Week 8

↳ Grades (mostly) up to date

↳ Midterm 10/21

- practice exams out
- one double sided cheat-sheet
- Prof Cass/Rosenman will proctor
- No office hours Monday or Wednesday (Teal @ conference)

Review: Supervised Learning

Linear Algebra	Page Rank	Eigendecompositions, power method
Linear Regression	Exact Optimization	Regression tasks, first optimizer
Gradient Descent	Polynomial Regression	General optimizer, more expressive
Probability	Logistic Regression	Classification, Bayes rule, linear model
[Support Vector Machines	Constrained Optimization]	De-emphasize
Kernel Methods	Neural Nets	Efficient feature transform, Both transform AND fit
Convolution Nets	Transformers	Architecture for spatial data, sequential data

Highlights:

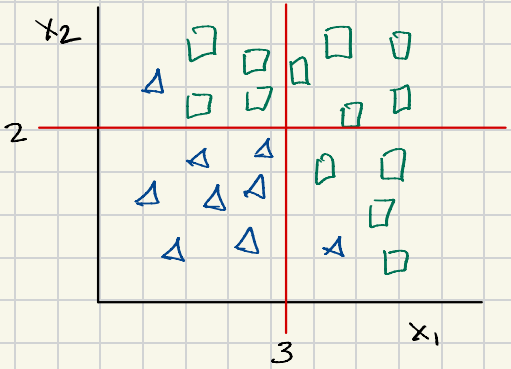
↳ Math foundations

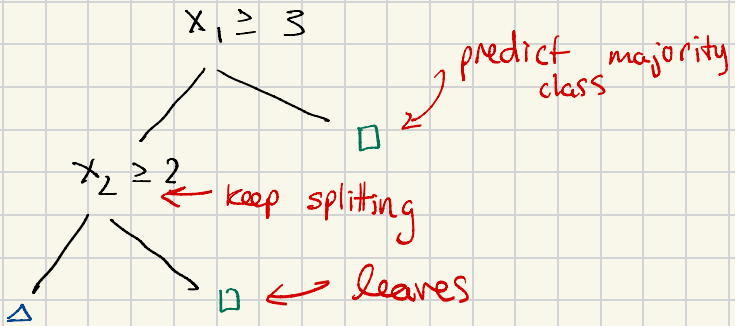
↳ Building generative AI tools

↳ Practice how to learn

Decision Trees

A simpler take on supervised learning





How accurate is the tree?

How would we make the tree more accurate?

Measuring Error

$$E_\ell[\mathcal{L}(\ell)] = \sum_{\ell} \mathcal{L}(\ell)\, p^{(\ell)}$$

ℓ ← leaves; $\mathcal{L}(\ell)$ ← loss on leaf; $p^{(\ell)}$ ← proportion of data that reach leaf

Exercise: Compute expected error of DT

Error Rate

$$\mathcal{L}_{\text{error}}(\ell) = 1 - p_0^{(\ell)}$$

← W2OG, suppose class 0 is majority

Gini Impurity

"Probability of misclassification if we randomly assigned labels by class proportion"

$$\mathcal{L}_{\text{Gini}}(\ell) = \sum_c p_c^{(\ell)} (1 - p_c^{(\ell)})$$

← class c

Information Gain

$$\mathcal{L}_{IG}(\ell) = \text{Entropy}(\ell) = -\sum_c p_c^{(\ell)} \log p_c^{(\ell)}$$

Optimization

Search over splits to find best

Regression

How do we make a regression tree?

Going forward: trees are weak learners, but

- ★ can be "boosted"
- ★ very efficient
- ★ very interpretable

Week 9

↳ Over half way! How was midterm?

↳ Anonymous mid-semester eval

- what's going well? what could be better?

↳ Talk on Monday 10/27 4:15 @ Estella 1021

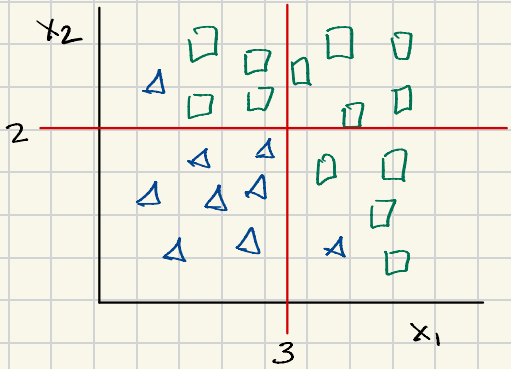
- my explainable AI research
- application of gradient boosted trees!

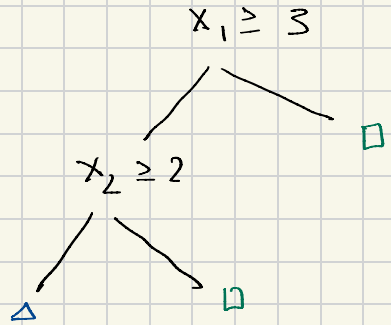
Context

↳ Neural nets very versatile

↳ Simpler method gave SOTA on tabular data (vector input, real number output)

Decision Trees





- Efficient
- Interpretable

... Let's boost them!

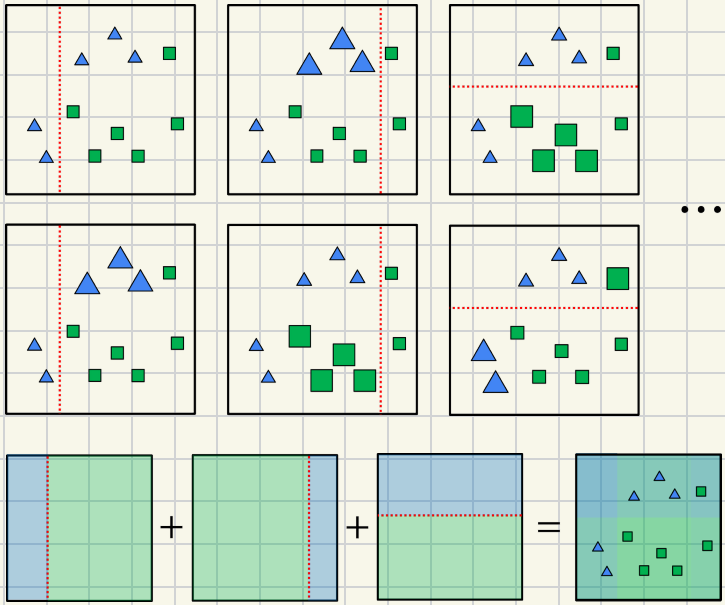
Adaptive Boosting (Ada Boost)

CLASSIFICATION ← $y \in \{-1, 1\}$

Idea: Use many trees, each one fixes the mistakes of the prior ones

$f_t : \mathbb{R}^d \to \{-1, 1\}$ is the t^{th} tree

$F_t : \mathbb{R}^d \to \mathbb{R}$ $\quad F_t(x) = \alpha_1 f_1(x) + \dots \alpha_{t-1} f_{t-1}(x) + \alpha_t f_t(x) = F_{t-1}(x) + \alpha_t f_t(x)$



Exponential Loss: $\mathcal{L}(F_t) = \sum_{i=1}^{n} e^{-y^{(i)} F_t(x^{(i)})}$ $\quad < 1$ if $\text{sign}(F_t(x^{(i)})) = y^{(i)}$, ≥ 1 else

Goal: Disentangle f_t from F_t, so we can optimize it

$$\mathcal{L}(F_t) = \sum_{i=1}^{n} e^{-y^{(i)}\left[F_{t-1}(x^{(i)}) + \alpha_t f_t(x^{(i)})\right]} = \sum_{i=1}^{n} e^{-y^{(i)} F_t(x^{(i)})} e^{-y^{(i)} \alpha_t f_t(x^{(i)})}$$

$w_{t-1}^{(i)}$ "How much error we made on i"

$$= \sum_{i=1}^{n} w_{t-1}^{(i)} e^{-y^{(i)} \alpha_t f_t(x^{(i)})} = \sum_{i: y^{(i)} = f_t(x^{(i)})} w_{t-1}^{(i)} e^{-\alpha_t} + \sum_{i: y^{(i)} \neq f_t(x^{(i)})} w_{t-1}^{(i)} e^{\alpha_t}$$

$$= \sum_{i=1}^{n} w_{t-1}^{(i)} e^{-\alpha_t} + \left(e^{\alpha_t} - e^{-\alpha_t}\right) \sum_{i: y^{(i)} \neq f_t(x^{(i)})} w_{t-1}^{(i)}$$

↖ only place f_t appears!

min $\mathcal{L}$ equiv to min weighted error

Choosing Ensemble Importances

Idea: Once f_t fixed, choose α_t to minimize loss

$$\mathcal{L}(F_t) = \sum_{i : y^{(i)} = f_t(x^{(i)})} w_{t-1}^{(i)} e^{-\alpha_t} + \sum_{i : y^{(i)} \neq f_t(x^{(i)})} w_{t-1}^{(i)} e^{+\alpha_t}$$

Gradient Boosting

REGRESSION

Idea: Run gradient descent on predictions rather than parameters

$f_t : \mathbb{R}^d \to \mathbb{R}$

$F_t : \mathbb{R}^d \to \mathbb{R}$ $\quad F_t(x) = \alpha_1 f_1(x) + \dots + \alpha_t f_t(x) = F_{t-1}(x) + \alpha_t f_{t-1}(x)$

$$\underset{f_t}{\operatorname{argmin}} \sum_{i=1}^{n} \mathcal{L}\left(y^{(i)}, F_{t-1}(x^{(i)}) + \alpha_t f_t(x^{(i)})\right)$$

Gradient Descent: $\quad F_t(x^{(i)}) \leftarrow F_{t-1}(x^{(i)}) - \alpha \dfrac{\partial \mathcal{L}(y^{(i)}, F_{t-1}(x^{(i)}))}{\partial F_{t-1}(x^{(i)})}$

learning rate

Choose f_t to approx this

Suppose $\mathcal{L}(y, F_{t-1}(x)) = \frac{1}{2}(F_{t-1}(x) - y)^2$

$$\frac{\partial \mathcal{L}}{\partial F_{t-1}(x)} = F_{t-1}(x) - y = \text{residual} \approx f_t(x)$$

How do we choose α?

Tab PFN

Xgboost, LGBM, etc were SOTA from mid-2010s to 2025

Now, Tab PFN can outperform with in-context learning

Like ChatGPT or Gemini, give training data and unlabelled test data as prompt

<10sec on Tab PFN outperforms 4 hours of xgboost (small + medium datasets)

Trained on 130 million synthetic datasets