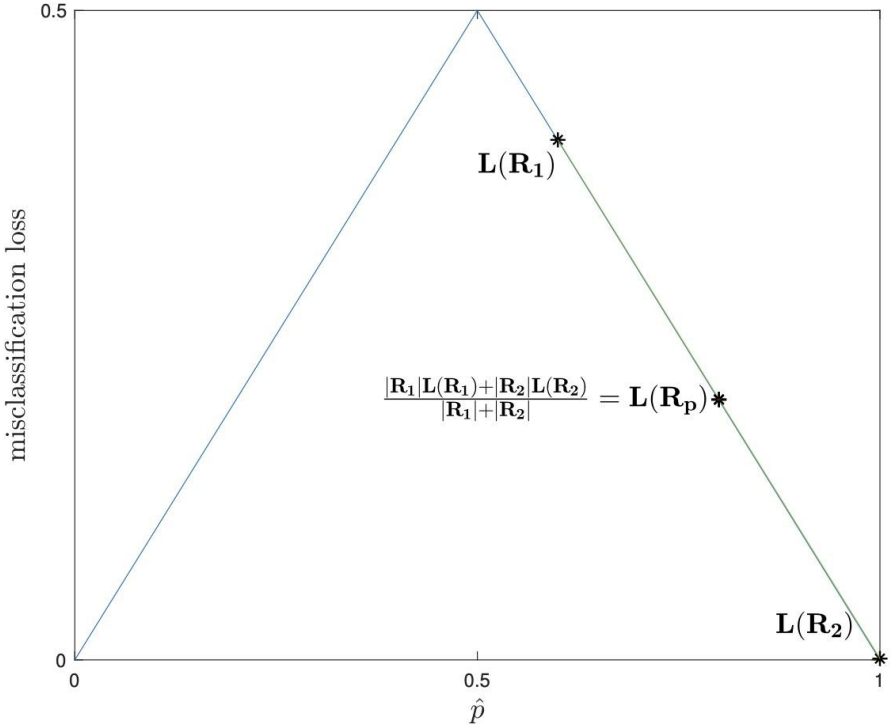
$$j, \theta = \arg\max_{j,\theta} G(j, \theta).$$

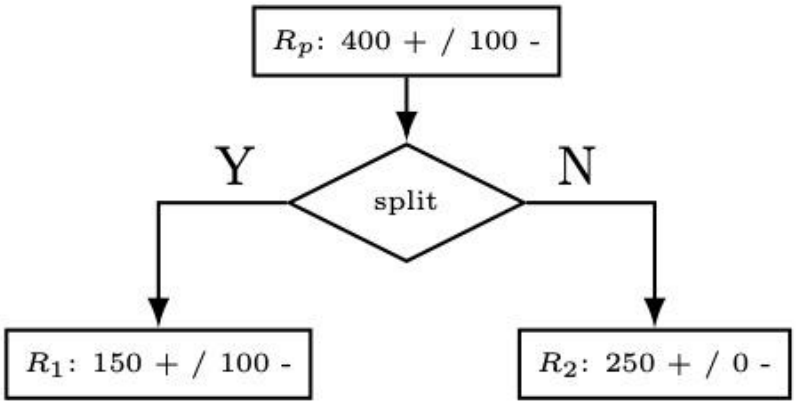
- Gain function: G(j, θ)
- j ∈ {1, . . . , d}
- θ ∈ R

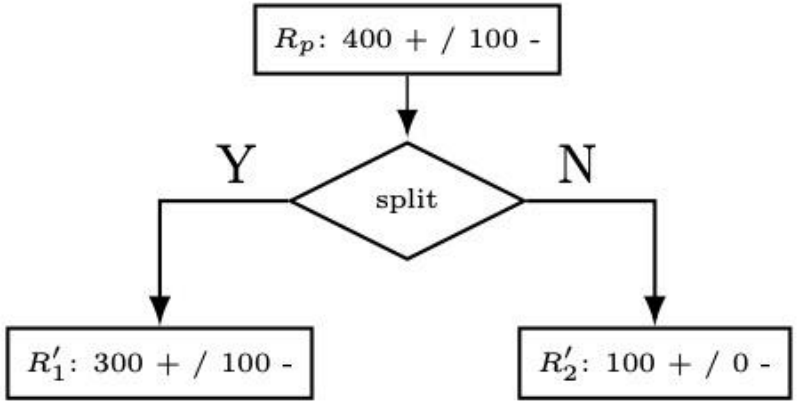
Maximizing gain by minimizing cost

$$G(j,\theta) = C(S) - \left[\frac{|L|}{|S|}C(L) + \frac{|R|}{|S|}C(R)\right]$$

$$L_{\text{misclass}}(R) = 1 - \max_{c}(\hat{p}_c)$$




R_p: 400 + / 100 -
Y
split
N
R_1: 150 + / 100 -
R_2: 250 + / 0 -


R_p: 400 + / 100 -
Y
split
N
R'_1: 300 + / 100 -
R'_2: 100 + / 0 -

Misclassification Loss Analysis

Parent Region:

- **Positive Examples (+):** 400
- **Negative Examples (-):** 100
- **Total Examples:** 500
- **Misclassification Loss:** $L_{\text{misclass}}(\text{Parent}) = 1 - \max(0.8, 0.2) = 0.2$

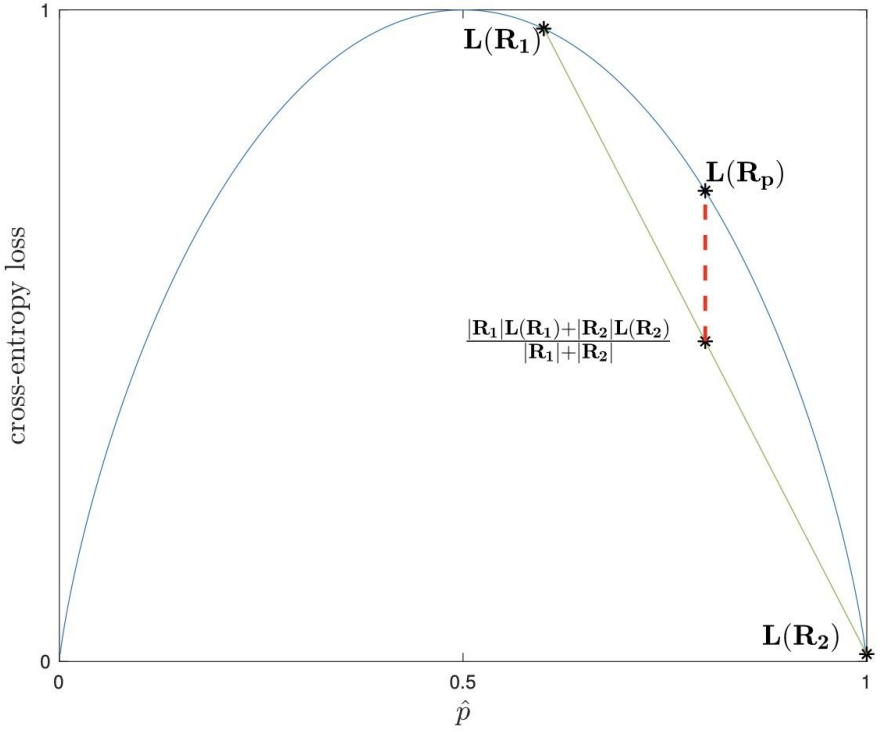
Split 1:

- **Child Region 1:** 150 (+), 100 (-)
  - **Misclassification Loss:** $L_{\text{misclass}}(\text{R1}) = 1 - \max(0.6, 0.4) = 0.4$
- **Child Region 2:** 250 (+), 0 (-)
  - **Misclassification Loss:** $L_{\text{misclass}}(\text{R2}) = 1 - \max(1, 0) = 0$
- **Total Misclassification Loss After Split 1:** $\frac{250}{500} \times 0.4 + \frac{250}{500} \times 0 = 0.2$

Split 2:

- **Child Region 1:** 300 (+), 100 (-)
  - **Misclassification Loss:** $L_{\text{misclass}}(\text{R1}) = 1 - \max(0.75, 0.25) = 0.25$
- **Child Region 2:** 100 (+), 0 (-)
  - **Misclassification Loss:** $L_{\text{misclass}}(\text{R2}) = 1 - \max(1, 0) = 0$
- **Total Misclassification Loss After Split 2:** $\frac{400}{500} \times 0.25 + \frac{100}{500} \times 0 = 0.2$

$$L_{cross}(R) = -\sum_c \hat{p}_c \log_2 \hat{p}_c$$



For regression:

$$\hat{y} = \frac{\sum_{i \in R} y_i}{|R|}$$

$$L_{\text{squared}}(R) = \frac{\sum_{i \in R} (y_i - \hat{y})^2}{|R|}$$

Regularization

Problems arising when we let the greedy algorithm described above run indefinitely:

- Overfitting to training data
- Poor generalization

Regularization methods

- Limiting the number of leaves (distinct regions) in the model.
- barring additional splits when there are too few datapoints in a node.
- not splitting if the gain described above is beneath some threshold.
- **Pruning**

Advantages and disadvantages

Advantages:

- Highly interpretable
- Pretty good fit relatively quickly, even on large datasets

Disadvantages:

- Lack of Additive Structure
- Poor generalization

Lack of Additive structure