EECS 553 Lecture 21
Machine Learning (ECE)
[class version]

Jeff Fessler

University of Michigan
©Laura Balzano and Jeff Fessler, UM, November 11, 2026, 08:18

F26

Logistics

© 2026 Fessler

- HW 7 due Wed. Nov. 11
- Exam 2: Mon. Nov. 16, 6-8PM; 1013 Dow; 2 sheets
  details in previous lectures

Recap / Outline

© 2026 Fessler

- Recap
  - GMM, EM
- Today
  - Gaussian mixture model (GMM) Recap
    - EM for GMM
  - Decision trees
    - Classification trees
    - Regression trees

K-Means is a centroid-based algorithm.

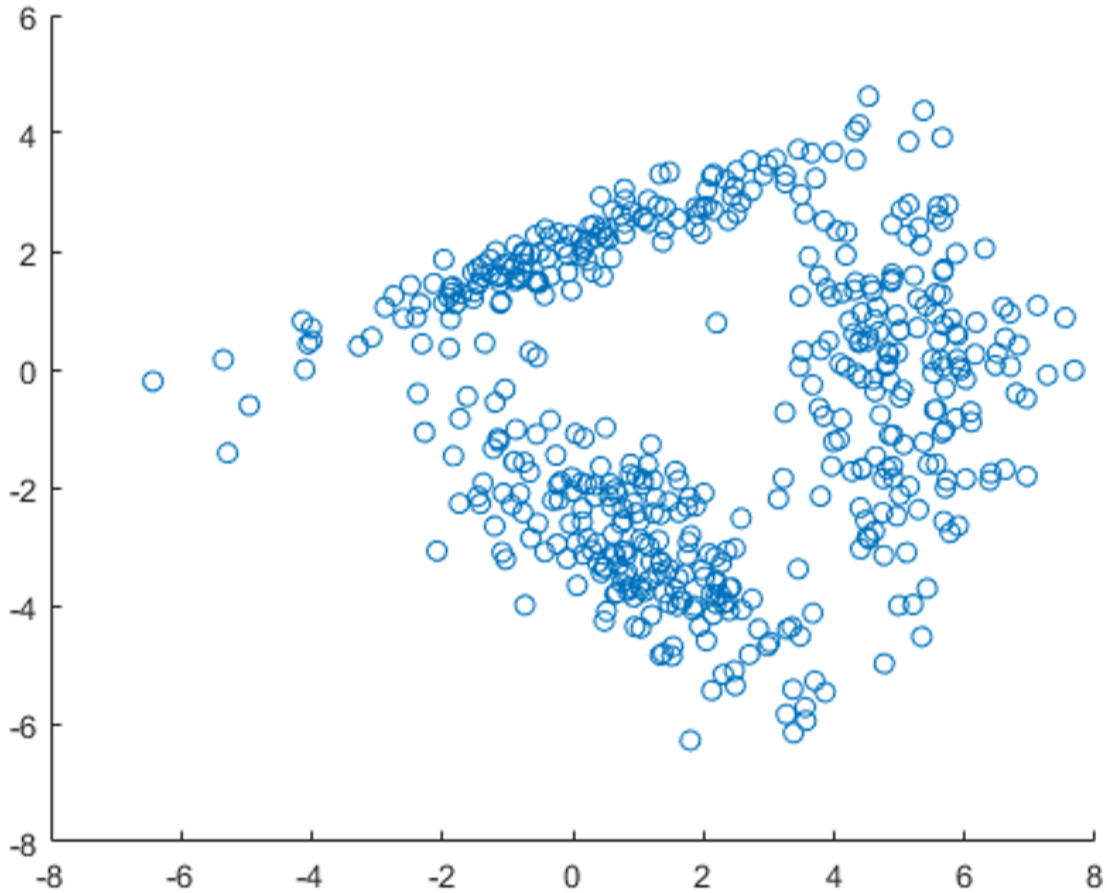


Figure courtesy Clay Scott.

Another approach is to assume a distributional (generative) model.

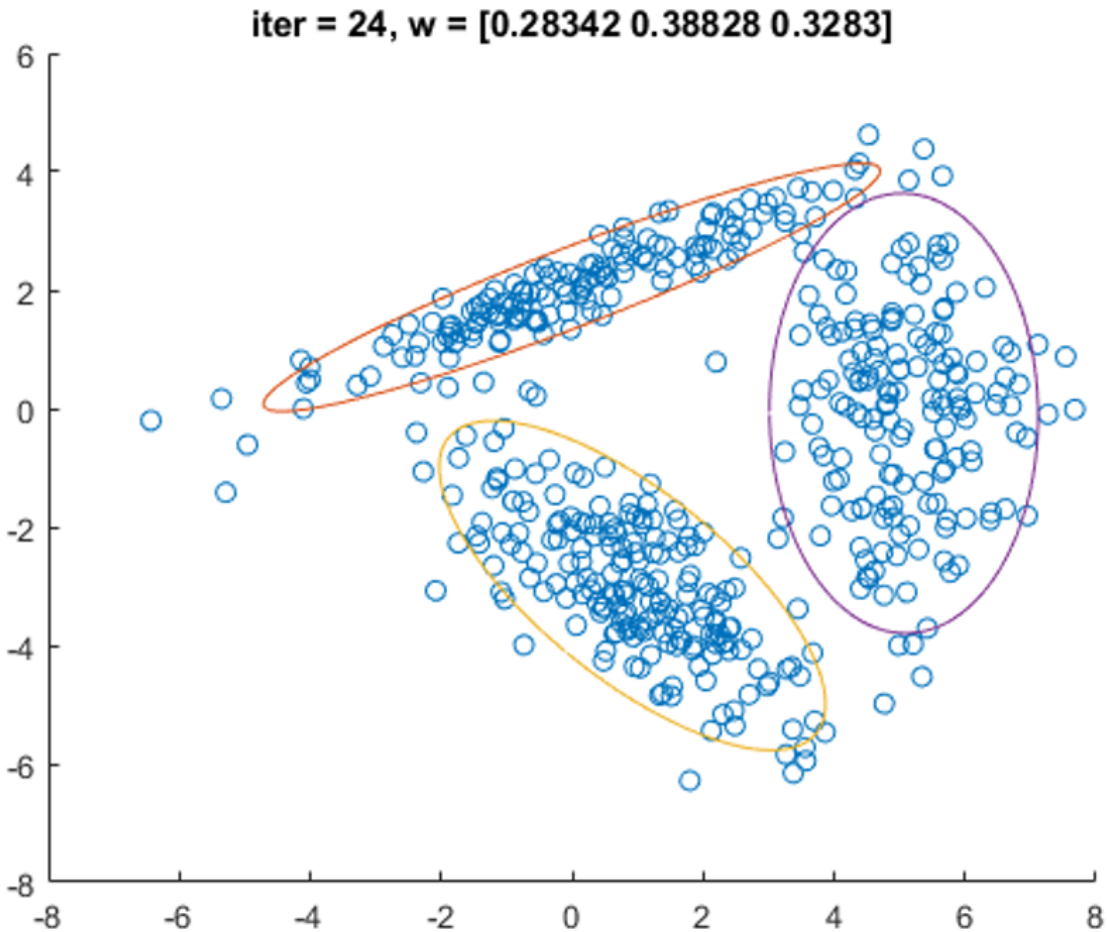


Figure courtesy Clay Scott.

A **Gaussian mixture model** with K components has the form

$$p(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{k=1}^{K} w_k \underbrace{\phi(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}_{\text{Normal pdf}}.$$

Q21.1 What constraints do we impose on the GMM parameters?

A: $w_k \geq 0$, $\sum_k w_k = 1$, $\boldsymbol{\mu}_k \geq \mathbf{0}$ and $\boldsymbol{\Sigma}_k \succ \mathbf{0}$.

B: $w_k, \boldsymbol{\mu}_k$ unconstrained and $\boldsymbol{\Sigma}_k \succ \mathbf{0}$

C: $w_k \geq 0$, $\sum_k w_k = 1$, and $\boldsymbol{\Sigma}_k \succ \mathbf{0}$.

D: $w_k \geq 0$, $\boldsymbol{\mu}_k \geq \mathbf{0}$, and $\boldsymbol{\Sigma}_k \succ 0$.

E: none of these

GMM log-likelihood

© 2026 Fessler

- Observed data $\mathcal{D} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$
- The log-likelihood for the GMM model is

$$\ell(\boldsymbol{\theta}; \mathcal{D}) \triangleq \sum_{i=1}^{n} \log \left(\sum_{k=1}^{K} w_k \, \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right).$$

- When $K > 1$, there is no closed-form, maximum-likelihood estimate of $\boldsymbol{\theta}$.
- (When $K = 1$ we use the closed-form maximum-likelihood estimates of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ in LDA and HW.)

Expectation maximization (EM): Recap

© 2026 Fessler

- Design/select latent variable $\underline{\boldsymbol{Z}} = (\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_n)$ for which E-step and M-step below are tractable.
- Initialize $\boldsymbol{\theta}_0$
- $t \leftarrow 0$
- `Repeat`

  E-step: "Compute" the expectation:

  $$Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t) = \mathbb{E}_{\underline{\boldsymbol{Z}} \sim \mathsf{p}(\underline{\boldsymbol{z}} | \underline{\boldsymbol{x}}; \boldsymbol{\theta}_t)} \left\{ \tilde{\ell}(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) \right\}$$

  M-step: Perform the maximization:

  $$\boldsymbol{\theta}_{t+1} \leftarrow \arg\max_{\boldsymbol{\theta}} \; Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t)$$

  $t \leftarrow t + 1$
- `Until` convergence criterion satisfied.
- By design, likelihood increases monotonically.

EM for GMM: complete-data log-likelihood

© 2026 Fessler

When $\Delta_{i,k} = \mathbb{I}_{\{Z_i = k\}}$, we have $\sum_{k=1}^{K} \Delta_{i,k} a_k = a_{Z_i}$ so the complete-data log-likelihood becomes:

$$\ell(\boldsymbol{\theta}; \underline{\boldsymbol{x}}, \underline{\boldsymbol{z}}) = \log\Big(\prod_{i=1}^{n} \mathsf{P}\{Z_i = z_i; \boldsymbol{\theta}\} \, \mathsf{p}(\boldsymbol{x}_i \mid z_i; \boldsymbol{\theta}) \Big)$$

$$= \log\left(\prod_{i=1}^{n} w_{z_i} \, \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}) \right) = \sum_{i=1}^{n} \log\big(\underbrace{w_{z_i} \, \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i})}_{} \big)$$

$$= \qquad\qquad$$

$$= \qquad\qquad (1)$$

because $f(b_{Z_i}) = f(\sum_k \Delta_{i,k} b_k) = \sum_k \Delta_{i,k} f(b_k)$.

- Denote the current iterate $\boldsymbol{\theta}_t = (w_1^{(t)}, \ldots, w_K^{(t)}, \boldsymbol{\mu}_1^{(t)}, \ldots, \boldsymbol{\mu}_K^{(t)}, \boldsymbol{\Sigma}_1^{(t)}, \ldots, \boldsymbol{\Sigma}_K^{(t)})$
- The only random variable in (1) is $\Delta_{i,k} = \Delta_{i,k}(Z_i)$.
- So the E-step requires calculating these "soft class assignments:"

$$\gamma_{i,k}^{(t)} \triangleq \mathbb{E}_{\underline{\boldsymbol{Z}} \sim \mathsf{p}(\underline{\boldsymbol{z}} | \underline{\boldsymbol{x}}; \boldsymbol{\theta}_t)} \{\Delta_{i,k}\} = \mathbb{E}\{\Delta_{i,k} \mid \underline{\boldsymbol{X}} = \underline{\boldsymbol{x}}; \boldsymbol{\theta}_t\}$$

$$= \mathsf{P}\{\Delta_{i,k} = 1 \mid \underline{\boldsymbol{X}} = \underline{\boldsymbol{x}}; \boldsymbol{\theta}_t\} = \mathsf{P}\{Z_i = k \mid \underline{\boldsymbol{X}} = \underline{\boldsymbol{x}}; \boldsymbol{\theta}_t\}$$

$$= \qquad \text{(Bayes rule)}$$

$$= \frac{w_k^{(t)} \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{j=1}^K w_j^{(t)} \phi(\boldsymbol{x}_i; \boldsymbol{\mu}_j^{(t)}, \boldsymbol{\Sigma}_j^{(t)})} \in [0, 1]. \text{ "responsibility"}$$

- Note:

- The M-step for GMM requires maximization:

$$\boldsymbol{\theta}_{t+1} = \arg\max_{\boldsymbol{\theta}} \; Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t)$$

where from (1) and the E-step:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}_t) = \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{i,k}^{(t)}$$

- The w_k update involves the **unit simplex**:

$$w_k^{(t+1)} = \arg\max_{w_k \geq 0 \,:\, \sum_k w_k = 1} \sum_{i=1}^{n} \sum_{k=1}^{K}$$

- The $\boldsymbol{\mu}_k$ update involves a quadratic objective function:

$$\boldsymbol{\mu}_k^{(t+1)} = \arg\max_{\boldsymbol{\mu}_k} -\frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{i,k}^{(t)} \left(\boldsymbol{x}_i - \boldsymbol{\mu}_k\right)^\top \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)$$

$$\implies \nabla_{\boldsymbol{\mu}_k} Q(\cdot, \boldsymbol{\theta}_t) = \sum_{i=1}^{n}$$

$$\implies \boldsymbol{\mu}_k^{(t+1)} =$$

  which is a responsibility-weighted average (convex combination) of the data points.

- This is a "soft assignment" version of the centroid calculation in the k-means algorithm.

- The $\boldsymbol{\Sigma}_k$ update involves the determinant (cf HW):

$$\boldsymbol{\Sigma}_k^{(t+1)} = \arg\max_{\boldsymbol{\Sigma}_k} -\frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_{i,k}^{(t)} \left[\log |\boldsymbol{\Sigma}_k| + (\boldsymbol{x}_i - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k) \right]$$

$$= \cdots$$

$$=$$

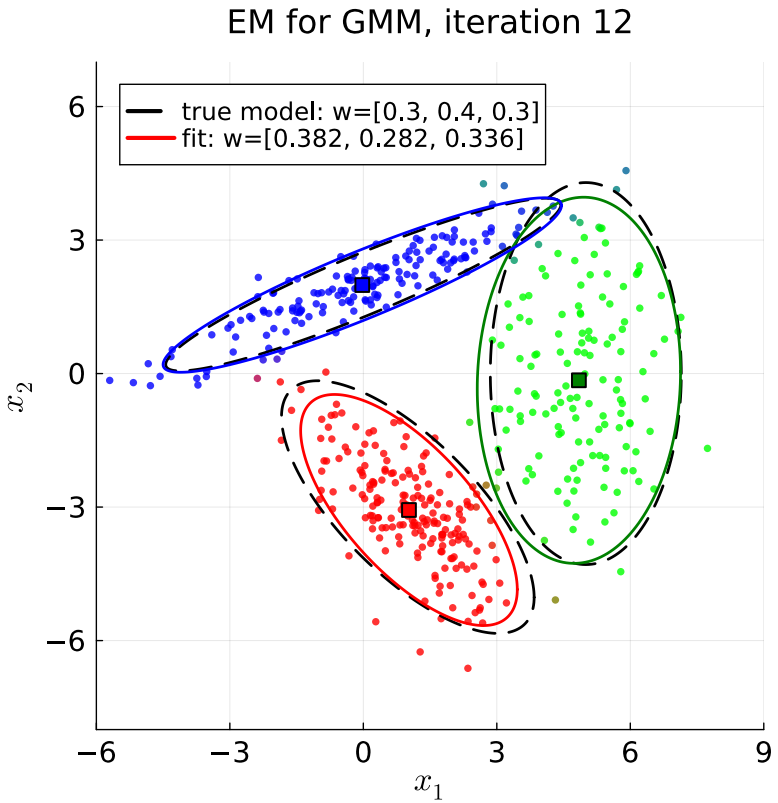
using some matrix derivatives from the matrix cookbook.

- This is a responsibility-weighted sample covariance.
- It uses the updated mean vectors $\{\boldsymbol{\mu}_k\}$.

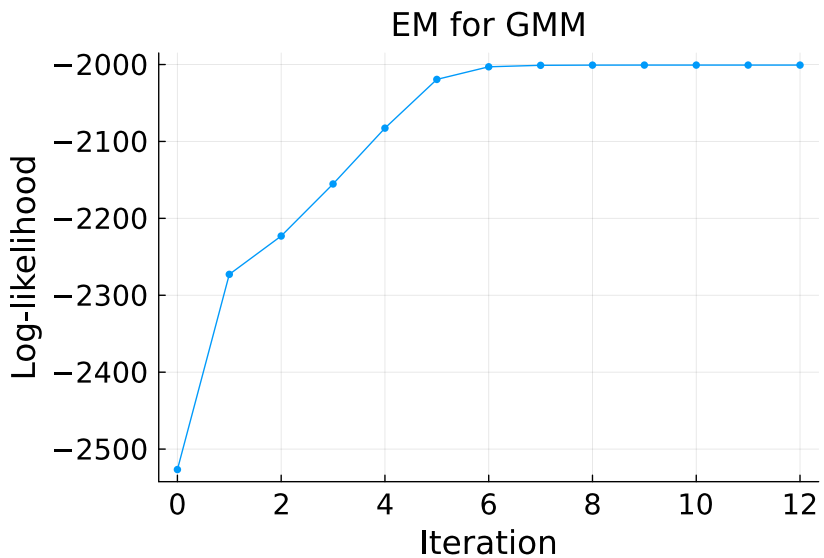
EM for GMM demo

© 2026 Fessler

Julia demo



Python demo



- RGB coloring of
- log-likelihood increases and eventually converges
- Unlike k-means, here we have "soft assignments"
- Like k-means, this involves a nonconvex optimization problem

Alternatives to EM for GMM?

© 2026 Fessler

Could we apply GD instead of EM for GMM parameter estimation?

-
-

Q21.2 The EM algorithm above guarantees that each $\boldsymbol{\Sigma}_k$ remains (positive semidefinite, invertible)?

A: (T,T) B: (T,F) C: (F,T) D: (F,F) **??**

- In GMM model fitting, the log-likelihood is .
- Solution: add

Clustering is a huge field

© 2026 Fessler

- We discussed
  - centroid based (*e.g.*, k-means)
  - distribution based (*e.g.*, GMM)

  See [2, p. 526] for one discussion of k-means vs GMM.
- Many other methods and research topics
  - hierarchical clustering
  - density-based clustering
  - graph clustering
  - clustering non-real-valued data
  - model selection
  - clustering quality metrics

Clustering and applications

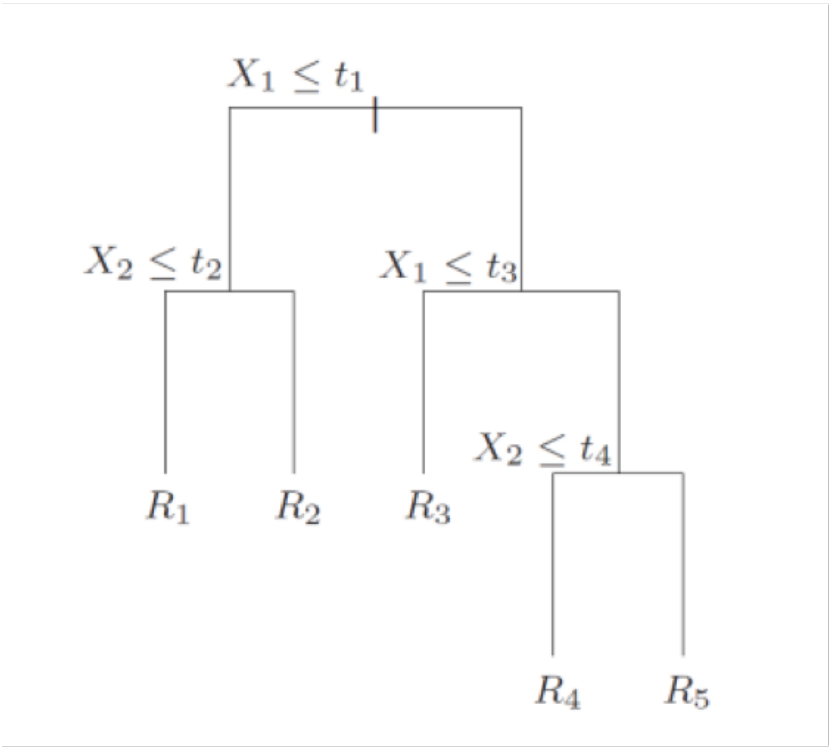
© 2026 Fessler

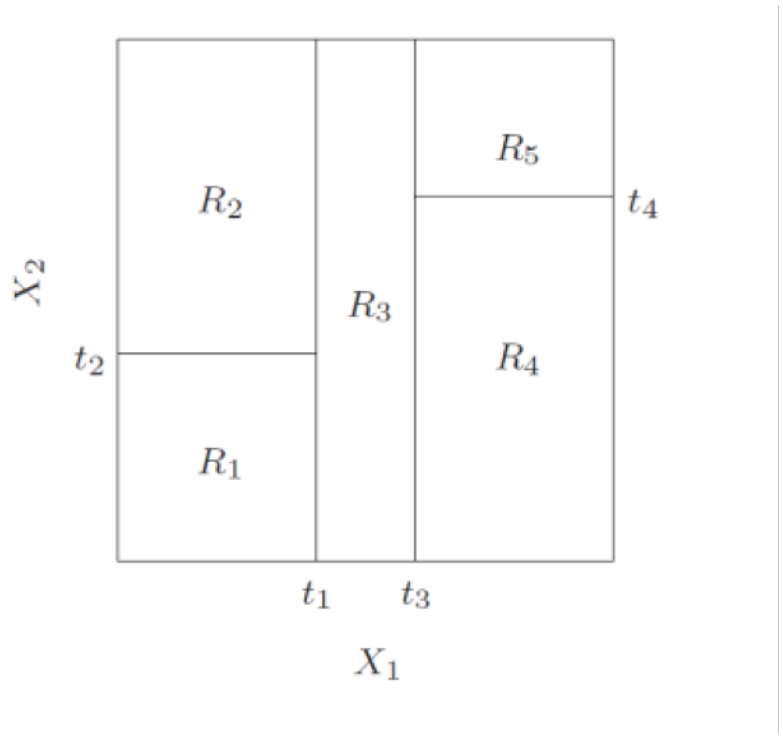
- The main goal of clustering is to discover interesting subpopulations.
- Examples:
  - Internet network traffic analysis: clustering users by what content they need
  - Marketing/sales: clustering customers
  - Clinical trials: clustering outcomes
  - Finance: cluster stocks
  - Bioinformatics: clustering cell types
- Clustering is often not the end of analysis. Clusters serve to generate hypotheses for subsequent analysis.

Tree structured partition of feature space

© 2026 Fessler

- Supervised learning with training data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$.
- **Decision trees** use **recursive partitioning**, aka a **tree-structured partition**, of feature space
- Focus on **partition** processes using one feature at a time.
- The root node is the whole feature space $\mathbb{R}^d$.





These $d = 2$ figures are courtesy [3].

CQ: Partition

© 2026 Fessler

Q21.3 For a **partition** of $\mathbb{R}^d$, some points may be in (no parts, more than one part), where "part" means one of the subsets of the partition?

A: (T,T) B: (T,F) C: (F,T) D: (F,F) **??**

Decision trees

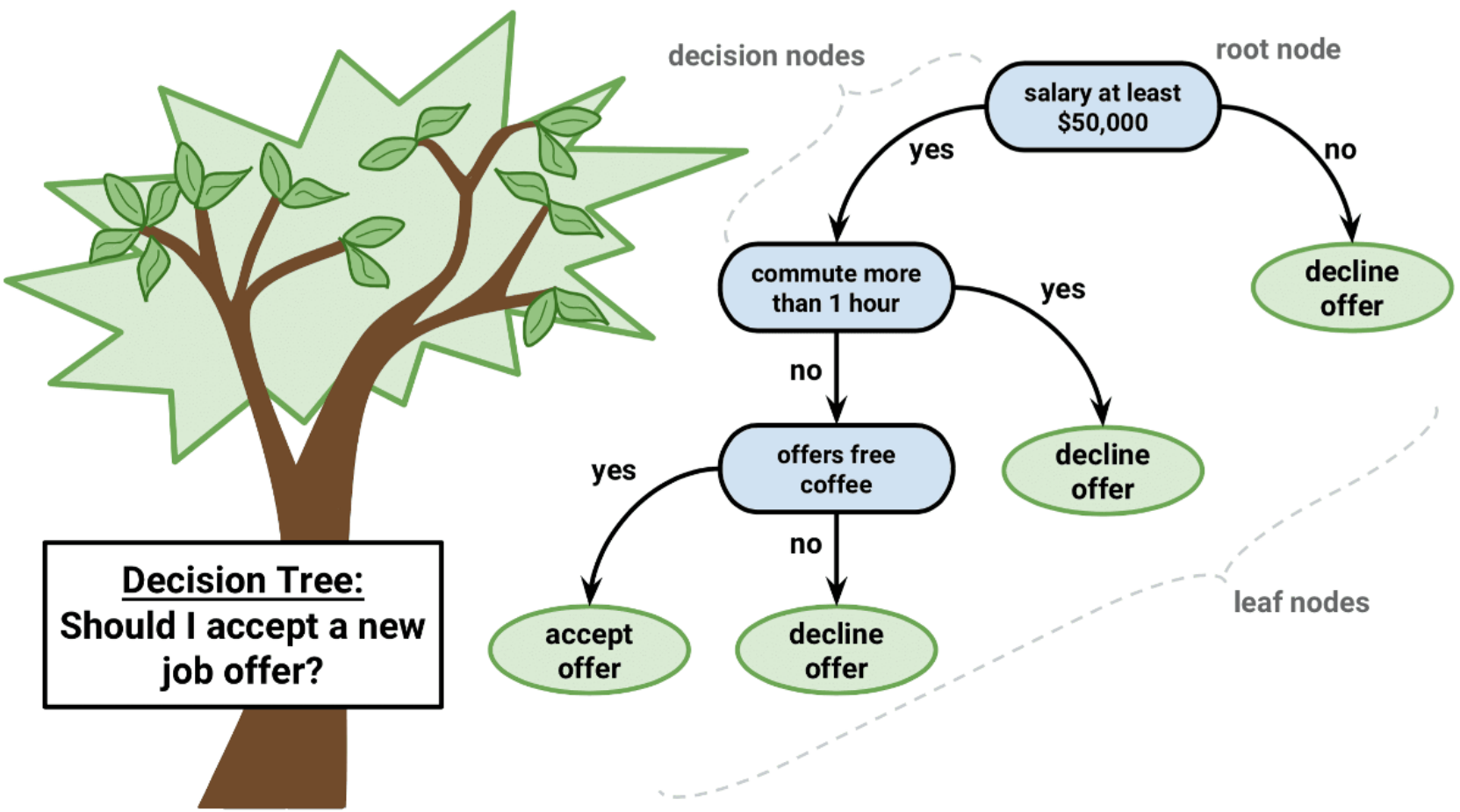
© 2026 Fessler

- A decision tree is a

  that is used for prediction.
- There are two main types:
  - A classification tree is a tree-structured partition where each rectangle/part of the partition is associated with a
  - A regression tree is a tree-structured partition where each rectangle/part of the partition is associated with a
- For both types of decision trees, prediction is accomplished by traversing the tree from the root to a leaf, and predicting the label/real number for that part.

Classification tree example

© 2026 Fessler

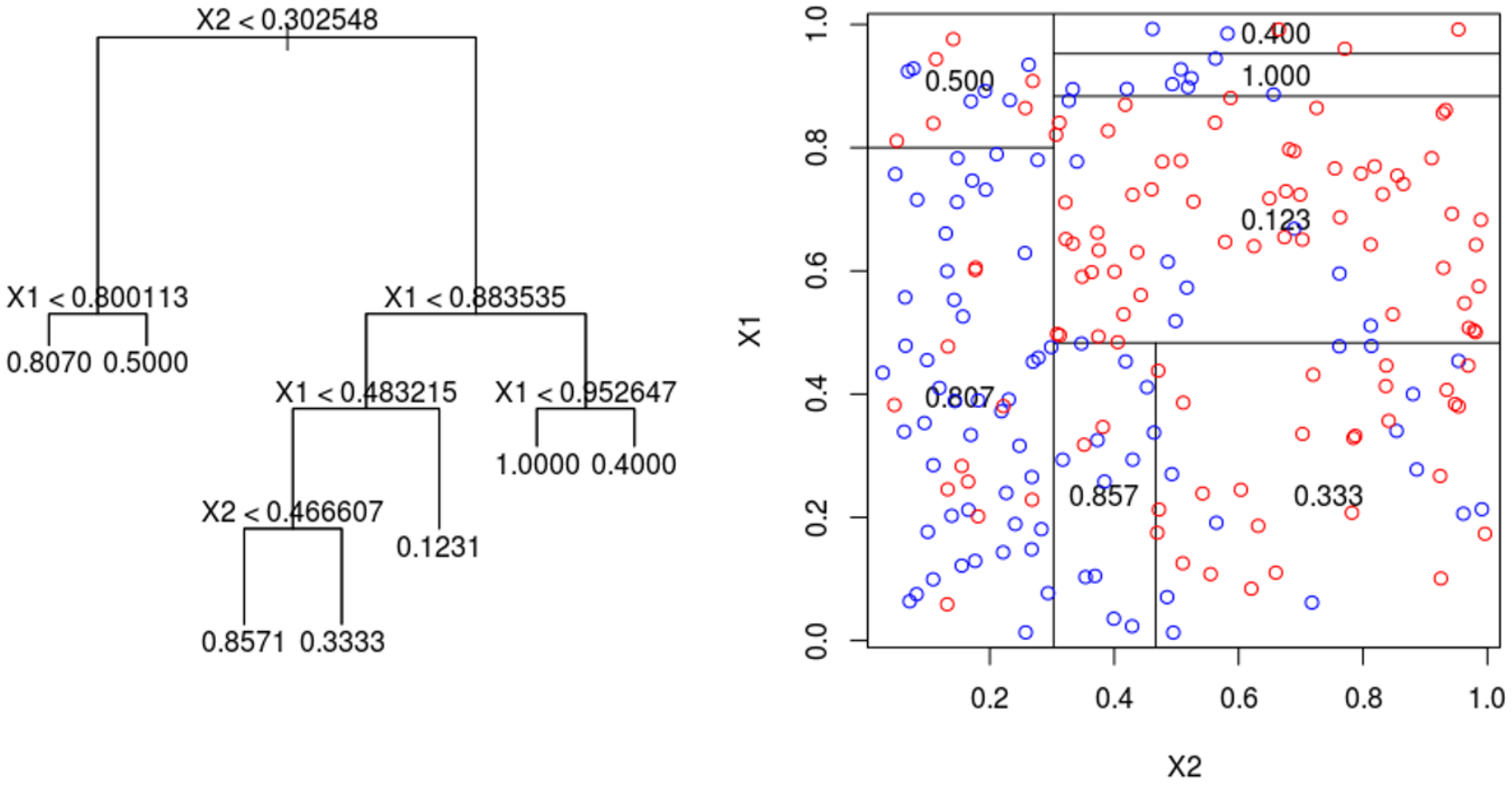


Here $\boldsymbol{x}_i$ for each potential job offer includes features like salary, commute time, whether they offer free coffee, and possibly more.

Figure courtesy datacamp.com.

Classification tree example

© 2026 Fessler

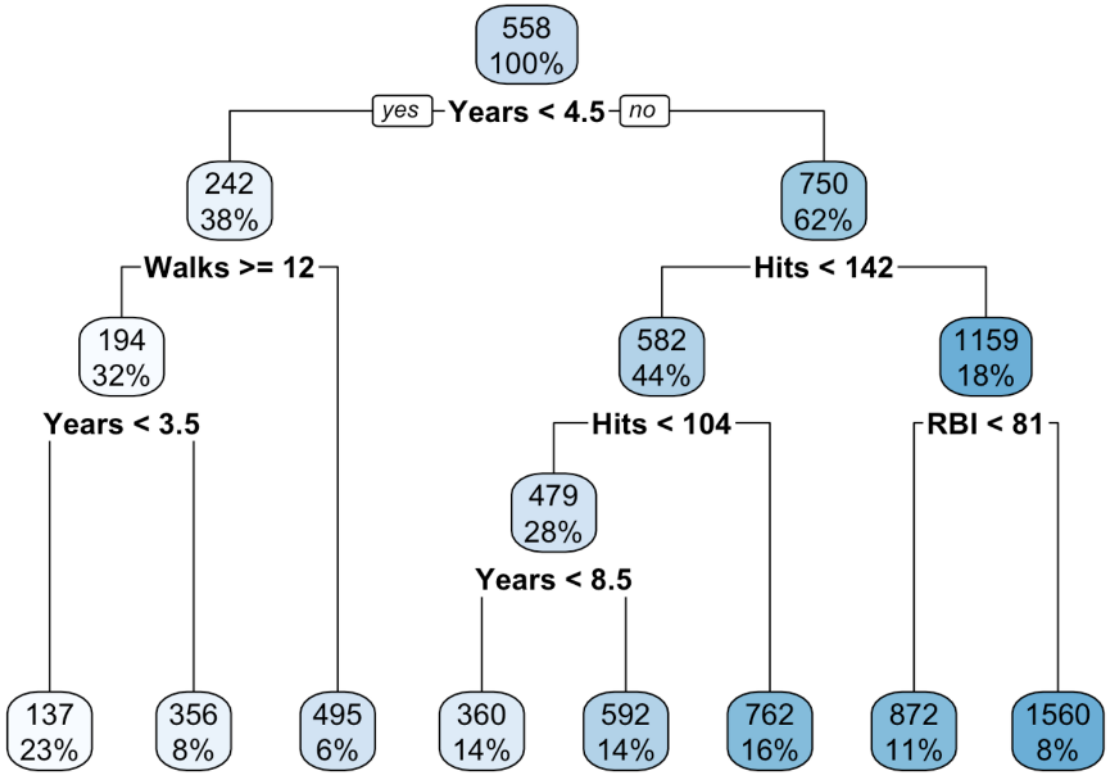


Predicted label $f(\boldsymbol{x})$ at leaf nodes = "majority vote" of class in rectangle (whichever class has the most labels).

Figure courtesy of datacamp.com.

Regression tree: Example

© 2026 Fessler



In this tree, left is "yes" and right is "no".

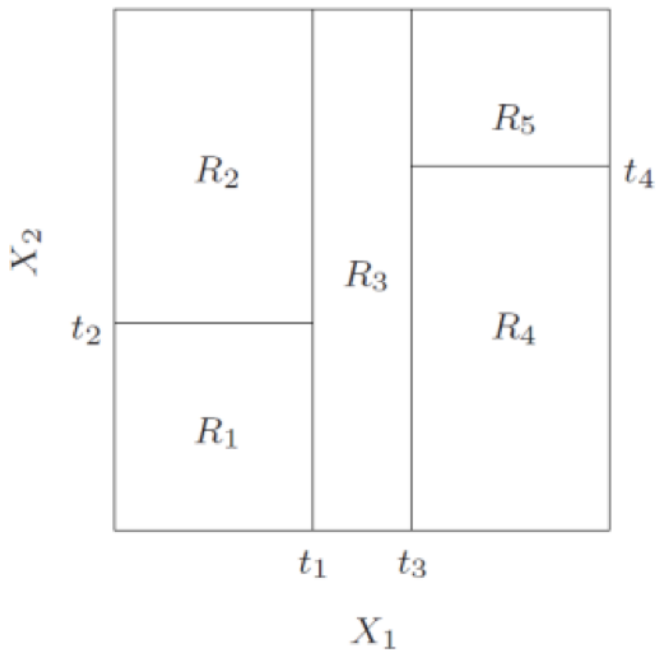
Figure courtesy of rpubs.com.

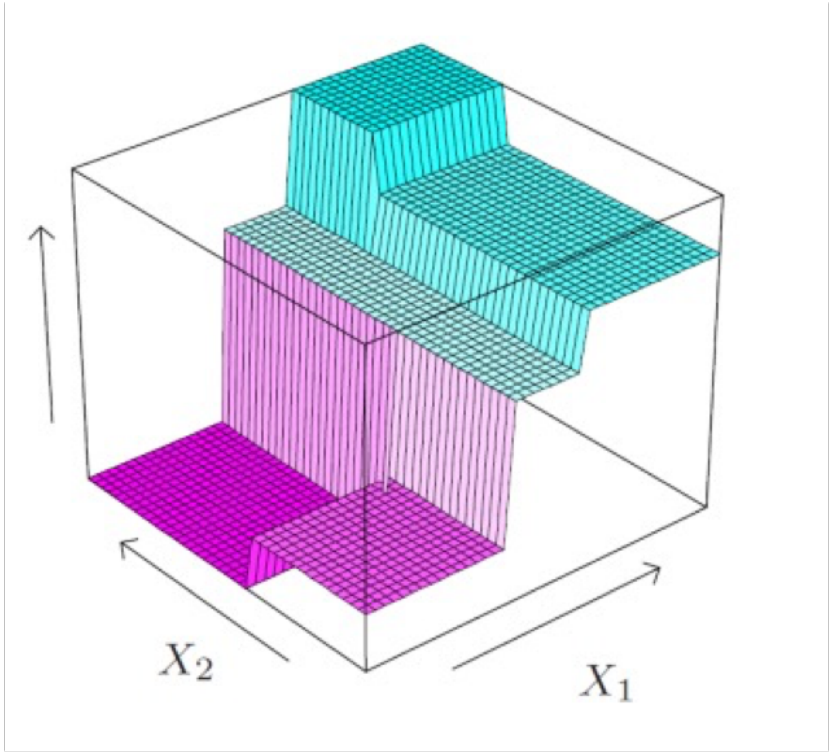
Here $\boldsymbol{x}_i$ for each player in Major League Baseball includes features like number of years in MLB, number of walks drawn last season, number of hits last season, RBI.

The response variable y_i is player *salary*.

Regression tree: Visualization

© 2026 Fessler





- Here the prediction (regression) is a **piece-wise constant function**.
- Suitably regular functions can be approximated arbitrarily well by piece-wise constant functions.

Figures courtesy [3].

Learning a decision tree

© 2026 Fessler

- Given training data $\{(\boldsymbol{x}_i, y_i)\}$ for classification or regression, how should we construct/design a decision tree?
- We would like to use ERM:

$$\arg\min_{f \in \mathcal{F}} \hat{R}_n(f), \qquad \hat{R}_n(f) =$$

  where our class of prediction functions $\mathcal{F}$ is the class of all decision trees (of some maximum depth).
- Sadly this optimization problem is **NP-hard**.
- Instead, learning is done heuristically and broken into two phases:

Learning a decision tree

© 2026 Fessler

- Growing a tree requires
  -
  -
- Pruning a tree is usually done after we greedily grow a large tree that fits the training data perfectly. Then we select areas to

Classification tree: Example partition

© 2026 Fessler

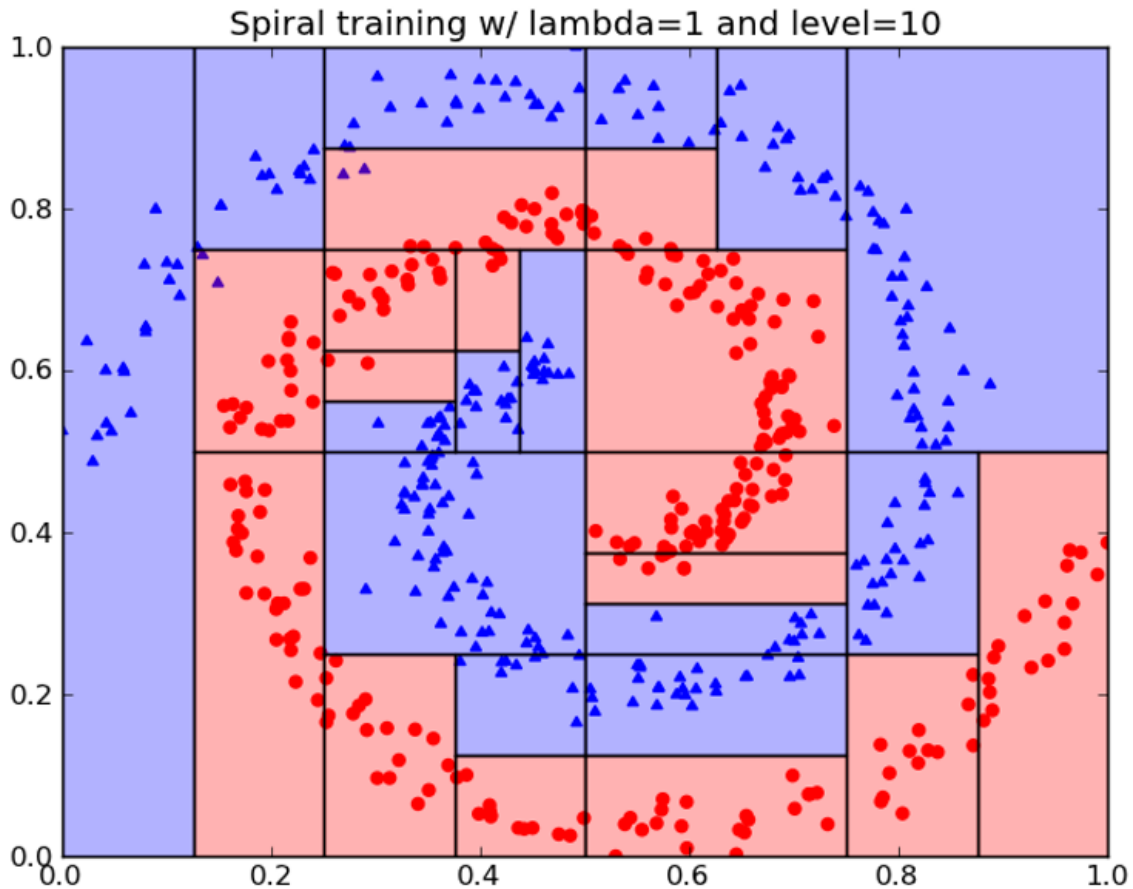


Figure courtesy uchicago.edu.

Decision tree: Summary

© 2026 Fessler

- Strengths:
  - Interpretable - think medicine!
  - Expressive
  - Fast (for both learning and prediction)
  - Can include categorical/mixed feature types

- Weaknesses:
  - Training optimally is NP-hard, and only suboptimal heuristics are used
  - Sensitive to training data

Q21.4 Once the space is partitioned, the label for each part of the partition (leaf node) in the classification tree is given by:

A: A function of the size of the part of the partition.

B: The average of the labels in the part of the partition.

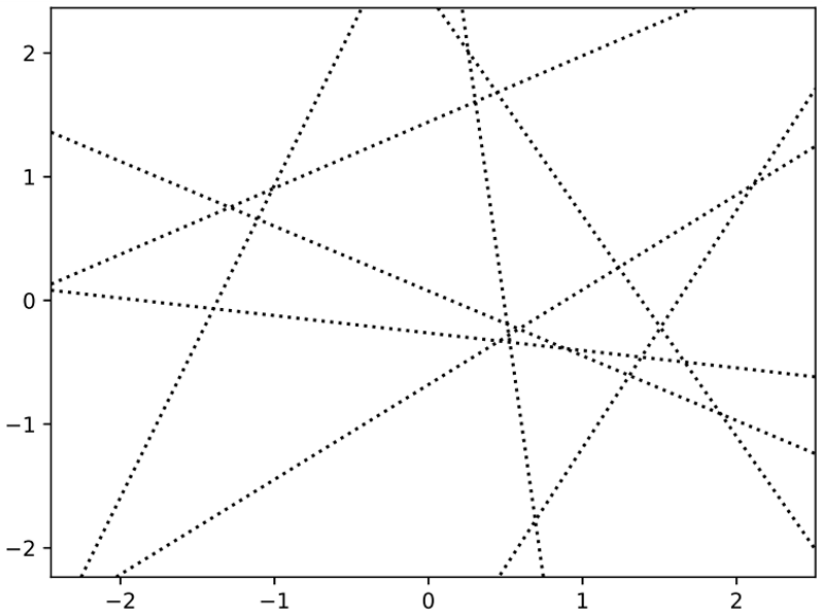
C: The class who has the highest number of labels among true labels for data points included in that part of the partition.

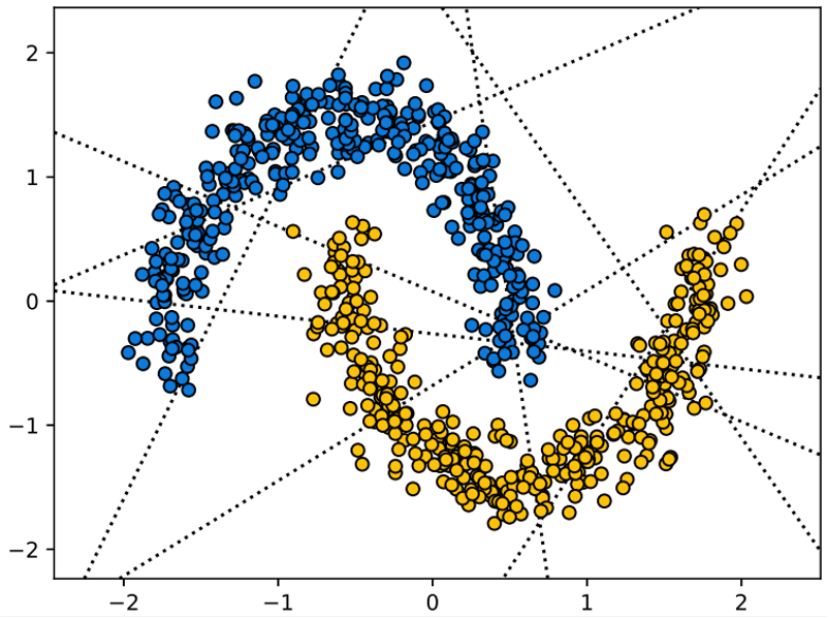
D: A label from $\{1, \ldots, K\}$ chosen uniformly at random.

E: none of these

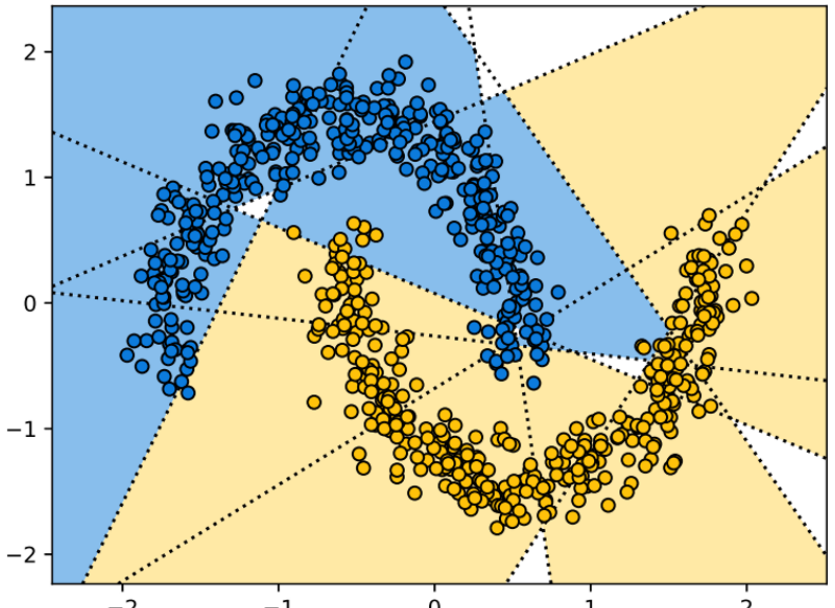
Hyperplane partition

© 2026 Fessler





- Here we use *arbitrary* hyperplanes instead of limiting to planes parallel to an axis.
- Again use majority vote method for classification.
- Ensemble methods (forthcoming) facilitate...



Figures courtesy Clay Scott.

References I

[1] K. P. Murphy.
Probabilistic machine learning: An introduction.
MIT Press, 2022.

[2] R. O. Duda, P. E. Hart, and D. G. Stork.
Pattern classification.
Wiley, New York, 2001.

[3] Trevor Hastie, Robert Tibshirani, and Jerome H Friedman.
The elements of statistical learning: data mining, inference, and prediction, volume 2.
Springer, 2009.

[4] Yoav Freund and Robert E Schapire.
A decision-theoretic generalization of on-line learning and an application to boosting.
Journal of computer and system sciences, 55(1):119–139, 1997.

[5] Robert E Schapire.
Explaining adaboost.
In *Empirical inference: festschrift in honor of vladimir N. Vapnik*, pages 37–52. Springer, 2013.

[6] Tianqi Chen and Carlos Guestrin.
Xgboost: A scalable tree boosting system.
In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.