Introduction to Supervised Learning

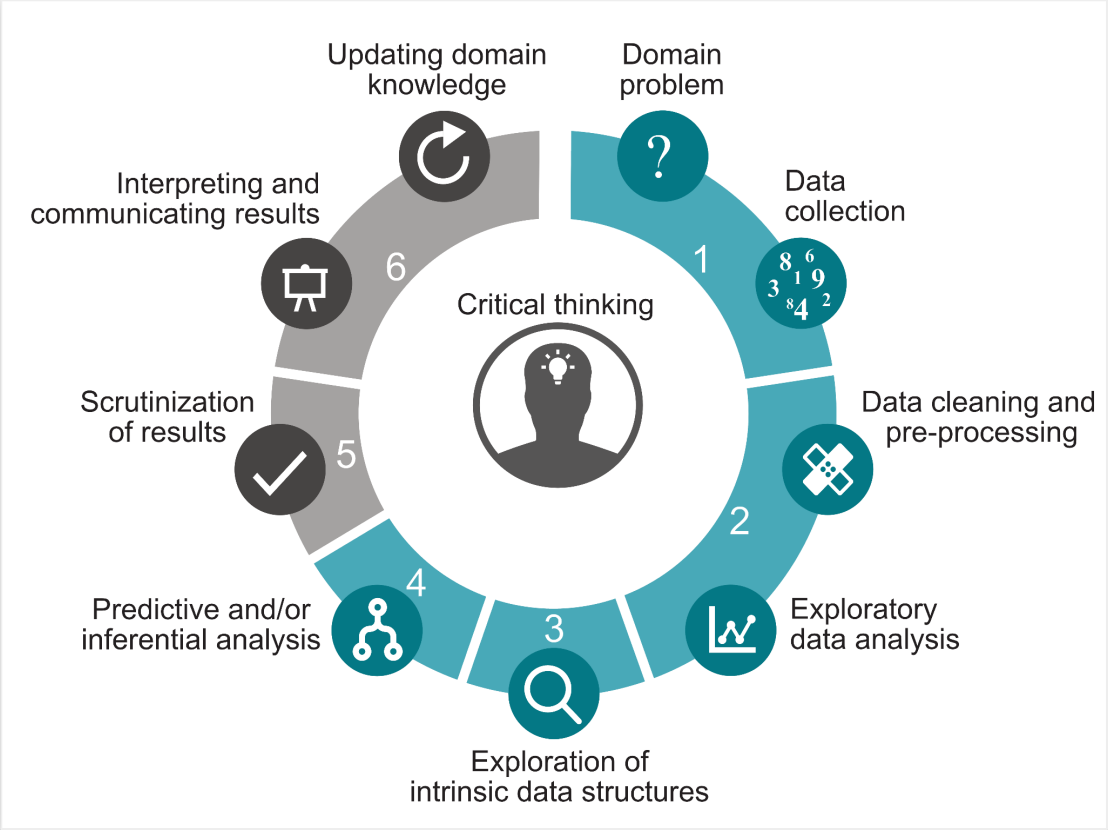
February 19, 2026

Today's plan: Introduction to Supervised Learning

1. Overview of Supervised Learning
2. Generalizability and the Bias-Variance Decomposition
3. Crash Course: Common ML Prediction Methods

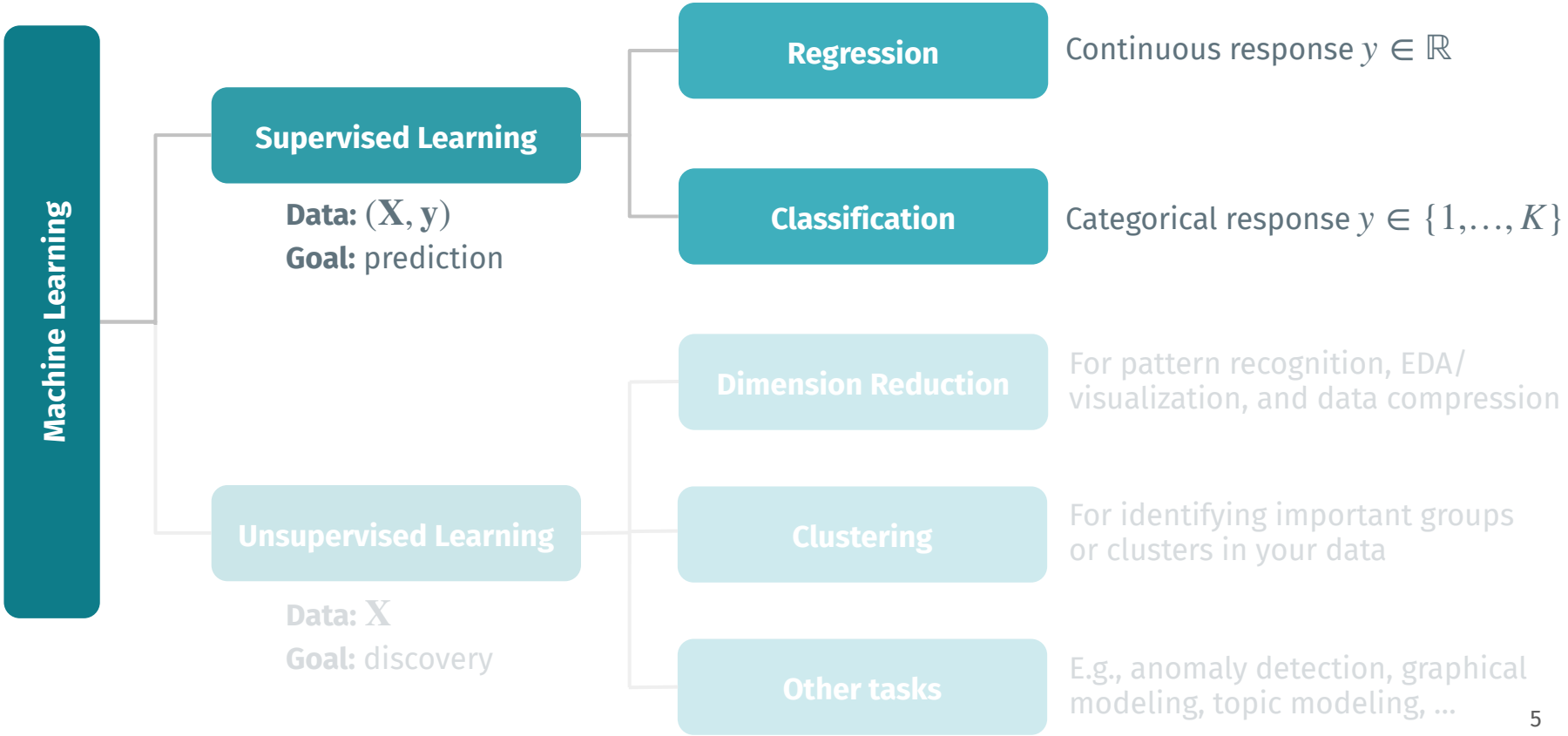
Overview of Supervised Learning

The Big Picture: Data Science Life Cycle



Credits: Veridical Data Science (VDS) Textbook

Overview of Machine Learning Terminology



Supervised Learning **Applications**

- **Precision medicine:** predicting disease diagnosis, disease risk, drug efficacy
- **Computer vision**: classifying images (e.g., cat vs dog vs flying monkey, tumor vs not tumor), image segmentation
- **Biometrics:** iris/facial detection (is it you or someone else)
- **Natural language processing:** sentiment analysis, machine translation, predicting the next word
- **Quantitative finance:** predicting stock market trends
- **Business analytics:** predicting sales, product demand, consumer behavior
- **Sports analytics:** March madness!!
- **Other:** predicting voter turnout, wildfires, spread of infectious diseases, …

Generalizability and Bias-Variance Decomposition

The #1 Goal of Supervised Learning

The goal of supervised learning is to learn insights from the current data that are generalizable to future, unseen data

The #1 Goal of Supervised Learning

The goal of supervised learning is to learn insights from the current data that are generalizable to future, unseen data

For example, if our primary task is prediction:

The #1 Goal of Supervised Learning

The goal of supervised learning is to learn insights from the current data that are generalizable to future, unseen data

For example, if our primary task is prediction:

- We do not care about making predictions on the current data
  - We already know the true responses/labels for the current data

The #1 Goal of Supervised Learning

The goal of supervised learning is to learn insights from the current data that are generalizable to future, unseen data

For example, if our primary task is prediction:

- We do not care about making predictions on the current data
  - We already know the true responses/labels for the current data
- In actuality, we want to build a prediction model that can make accurate predictions on data that we have yet to see but will see in the future

The #1 Goal of Supervised Learning

The goal of supervised learning is to learn insights from the current data that are generalizable to future, unseen data

For example, if our primary task is prediction:

- We do not care about making predictions on the current data
  - We already know the true responses/labels for the current data
- In actuality, we want to build a prediction model that can make accurate predictions on data that we have yet to see but will see in the future

This ability to "generalize" to new data is referred to as **generalizability**

Generalization Error

Generalization Error: measure of the model/algorithm's prediction accuracy on new unseen data

Generalization Error

Generalization Error: measure of the model/algorithm's prediction accuracy on new unseen data

- There are different ways (i.e., *loss/objective/cost* functions) to quantify generalization error:

Generalization Error

Generalization Error: measure of the model/algorithm's prediction accuracy on new unseen data

- There are different ways (i.e., *loss/objective/cost* functions) to quantify generalization error:
  - Regression:

Generalization Error

Generalization Error: measure of the model/algorithm's prediction accuracy on new unseen data

- There are different ways (i.e., *loss/objective/cost* functions) to quantify generalization error:
  - Regression:
    - **Mean-squared error** (i.e., $1/n \cdot SSE$)
    - Mean absolute error

Generalization Error

Generalization Error: measure of the model/algorithm's prediction accuracy on new unseen data

- There are different ways (i.e., *loss/objective/cost* functions) to quantify generalization error:
  - Regression:
    - **Mean-squared error** (i.e., $1/n \cdot SSE$)
    - Mean absolute error
  - Classification:

Generalization Error

Generalization Error: measure of the model/algorithm's prediction accuracy on new unseen data

- There are different ways (i.e., *loss/objective/cost* functions) to quantify generalization error:
  - Regression:
    - **Mean-squared error** (i.e., $1/n \cdot SSE$)
    - Mean absolute error
  - Classification:
    - **Cross-entropy (log) loss**
    - KL divergence

Generalization Error

Generalization Error: measure of the model/algorithm's prediction accuracy on new unseen data

- There are different ways (i.e., *loss/objective/cost* functions) to quantify generalization error:
  - Regression:
    - **Mean-squared error** (i.e., $1/n \cdot SSE$)
    - Mean absolute error
  - Classification:
    - **Cross-entropy (log) loss**
    - KL divergence

If the primary task is prediction, we are often most interested in finding the model with the **smallest generalization error**

Bias-Variance Decomposition

The generalization error (as measured by MSE) can be decomposed to reveal a **bias-variance decomposition**:

See derivation here.

Bias-Variance Decomposition

The generalization error (as measured by MSE) can be decomposed to reveal a **bias-variance decomposition**:

- Let y denote the true response, and suppose that

$$y = f(x) + \varepsilon, \quad \text{where } \mathbb{E}[\varepsilon] = 0 \text{ and } \text{Var}(\varepsilon) = \sigma^2.$$

See derivation here.

Bias-Variance Decomposition

The generalization error (as measured by MSE) can be decomposed to reveal a **bias-variance decomposition**:

- Let y denote the true response, and suppose that

$$y = f(x) + \varepsilon, \quad \text{where } \mathbb{E}[\varepsilon] = 0 \text{ and } \text{Var}(\varepsilon) = \sigma^2.$$

- Let $\hat{f}$ denote the model so that $\hat{f}(x)$ denotes the model predictions.

See derivation here.

Bias-Variance Decomposition

The generalization error (as measured by MSE) can be decomposed to reveal a **bias-variance decomposition**:

- Let y denote the true response, and suppose that

$$y = f(x) + \varepsilon, \quad \text{where } \mathbb{E}[\varepsilon] = 0 \text{ and } \text{Var}(\varepsilon) = \sigma^2.$$

- Let $\hat{f}$ denote the model so that $\hat{f}(x)$ denotes the model predictions.
- Then it can be shown that

See derivation here.

Bias-Variance Decomposition

The generalization error (as measured by MSE) can be decomposed to reveal a **bias-variance decomposition**:

- Let y denote the true response, and suppose that

$$y = f(x) + \varepsilon, \quad \text{where } \mathbb{E}[\varepsilon] = 0 \text{ and } \text{Var}(\varepsilon) = \sigma^2.$$

- Let $\hat{f}$ denote the model so that $\hat{f}(x)$ denotes the model predictions.
- Then it can be shown that

$$\mathbb{E}[(y - \hat{f}(x))^2] = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \sigma^2$$

See derivation here.

Bias-Variance Decomposition

The generalization error (as measured by MSE) can be decomposed to reveal a **bias-variance decomposition**:

- Let y denote the true response, and suppose that

$$y = f(x) + \varepsilon, \quad \text{where } \mathbb{E}[\varepsilon] = 0 \text{ and } \text{Var}(\varepsilon) = \sigma^2.$$

- Let $\hat{f}$ denote the model so that $\hat{f}(x)$ denotes the model predictions.
- Then it can be shown that

$$\underbrace{\mathbb{E}[(y - \hat{f}(x))^2]}_{\text{Population MSE}} = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \sigma^2$$

See derivation here.

Bias-Variance Decomposition

The generalization error (as measured by MSE) can be decomposed to reveal a **bias-variance decomposition**:

- Let y denote the true response, and suppose that

$$y = f(x) + \varepsilon, \quad \text{where } \mathbb{E}[\varepsilon] = 0 \text{ and } \text{Var}(\varepsilon) = \sigma^2.$$

- Let $\hat{f}$ denote the model so that $\hat{f}(x)$ denotes the model predictions.
- Then it can be shown that

$$\underbrace{\mathbb{E}[(y - \hat{f}(x))^2]}_{\text{Population MSE}} = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \underbrace{\sigma^2}_{\text{Irreproducible Error}}$$

See derivation here.

Bias-Variance Decomposition

The generalization error (as measured by MSE) can be decomposed to reveal a **bias-variance decomposition**:

- Let y denote the true response, and suppose that

$$y = f(x) + \varepsilon, \quad \text{where } \mathbb{E}[\varepsilon] = 0 \text{ and } \text{Var}(\varepsilon) = \sigma^2.$$

- Let $\hat{f}$ denote the model so that $\hat{f}(x)$ denotes the model predictions.
- Then it can be shown that

$$\underbrace{\mathbb{E}[(y - \hat{f}(x))^2]}_{\text{Population MSE}} = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \underbrace{\sigma^2}_{\text{Irreproducible Error}}$$

where $\text{Bias}(\hat{f}(x)) = \mathbb{E}[\hat{f}(x)] - f(x)$

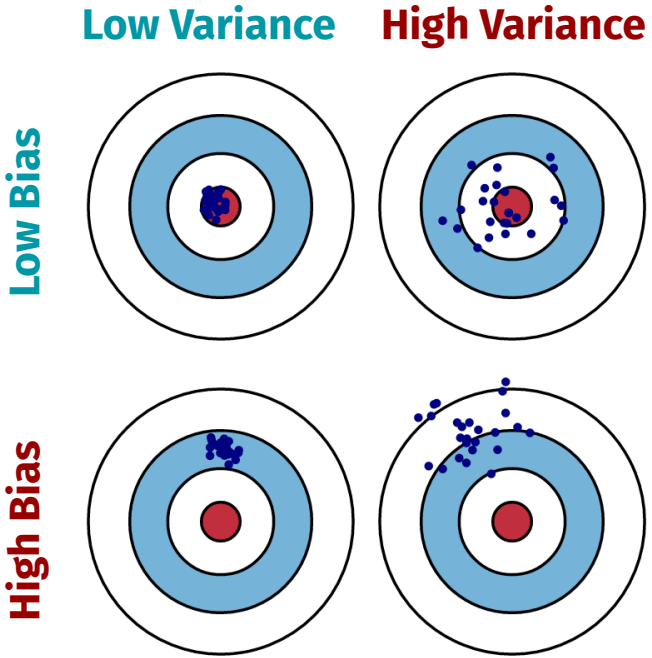
$$\text{Var}(\hat{f}(x)) = \mathbb{E}[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2]$$

See derivation here.

Bias-Variance Decomposition

$$\underbrace{\mathbb{E}[(y - \hat{f}(x))^2]}_{\text{Population MSE}} = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \underbrace{\sigma^2}_{\text{Irreproducible Error}}$$

- **Bias:** on average, how wrong is your prediction from the truth
- **Variance:** if you obtain a new but similar dataset, how much does this change your predictions



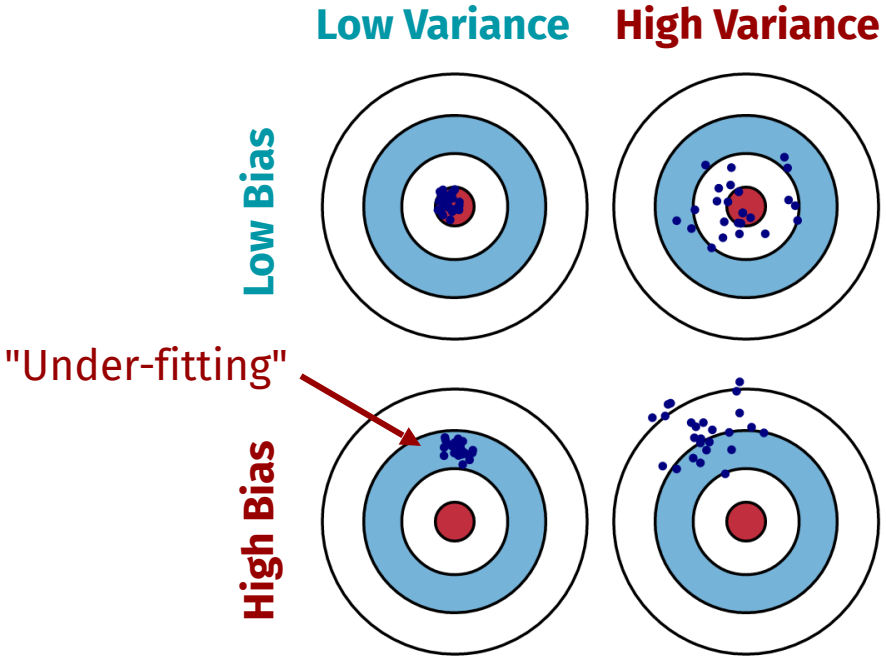
Bias-Variance Decomposition

$$\mathbb{E}[(y - \hat{f}(x))^2] = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \sigma^2$$

Population MSE

Irreproducible Error

- **Bias:** on average, how wrong is your prediction from the truth
- **Variance:** if you obtain a new but similar dataset, how much does this change your predictions



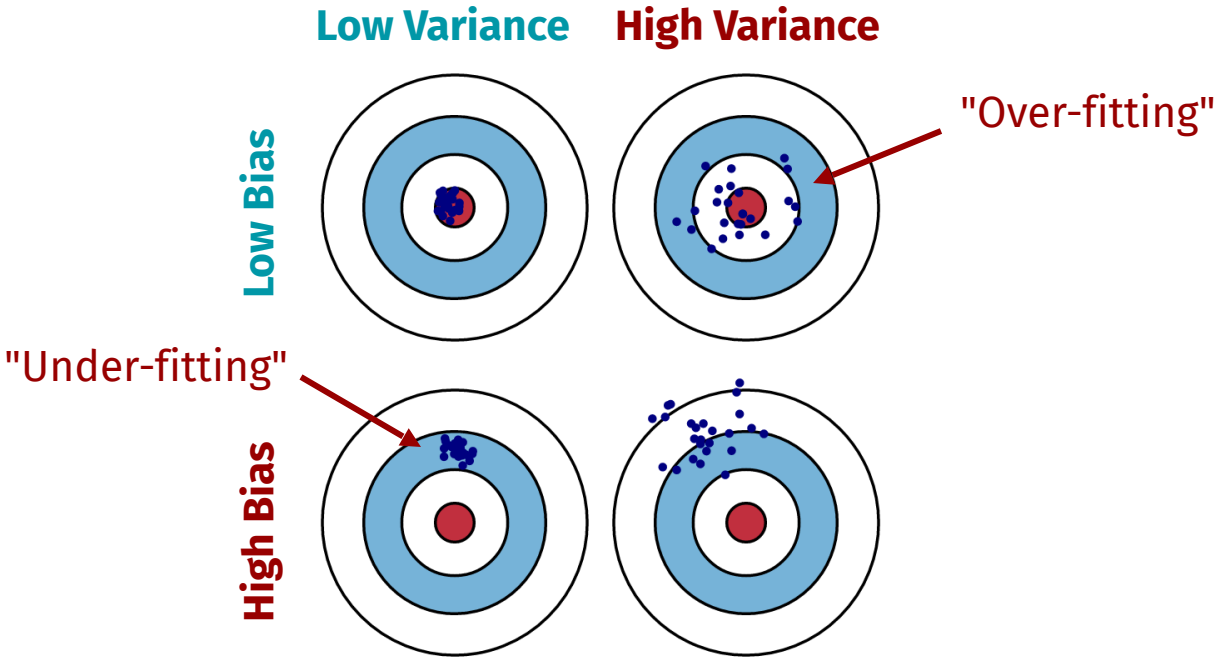
Bias-Variance Decomposition

$$\mathbb{E}[(y - \hat{f}(x))^2] = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \sigma^2$$

Population MSE

Irreproducible Error

- **Bias:** on average, how wrong is your prediction from the truth
- **Variance:** if you obtain a new but similar dataset, how much does this change your predictions



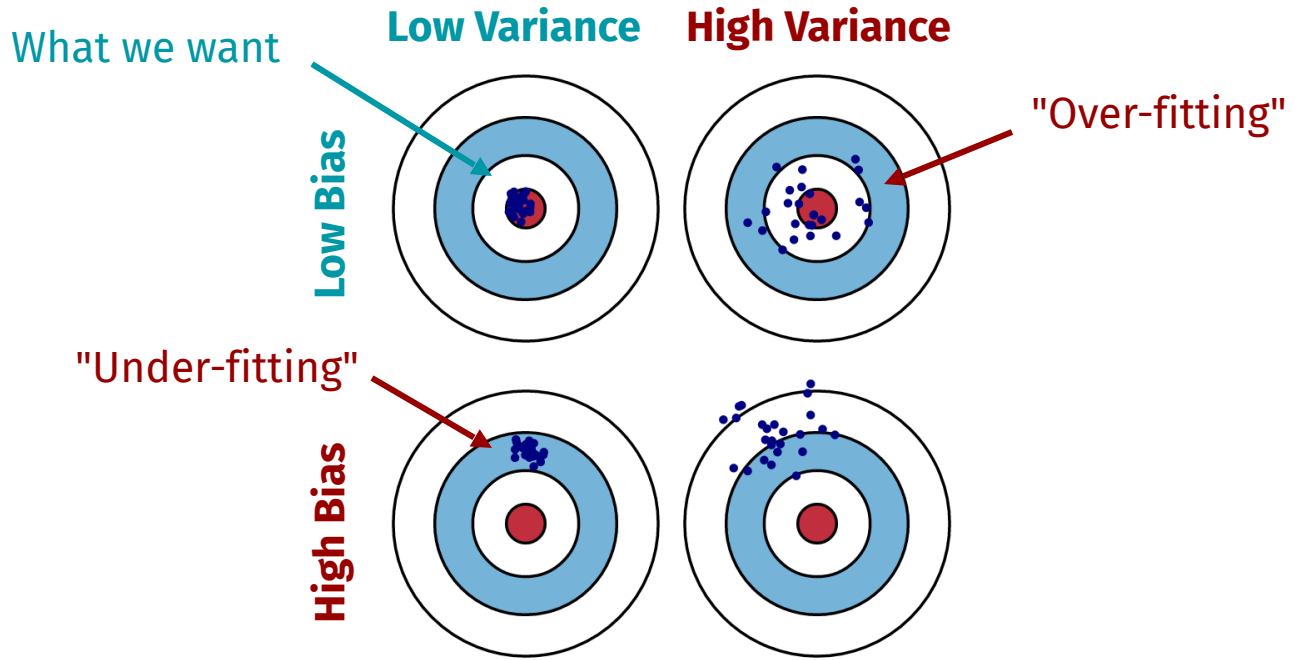
Bias-Variance Decomposition

$$\mathbb{E}[(y - \hat{f}(x))^2] = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \sigma^2$$

Population MSE

Irreproducible Error

- **Bias:** on average, how wrong is your prediction from the truth
- **Variance:** if you obtain a new but similar dataset, how much does this change your predictions



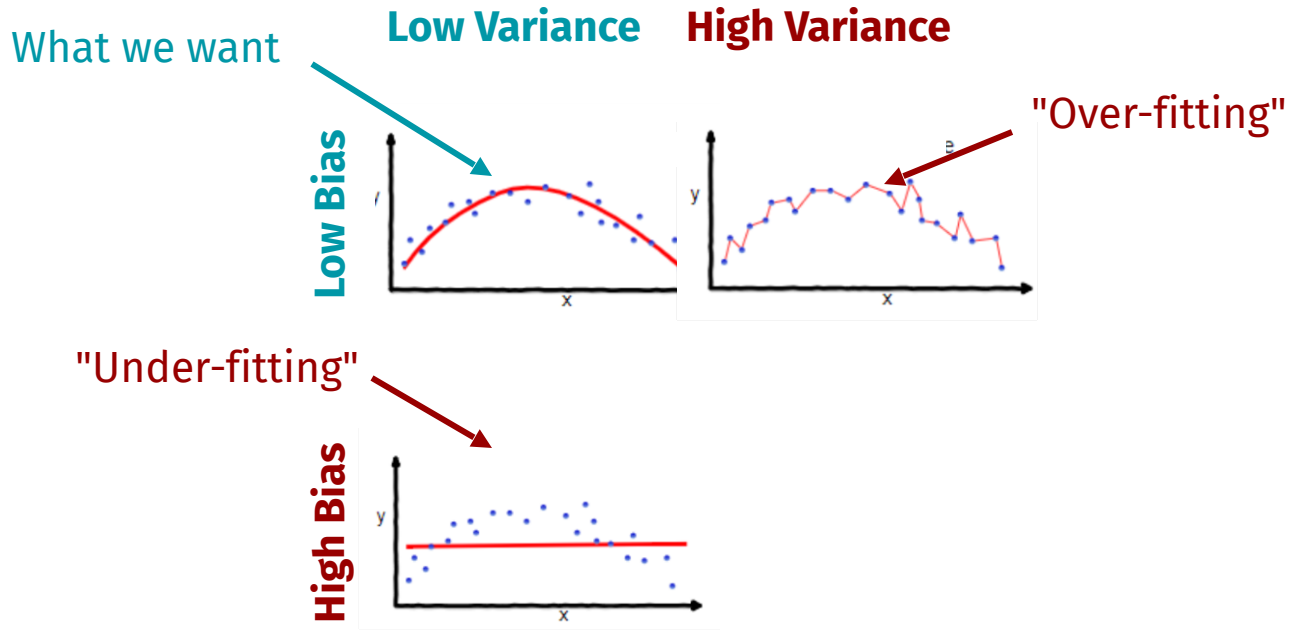
Bias-Variance Decomposition

$$\mathbb{E}[(y - \hat{f}(x))^2] = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \sigma^2$$

Population MSE

Irreproducible Error

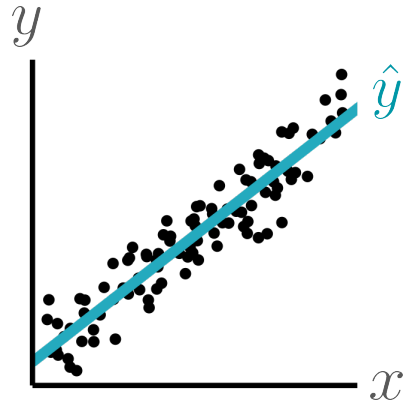
- **Bias:** on average, how wrong is your prediction from the truth
- **Variance:** if you obtain a new but similar dataset, how much does this change your predictions



Linear Regression (or Ordinary Least Squares)

Given a covariate data matrix $\mathbf{X}$ and response vector $\mathbf{y} \in \mathbb{R}^n$

linear regression finds the *line of best fit*.

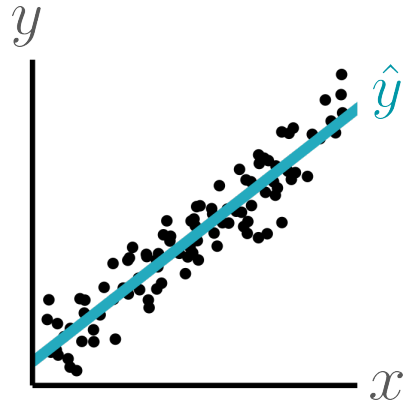


Linear Regression (or Ordinary Least Squares)

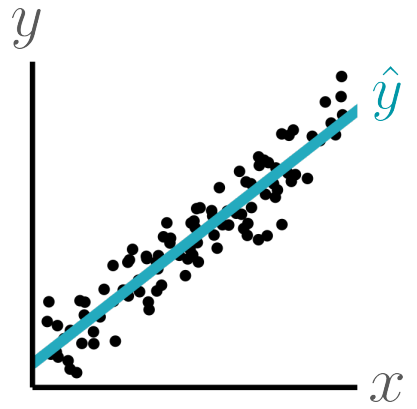
Given a covariate data matrix $\mathbf{X}$ and response vector $\mathbf{y} \in \mathbb{R}^n$

linear regression finds the *line of best fit*.

+ How do we fit linear regression? Minimize MSE.



Linear Regression (or Ordinary Least Squares)



Given a covariate data matrix $\mathbf{X}$ and response vector $\mathbf{y} \in \mathbb{R}^n$

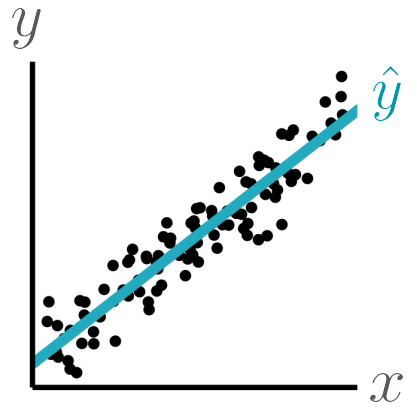
linear regression finds the *line of best fit*.

+ How do we fit linear regression? Minimize MSE.

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \frac{1}{n} \|\mathbf{y} - \mathbf{X}\beta\|_2^2 = \frac{1}{n} \sum_{i=1}^{n} \{y_i - (\beta_1 X_{i1} + \ldots + \beta_p X_{ip})\}^2$$

* ignoring the intercept to simplify notation

Linear Regression (or Ordinary Least Squares)



Given a covariate data matrix $\mathbf{X}$ and response vector $\mathbf{y} \in \mathbb{R}^n$

linear regression finds the *line of best fit*.

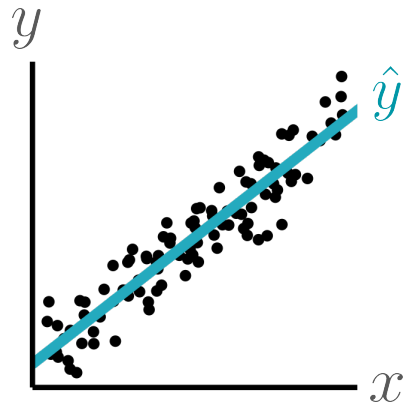
+ How do we fit linear regression? Minimize MSE.

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \frac{1}{n} \|\mathbf{y} - \mathbf{X}\beta\|_2^2 = \frac{1}{n} \sum_{i=1}^{n} \{y_i - (\beta_1 X_{i1} + \ldots + \beta_p X_{ip})\}^2$$

+ How do we make predictions from the fitted linear regression?

* ignoring the intercept to simplify notation

Linear Regression (or Ordinary Least Squares)



Given a covariate data matrix $\mathbf{X}$ and response vector $\mathbf{y} \in \mathbb{R}^n$

linear regression finds the *line of best fit*.

+ How do we fit linear regression? Minimize MSE.

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \frac{1}{n} \|\mathbf{y} - \mathbf{X}\beta\|_2^2 = \frac{1}{n} \sum_{i=1}^{n} \{y_i - (\beta_1 X_{i1} + \ldots + \beta_p X_{ip})\}^2$$

+ How do we make predictions from the fitted linear regression?

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\beta} \quad \text{or} \quad \hat{y}_i = \hat{\beta}_1 X_{i1} + \ldots + \hat{\beta}_p X_{ip} \ \text{(for } i = 1, \ldots, n)$$

* ignoring the intercept to simplify notation

Linear regression in light of the Bias-Variance decomposition

Gauss-Markov Theorem: Linear regression (OLS) = best unbiased linear estimator*
(i.e., OLS is unbiased and has the smallest variance of all linear estimators)

$$\underbrace{\mathbb{E}[(y - \hat{f}(x))^2]}_{\text{Population MSE}} = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \underbrace{\sigma^2}_{\text{Irreproducible Error}}$$

* Need centered, homoskedastic, independent noise assumption.
Formal statement and proof of Gauss-Markov theorem here.

Linear regression in light of the Bias-Variance decomposition

Gauss-Markov Theorem: Linear regression (OLS) = best unbiased linear estimator*

(i.e., OLS is unbiased and has the smallest variance of all linear estimators)

$$\mathbb{E}[(y - \hat{f}(x))^2] = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \sigma^2$$

Population MSE

Irreproducible Error

* Need centered, homoskedastic, independent noise assumption.
Formal statement and proof of Gauss-Markov theorem here.

Linear regression in light of the Bias-Variance decomposition

Gauss-Markov Theorem: Linear regression (OLS) = best unbiased linear estimator*

(i.e., OLS is unbiased and has the smallest variance of all linear estimators)

$$\underbrace{\mathbb{E}[(y - \hat{f}(x))^2]}_{\text{Population MSE}} = \cancel{\text{Bias}^2(\hat{f}(x))} + \text{Var}(\hat{f}(x)) + \underbrace{\sigma^2}_{\text{Irreproducible Error}}$$

Minimized over all unbiased linear estimators

* Need centered, homoskedastic, independent noise assumption.
Formal statement and proof of Gauss-Markov theorem here.

Linear regression in light of the Bias-Variance decomposition

Gauss-Markov Theorem: Linear regression (OLS) = best unbiased linear estimator*
(i.e., OLS is unbiased and has the smallest variance of all linear estimators)

$$\mathbb{E}[(y - \hat{f}(x))^2] = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \sigma^2$$

Population MSE

Irreproducible Error

Minimized over all unbiased linear estimators

By Gauss-Markov, linear regression (OLS) gives the smallest generalization error if we restrict $\hat{f}$ to be an unbiased linear estimator

* Need centered, homoskedastic, independent noise assumption.
Formal statement and proof of Gauss-Markov theorem here.

Linear regression in light of the Bias-Variance decomposition

Gauss-Markov Theorem: Linear regression (OLS) = best unbiased linear estimator*
(i.e., OLS is unbiased and has the smallest variance of all linear estimators)

$$\mathbb{E}[(y - \hat{f}(x))^2] = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \sigma^2$$

Population MSE

Irreproducible Error

Minimized over all unbiased linear estimators

By Gauss-Markov, linear regression (OLS) gives the smallest generalization error if we restrict $\hat{f}$ to be an unbiased linear estimator

What if we allow a little bit a bias but are able to substantially reduce variance?

* Need centered, homoskedastic, independent noise assumption.
Formal statement and proof of Gauss-Markov theorem here.

Crash Course: Common ML Prediction Methods

Regularized (ridge) linear regression

Motivating idea: sacrifice unbiasedness in order to reduce model variance

Regularized (ridge) linear regression

Motivating idea: sacrifice unbiasedness in order to reduce model variance

+ Where does the high variance come from? Partially due to big coefficients

Regularized (ridge) linear regression

Motivating idea: sacrifice unbiasedness in order to reduce model variance

+ Where does the high variance come from? Partially due to big coefficients

Regularized linear regression: do linear regression but *regularize/penalize* coefficients so that they don't become too big

Regularized (ridge) linear regression

Motivating idea: sacrifice unbiasedness in order to reduce model variance

+ Where does the high variance come from? Partially due to big coefficients

Regularized linear regression: do linear regression but *regularize/penalize* coefficients so that they don't become too big

Regularized (ridge) linear regression

Motivating idea: sacrifice unbiasedness in order to reduce model variance

- Where does the high variance come from? Partially due to big coefficients

Regularized linear regression: do linear regression but *regularize/penalize* coefficients so that they don't become too big

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\substack{\text{penalty} \\ \text{term}}}$$

Regularized (ridge) linear regression

Motivating idea: sacrifice unbiasedness in order to reduce model variance

- Where does the high variance come from? Partially due to big coefficients

Regularized linear regression: do linear regression but *regularize/penalize* coefficients so that they don't become too big

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\substack{\text{penalty} \\ \text{term}}}$$

("ridge penalty")

Regularized (ridge) linear regression

Motivating idea: sacrifice unbiasedness in order to reduce model variance

- Where does the high variance come from? Partially due to big coefficients

Regularized linear regression: do linear regression but *regularize/penalize* coefficients so that they don't become too big

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\substack{\text{penalty} \\ \text{term}}}$$

("ridge penalty")

- $\hat{\beta}$ are the estimated coefficients (learned by the model)
- λ > 0 is a tuning parameter (chosen a priori by us)

Interpreting λ

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\substack{\text{penalty} \\ \text{term}}}$$

("ridge penalty")

- Intuitively, λ controls the amount or strength of regularization

Interpreting λ

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\substack{\text{penalty} \\ \text{term}}}$$

("ridge penalty")

- Intuitively, λ controls the amount or strength of regularization
  - When $\lambda = 0$, then $\hat{\beta} = \hat{\beta}^{OLS}$ (no bias, ↑ variance)

Interpreting λ

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\substack{\text{penalty} \\ \text{term}}}$$

("ridge penalty")

- Intuitively, λ controls the amount or strength of regularization
  - When $\lambda = 0$, then $\hat{\beta} = \hat{\beta}^{OLS}$ (no bias, ↑ variance)
  - When $\lambda = \infty$, then $\hat{\beta} = 0$ (↑ bias, no variance)

Interpreting λ

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\substack{\text{penalty} \\ \text{term}}}$$

("ridge penalty")

- Intuitively, λ controls the amount or strength of regularization
  - When $\lambda = 0$, then $\hat{\beta} = \hat{\beta}^{OLS}$ (no bias, ↑ variance)
  - When $\lambda = \infty$, then $\hat{\beta} = 0$ (↑ bias, no variance)
  - λ somewhere in between → some shrinkage

Interpreting λ

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\substack{\text{penalty} \\ \text{term}}}$$

("ridge penalty")

- Intuitively, λ controls the amount or strength of regularization
  - When $\lambda = 0$, then $\hat{\beta} = \hat{\beta}^{OLS}$ (no bias, ↑ variance)
  - When $\lambda = \infty$, then $\hat{\beta} = 0$ (↑ bias, no variance)
  - λ somewhere in between → some shrinkage
- Want to strike a balance between adding bias and reducing variance

Interpreting λ

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\substack{\text{penalty} \\ \text{term}}}$$

("ridge penalty")

- Intuitively, λ controls the amount or strength of regularization
  - When $\lambda = 0$, then $\hat{\beta} = \hat{\beta}^{OLS}$ (no bias, ↑ variance)
  - When $\lambda = \infty$, then $\hat{\beta} = 0$ (↑ bias, no variance)
  - λ somewhere in between → some shrinkage
- Want to strike a balance between adding bias and reducing variance
- We can select an appropriate λ using cross-validation (discussed next time)

Ridge regression MSE existence theorem

Assume:

$$\mathbf{y} = \mathbf{X}\beta + \varepsilon\,, \quad \text{where } \mathbb{E}[\varepsilon] = 0 \text{ and } \mathrm{Var}(\varepsilon) = \sigma^2\mathbf{I}$$

Then there **always** exists a λ such that Ridge(λ) MSE is less than the OLS MSE.

Types of regularized linear regression

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda P(\boldsymbol{\beta})}_{\substack{\text{penalty} \\ \text{term}}}$$

Ridge regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_2^2 = \sum_{j=1}^{p} \beta_j^2$$

LASSO regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|$$

Elastic net regression

$$P(\boldsymbol{\beta}) = \alpha\|\boldsymbol{\beta}\|_1 + (1-\alpha)\|\boldsymbol{\beta}\|_2^2$$

Types of regularized linear regression

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda P(\boldsymbol{\beta})}_{\substack{\text{penalty} \\ \text{term}}}$$

Ridge regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_2^2 = \sum_{j=1}^{p} \beta_j^2$$

LASSO regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|$$

Elastic net regression

$$P(\boldsymbol{\beta}) = \alpha\|\boldsymbol{\beta}\|_1 + (1-\alpha)\|\boldsymbol{\beta}\|_2^2$$

- \+ Generally good for prediction (MSE existence theorem)
- \+ Coefficients of correlated features tend to be shrunk together (i.e., are similar)

Types of regularized linear regression

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda P(\boldsymbol{\beta})}_{\substack{\text{penalty} \\ \text{term}}}$$

Ridge regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_2^2 = \sum_{j=1}^{p} \beta_j^2$$

- Generally good for prediction (MSE existence theorem)
- Coefficients of correlated features tend to be shrunk together (i.e., are similar)

LASSO regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|$$

- Performs automatic feature selection (i.e., "zeros" out features; induces *sparsity*)
- Often selects one feature from correlated group (all others will be zeroed out)

Elastic net regression

$$P(\boldsymbol{\beta}) = \alpha\|\boldsymbol{\beta}\|_1 + (1-\alpha)\|\boldsymbol{\beta}\|_2^2$$

Types of regularized linear regression

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda P(\boldsymbol{\beta})}_{\substack{\text{penalty} \\ \text{term}}}$$

Ridge regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_2^2 = \sum_{j=1}^{p} \beta_j^2$$

- Generally good for prediction (MSE existence theorem)
- Coefficients of correlated features tend to be shrunk together (i.e., are similar)

LASSO regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|$$

- Performs automatic feature selection (i.e., "zeros" out features; induces *sparsity*)
- Often selects one feature from correlated group (all others will be zeroed out)

Elastic net regression

$$P(\boldsymbol{\beta}) = \alpha\|\boldsymbol{\beta}\|_1 + (1-\alpha)\|\boldsymbol{\beta}\|_2^2$$

- Compromise between ridge and LASSO

Types of regularized linear regression

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda P(\boldsymbol{\beta})}_{\substack{\text{penalty} \\ \text{term}}}$$

Ridge regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_2^2 = \sum_{j=1}^{p} \beta_j^2$$

+ Generally good for prediction (MSE existence theorem)
+ Coefficients of correlated features tend to be shrunk together (i.e., are similar)
- Dense model (no feature selection)

LASSO regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|$$

+ Performs automatic feature selection (i.e., "zeros" out features; induces *sparsity*)
+ Often selects one feature from correlated group (all others will be zeroed out)

Elastic net regression

$$P(\boldsymbol{\beta}) = \alpha\|\boldsymbol{\beta}\|_1 + (1-\alpha)\|\boldsymbol{\beta}\|_2^2$$

+ Compromise between ridge and LASSO

Types of regularized linear regression

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda P(\boldsymbol{\beta})}_{\substack{\text{penalty} \\ \text{term}}}$$

Ridge regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_2^2 = \sum_{j=1}^{p} \beta_j^2$$

+ Generally good for prediction (MSE existence theorem)
+ Coefficients of correlated features tend to be shrunk together (i.e., are similar)
- Dense model (no feature selection)

LASSO regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|$$

+ Performs automatic feature selection (i.e., "zeros" out features; induces *sparsity*)
+ Often selects one feature from correlated group (all others will be zeroed out)
- Gives lower accuracy if truth is not sparse

Elastic net regression

$$P(\boldsymbol{\beta}) = \alpha\|\boldsymbol{\beta}\|_1 + (1-\alpha)\|\boldsymbol{\beta}\|_2^2$$

+ Compromise between ridge and LASSO

Types of regularized linear regression

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \underbrace{\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2}_{\text{loss function}} + \underbrace{\lambda P(\boldsymbol{\beta})}_{\substack{\text{penalty} \\ \text{term}}}$$

Ridge regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_2^2 = \sum_{j=1}^{p} \beta_j^2$$

+ Generally good for prediction (MSE existence theorem)
+ Coefficients of correlated features tend to be shrunk together (i.e., are similar)
- Dense model (no feature selection)

LASSO regression

$$P(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|$$

+ Performs automatic feature selection (i.e., "zeros" out features; induces *sparsity*)
+ Often selects one feature from correlated group (all others will be zeroed out)
- Gives lower accuracy if truth is not sparse

Elastic net regression

$$P(\boldsymbol{\beta}) = \alpha\|\boldsymbol{\beta}\|_1 + (1-\alpha)\|\boldsymbol{\beta}\|_2^2$$

+ Compromise between ridge and LASSO
- More difficult to tune two hyperparameters

Another bias-variance decomposition example

$$\mathbb{E}[(y - \hat{f}(x))^2] = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \sigma^2$$

Population MSE

Irreproducible Error

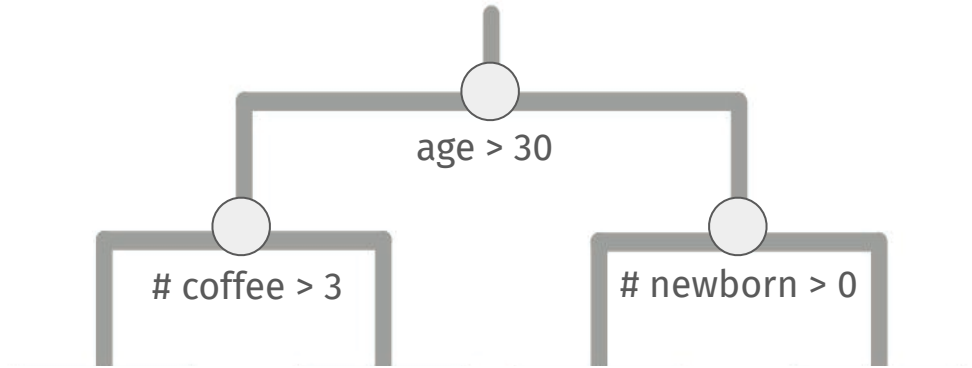
The phenomenon of the bias-variance tradeoff is not limited to linear models

There are also examples of the bias-variance tradeoff at work in nonlinear models:

- **Tree-based Models**
- Deep learning
  - Many techniques such as dropout, early stopping, algorithmic regularization (e.g., using stochastic gradient descent) are applied to reduce the variance of the deep learning model

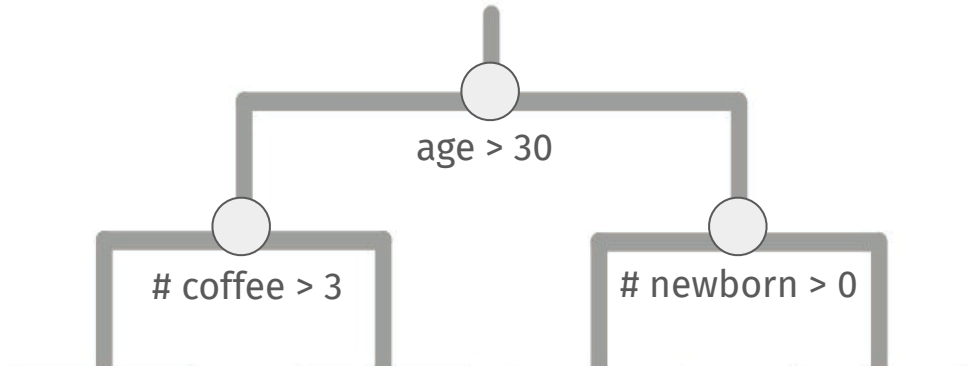
Decision Tree Model

Decision tree: a sequence of binary (yes/no) decisions based upon certain features and their feature values



Decision Tree Model

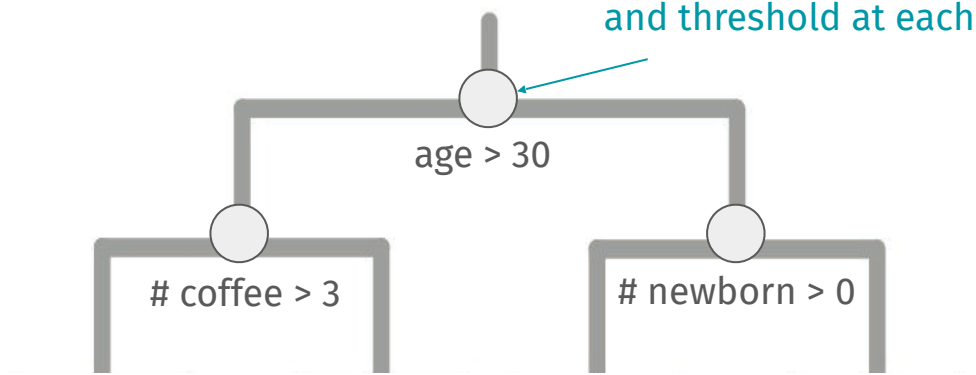
Decision tree: a sequence of binary (yes/no) decisions based upon certain features and their feature values



Decision Tree Model

Decision tree: a sequence of binary (yes/no) decisions based upon certain features and their feature values

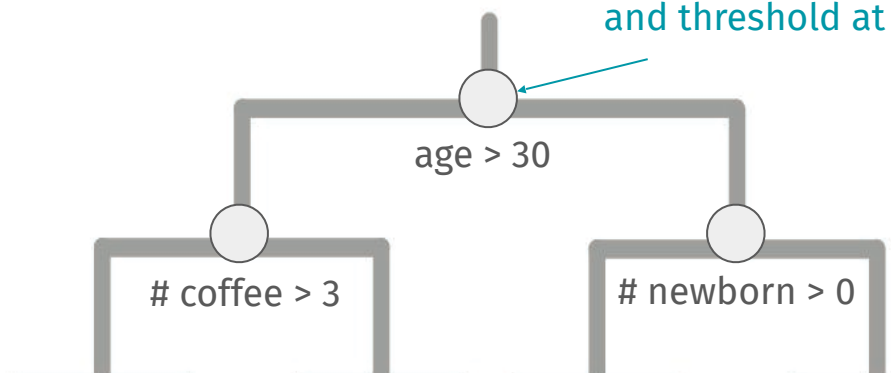
How do we identify the best feature and threshold at each split?



Decision Tree Model

Decision tree: a sequence of binary (yes/no) decisions based upon certain features and their feature values

How do we identify the best feature and threshold at each split?

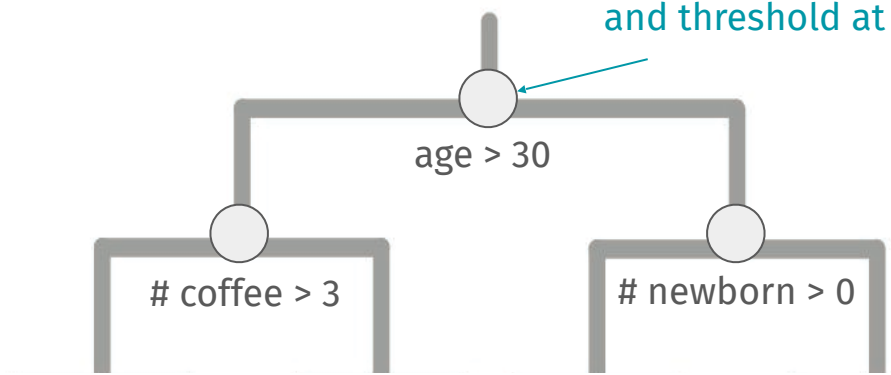


How do we determine when to stop growing the tree?

Decision Tree Model

Decision tree: a sequence of binary (yes/no) decisions based upon certain features and their feature values

How do we identify the best feature and threshold at each split?



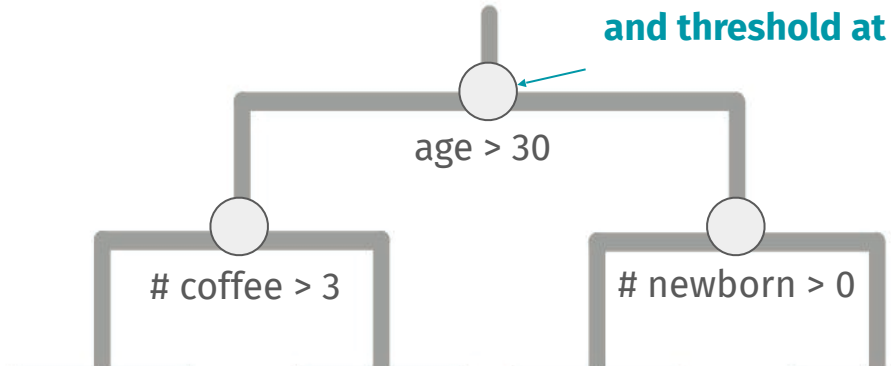
How do we determine when to stop growing the tree?

How do we make predictions at each leaf (terminal) node

Decision Tree Model

Decision tree: a sequence of binary (yes/no) decisions based upon certain features and their feature values

How do we identify the best feature and threshold at each split?

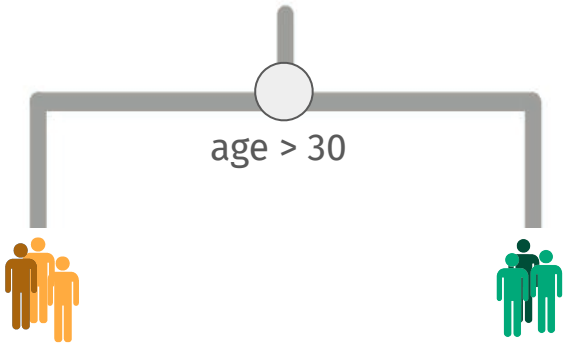


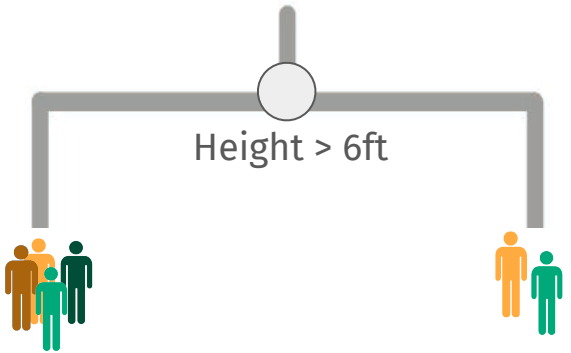
How do we determine when to stop growing the tree?

How do we make predictions at each leaf (terminal) node

Decision Tree Model

- **How do we identify the best feature and threshold at each split in the tree?**
  - Brute-force search over all possible combinations of features and thresholds (there are fast ways to do this for trees)
  - Choose the feature and threshold which maximizes (or minimizes) some splitting criterion (or loss function)
  - A common splitting criterion, **decrease in impurity**, results in choosing the feature and threshold which yields the two most "pure" (or homogeneous) groups after making the split





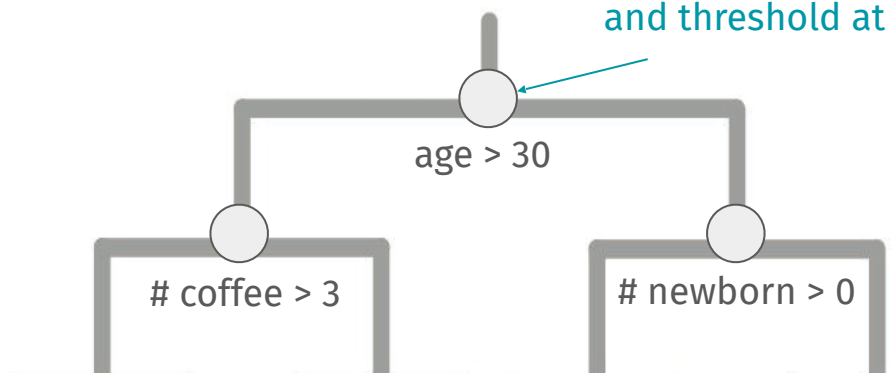
Decision Tree Model

- **How do we identify the best feature and threshold at each split in the tree?**
  - Brute-force search over all possible combinations of features and thresholds (there are fast ways to do this for trees)
  - Choose the feature and threshold which maximizes (or minimizes) some splitting criterion (or loss function)
  - A common splitting criterion, **decrease in impurity**, results in choosing the feature and threshold which yields the two most "pure" (or homogeneous) groups after making the split

Decision Tree Model

Decision tree: a sequence of binary (yes/no) decisions based upon certain features and their feature values

How do we identify the best feature and threshold at each split?



How do we determine when to stop growing the tree?

How do we make predictions at each leaf (terminal) node

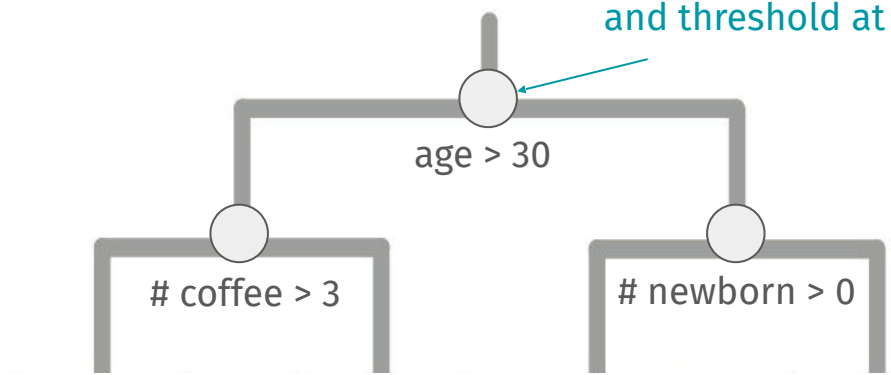
Decision Tree Model

- **How do we determine when to stop growing the tree?**
  - This is generally determined by one or more hyperparameters such as:
    - *Maximum depth of the tree* (grow until this depth is reached)
    - *Minimum number of samples* in each leaf node (grow until leaf node has x # of samples)
    - *Minimum decrease in impurity required* to split a node (grow until minimum decrease in impurity is not achieved by any possible feature/threshold combination)
    - …
  - Can tune these hyperparameter(s) via cross-validation

Decision Tree Model

Decision tree: a sequence of binary (yes/no) decisions based upon certain features and their feature values

How do we identify the best feature and threshold at each split?

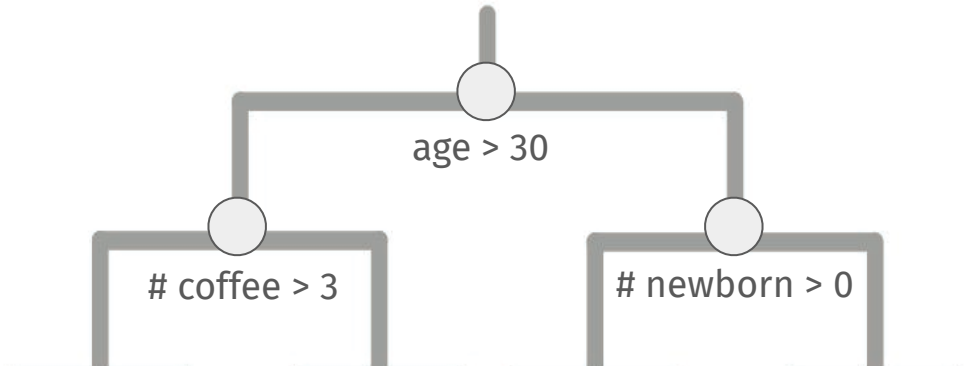


How do we determine when to stop growing the tree?

How do we make predictions at each leaf (terminal) node

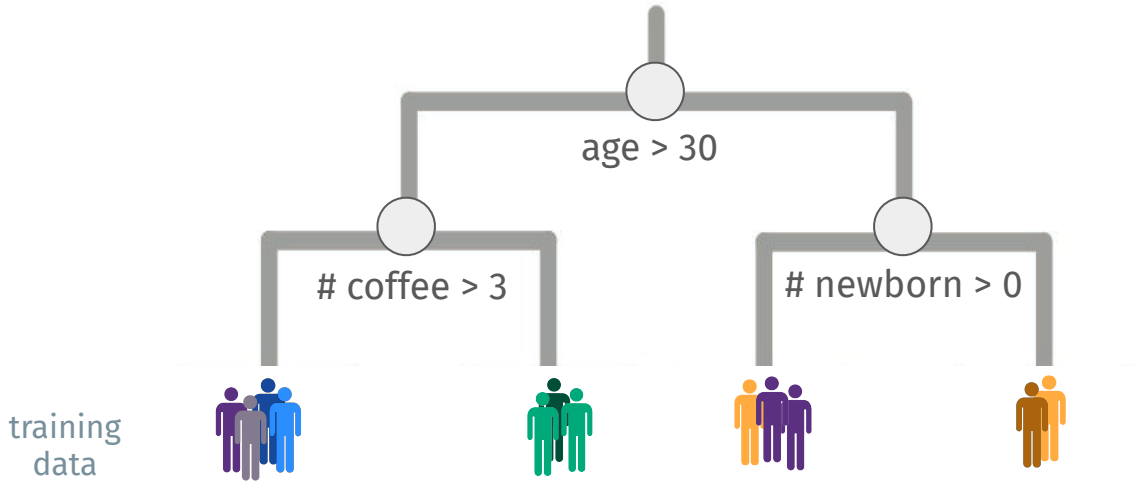
Decision Tree Model

- **How do we make predictions at each leaf (terminal) node**
  - Predicted value in each leaf node = mean response of all training observations that fell into that leaf node



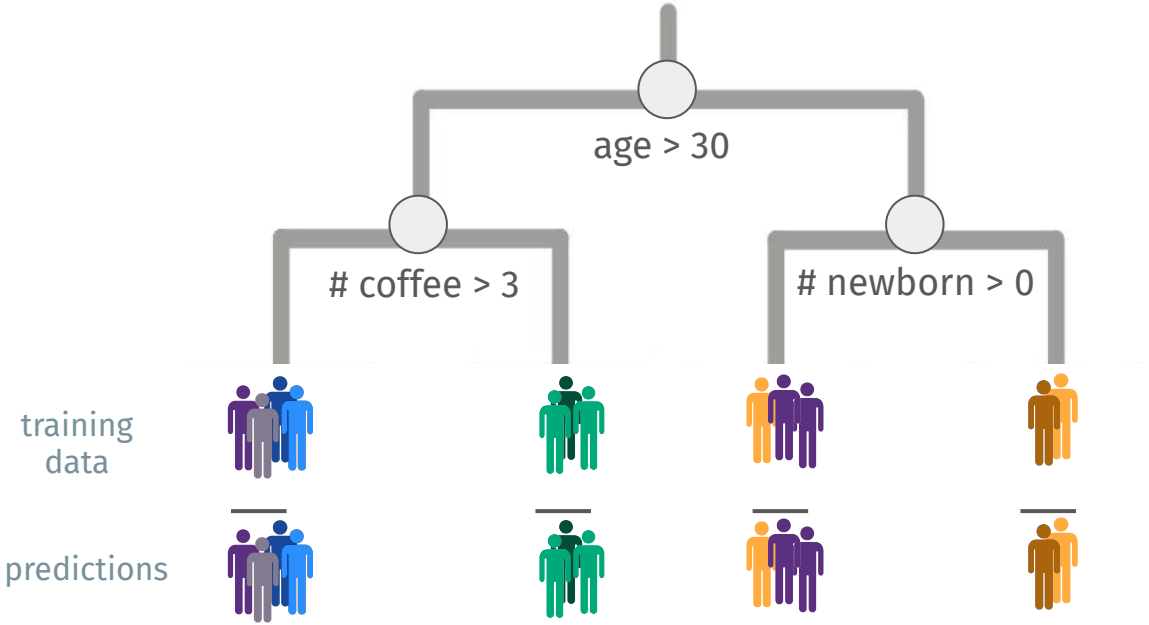
Decision Tree Model

- **How do we make predictions at each leaf (terminal) node**
  - Predicted value in each leaf node = mean response of all training observations that fell into that leaf node



Decision Tree Model

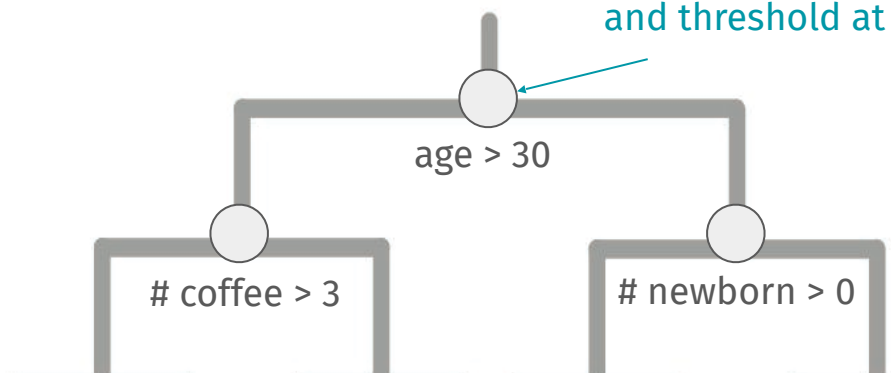
- **How do we make predictions at each leaf (terminal) node**
  - Predicted value in each leaf node = mean response of all training observations that fell into that leaf node



Decision Tree Model

Decision tree: a sequence of binary (yes/no) decisions based upon certain features and their feature values

How do we identify the best feature and threshold at each split?



How do we determine when to stop growing the tree?

How do we make predictions at each leaf (terminal) node

Random Forests (RF)

A **collection (or ensemble) of decision trees**, where

Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

A **collection (or ensemble) of decision trees**, where

Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

A **collection (or ensemble) of decision trees**, where

Leo Breiman. "Random forests." *Machine learning* (2001)

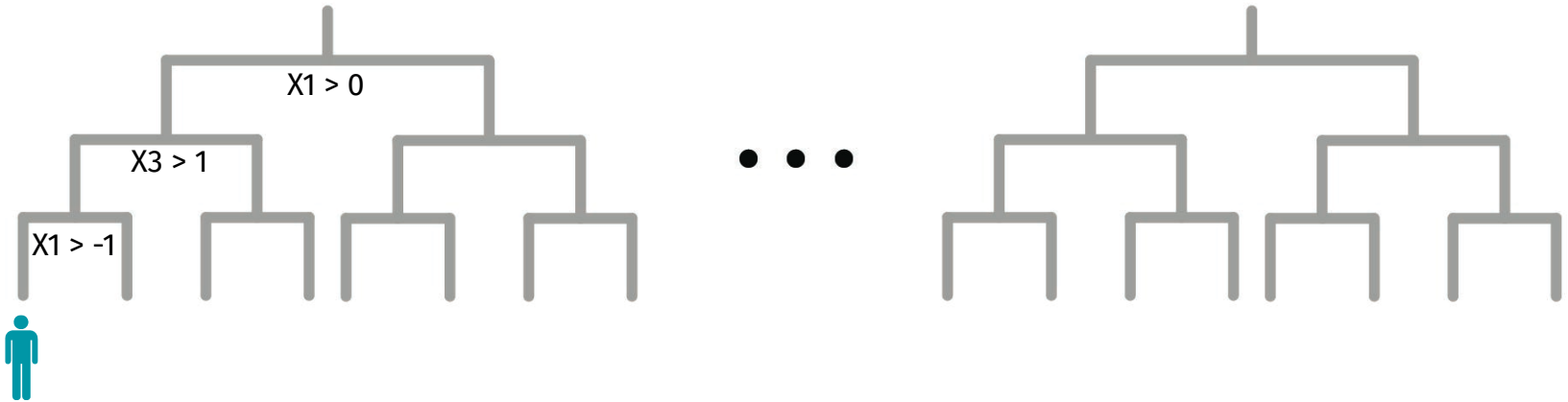
Random Forests (RF)

A **collection (or ensemble) of decision trees**, where

Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

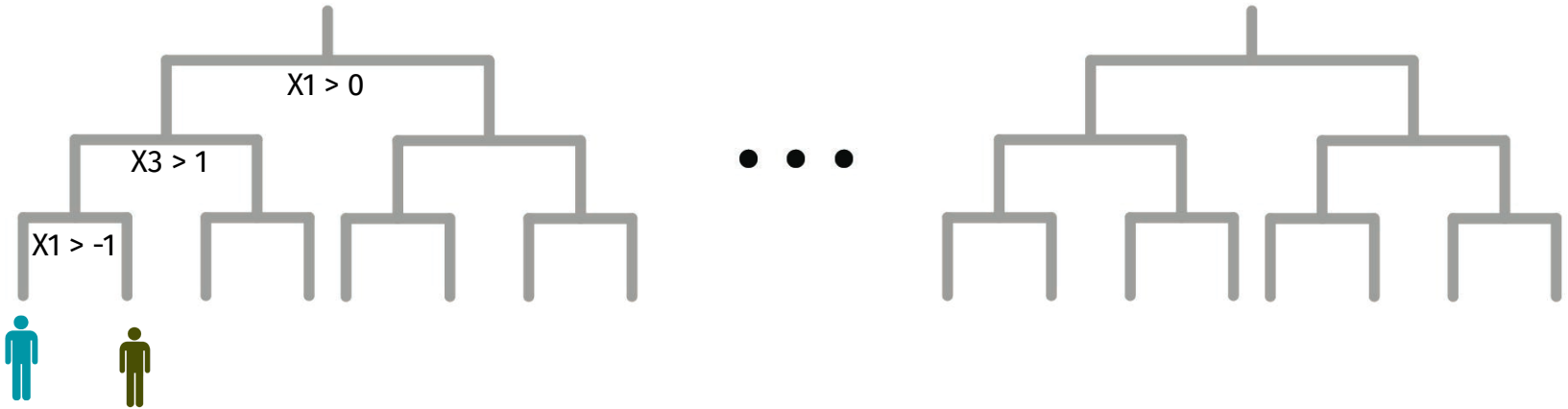
A **collection (or ensemble) of decision trees**, where



Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

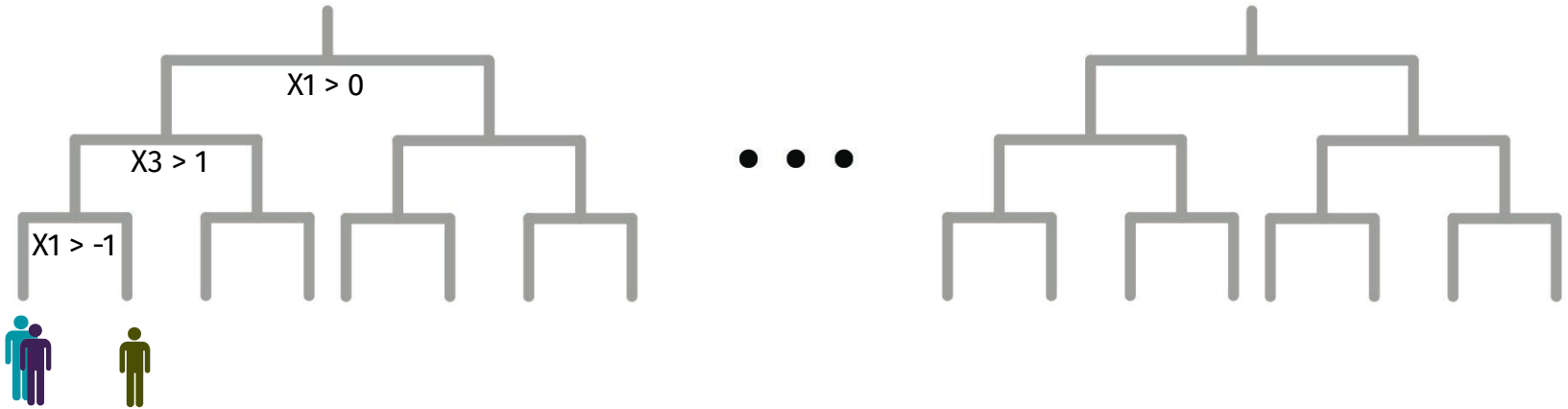
A **collection (or ensemble) of decision trees**, where



Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

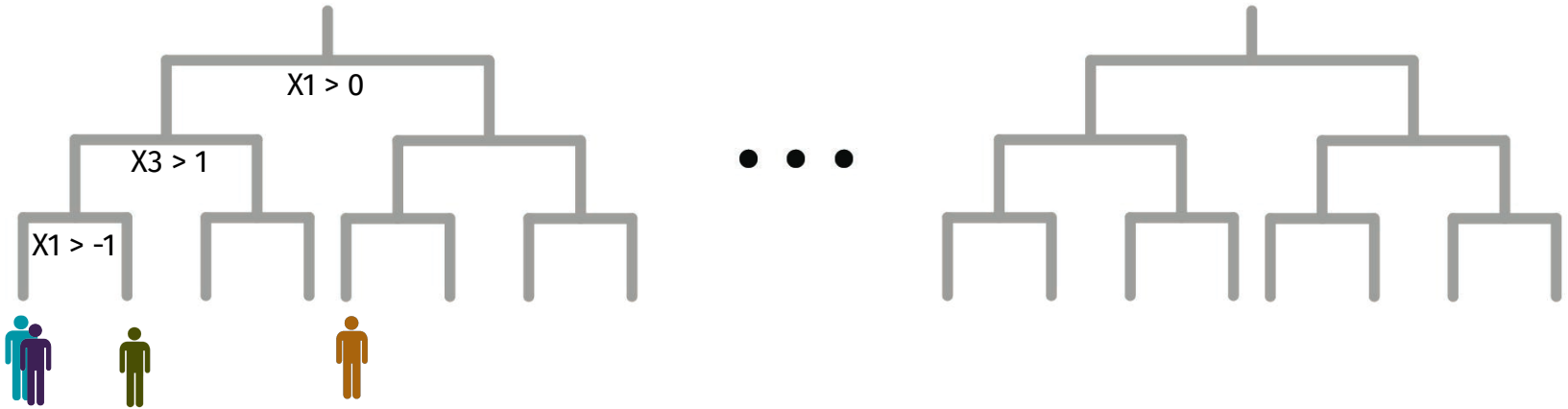
A **collection (or ensemble) of decision trees**, where



Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

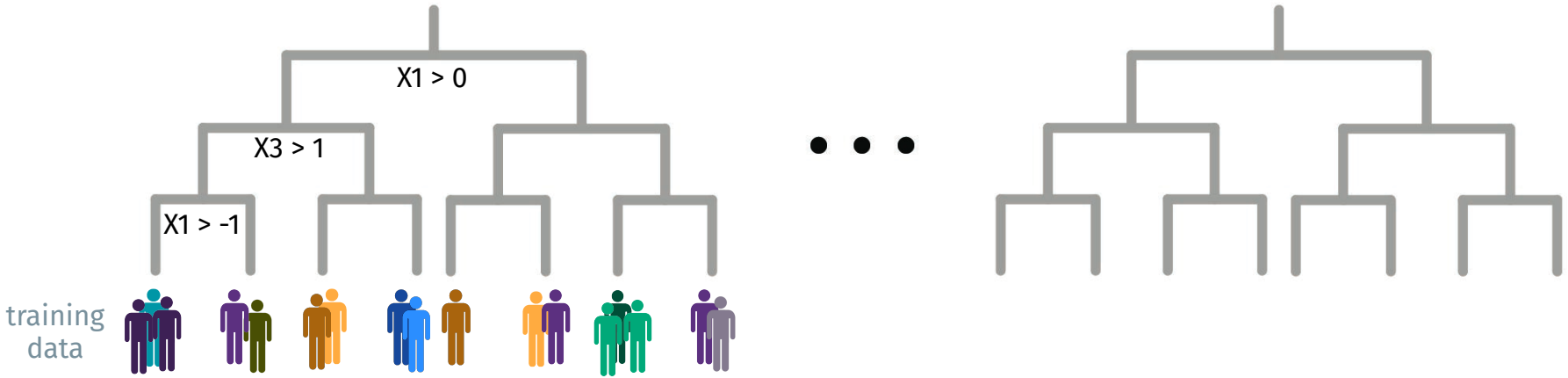
A **collection (or ensemble) of decision trees**, where



Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

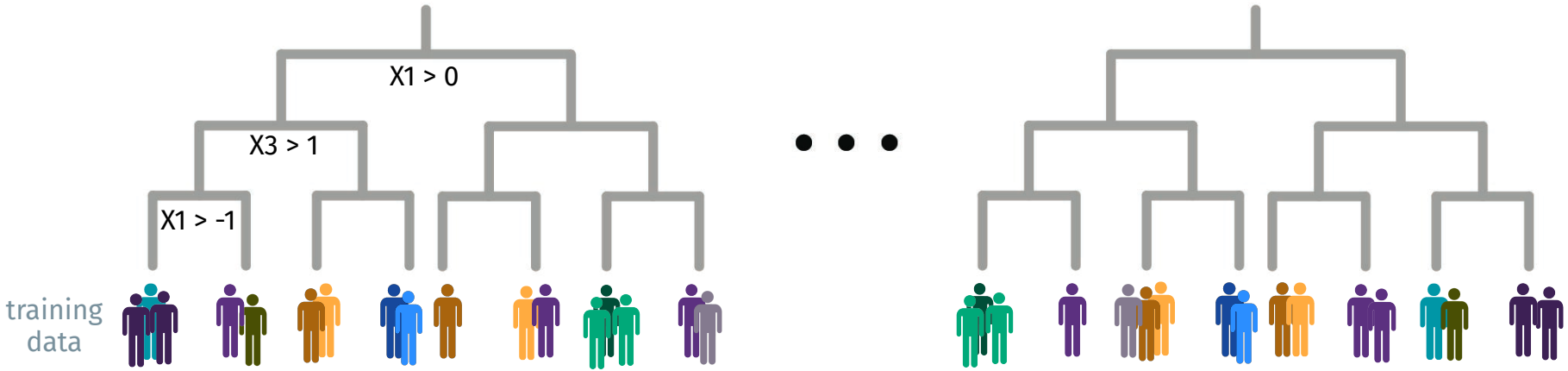
A **collection (or ensemble) of decision trees**, where



Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

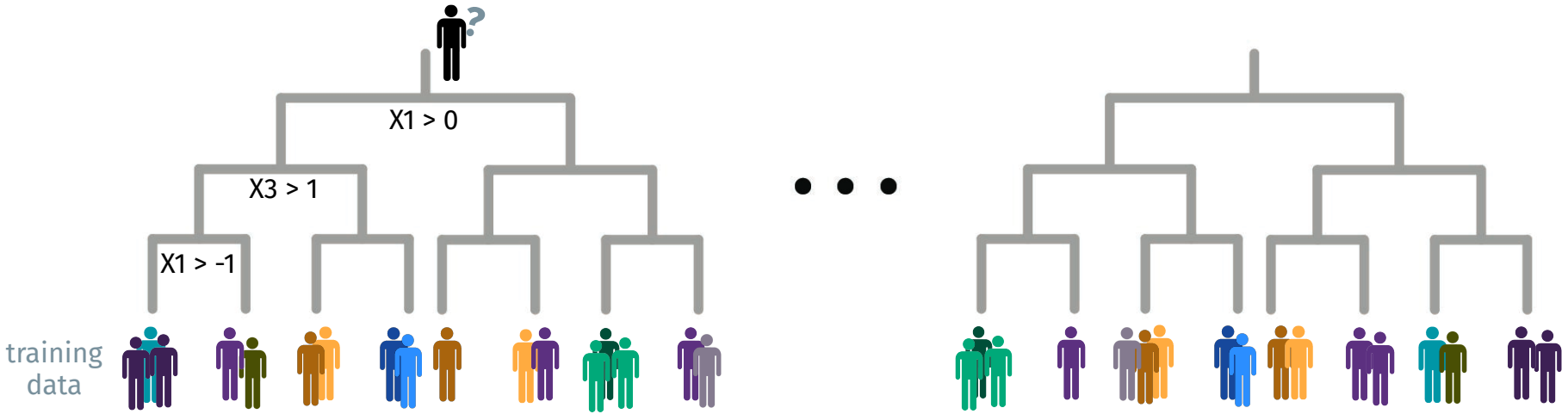
A **collection (or ensemble) of decision trees**, where



Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

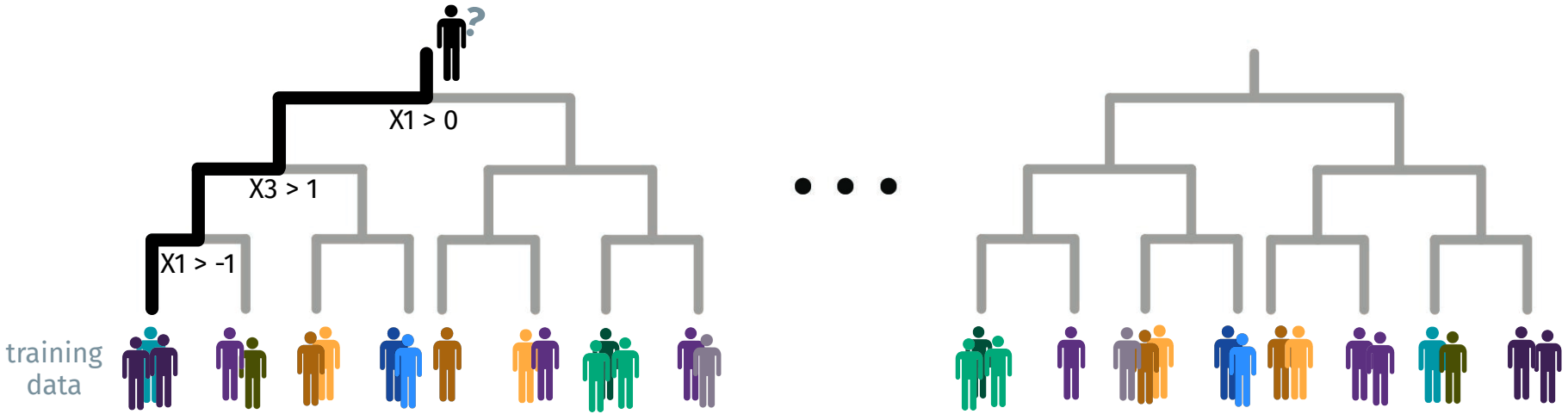
A **collection (or ensemble) of decision trees**, where



Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

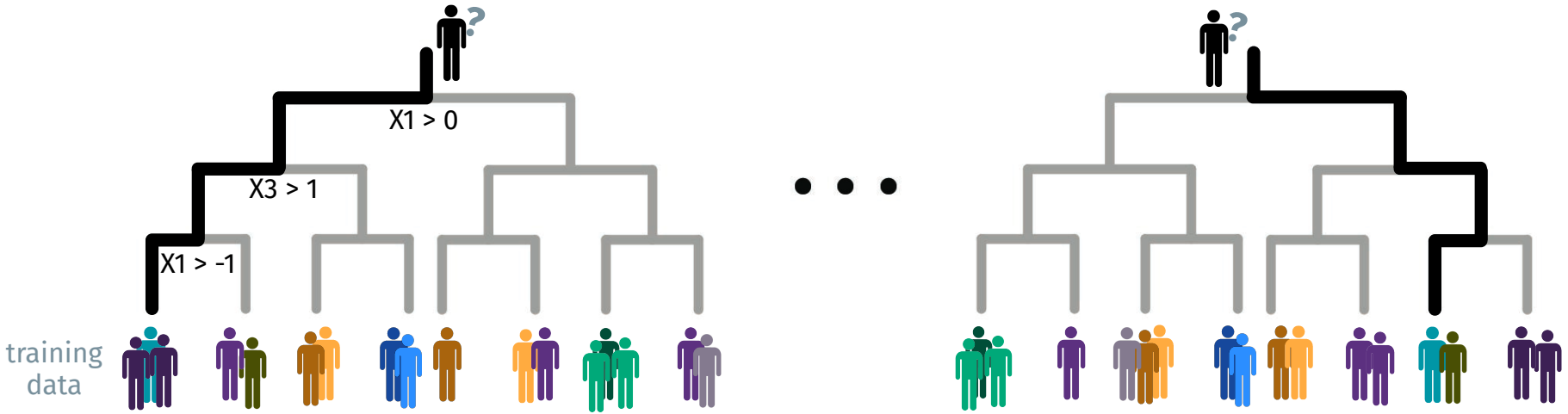
A **collection (or ensemble) of decision trees**, where



Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

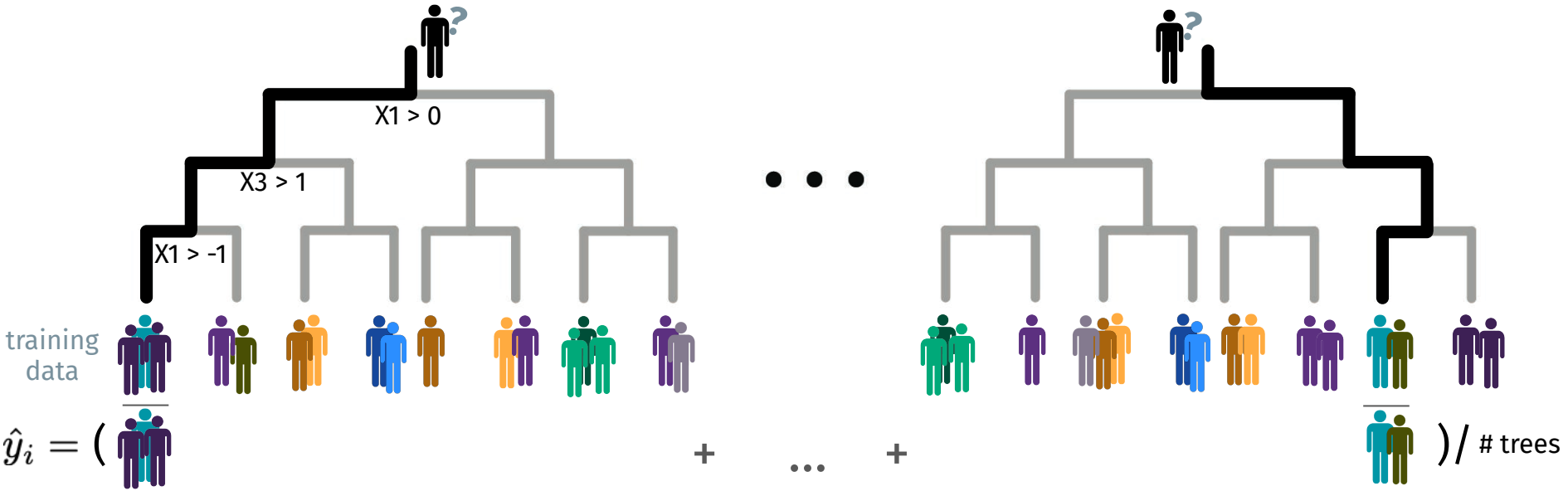
A **collection (or ensemble) of decision trees**, where



Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

A **collection (or ensemble) of decision trees**, where

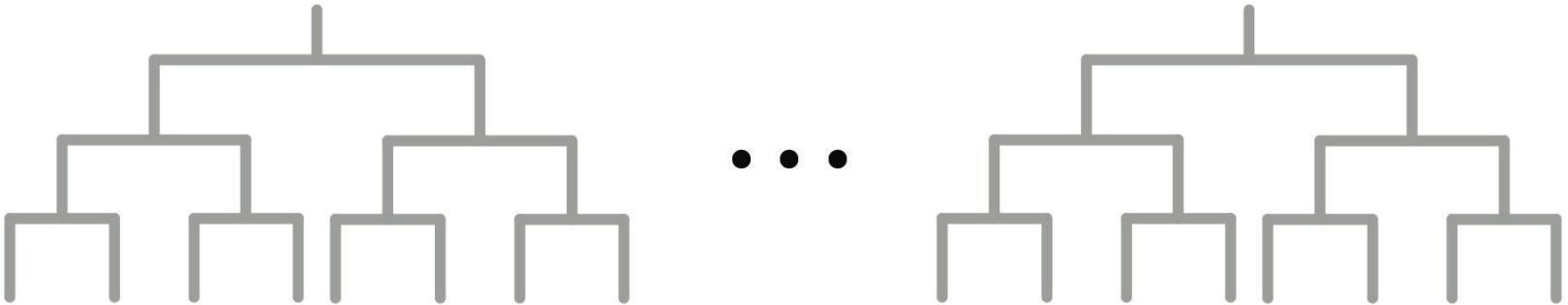


Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

A **collection (or ensemble) of decision trees**, where

- each tree is fitted on a different **bootstrap** version of the data
- **features are subsampled** at each node

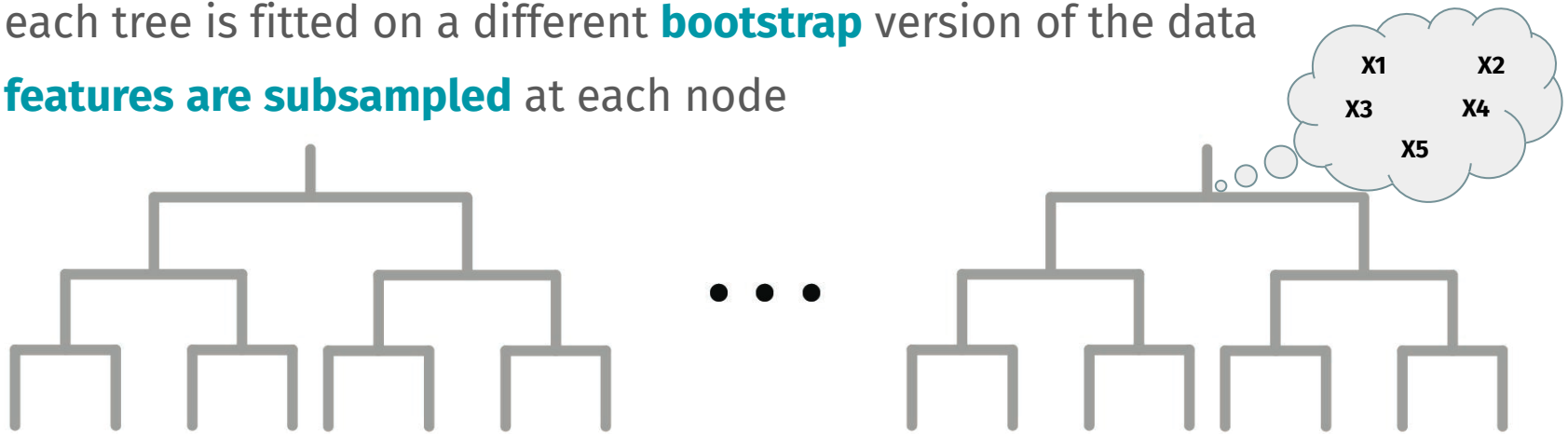


Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

A **collection (or ensemble) of decision trees**, where

- each tree is fitted on a different **bootstrap** version of the data
- **features are subsampled** at each node

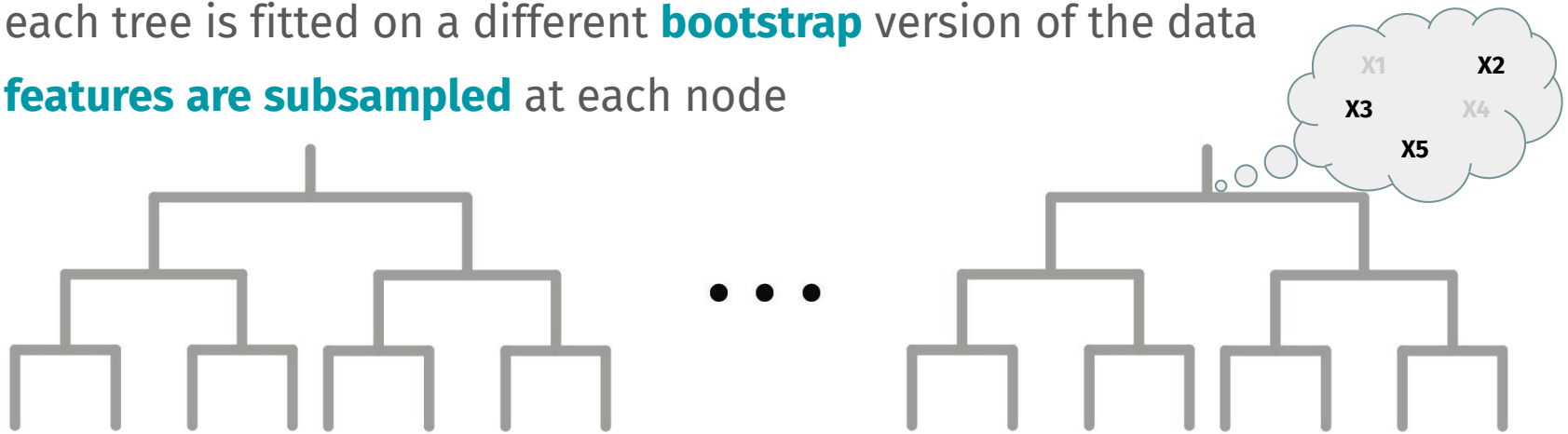


Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

A **collection (or ensemble) of decision trees**, where

- each tree is fitted on a different **bootstrap** version of the data
- **features are subsampled** at each node

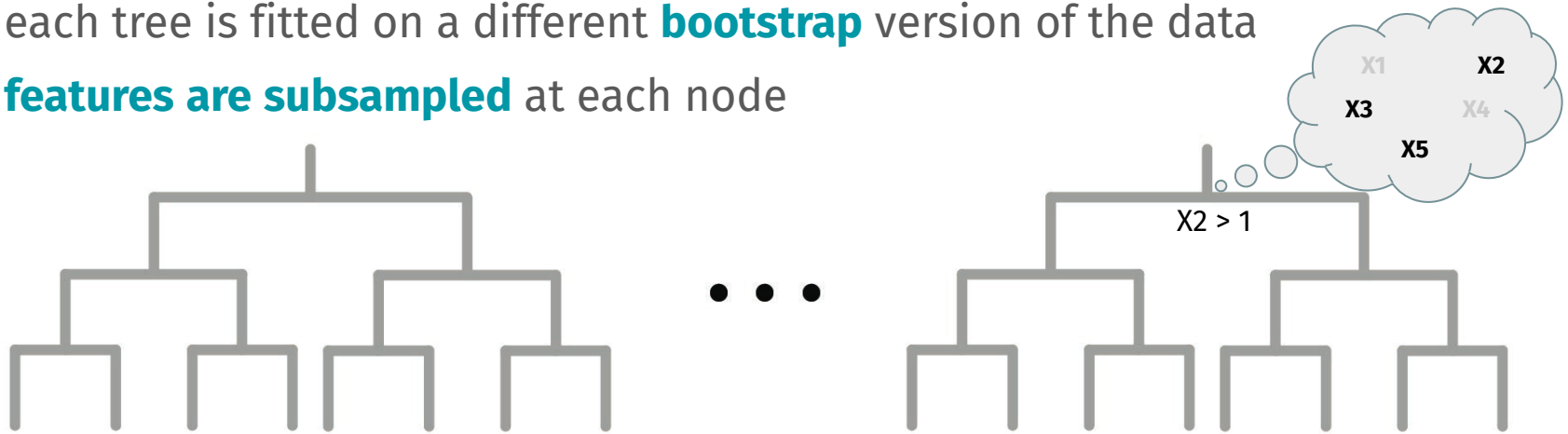


Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

A **collection (or ensemble) of decision trees**, where

- each tree is fitted on a different **bootstrap** version of the data
- **features are subsampled** at each node



Leo Breiman. "Random forests." *Machine learning* (2001)

Random Forests (RF)

Why do random forests work? Tldr; bias-variance decomposition

$$\underbrace{\mathbb{E}[(y - \hat{f}(x))^2]}_{\text{Population MSE}} = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \underbrace{\sigma^2}_{\text{Irreproducible Error}}$$

Random Forests (RF)

Why do random forests work? Tldr; bias-variance decomposition

$$\underbrace{\mathbb{E}[(y - \hat{f}(x))^2]}_{\text{Population MSE}} = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \underbrace{\sigma^2}_{\text{Irreproducible Error}}$$

- First, train *very deep* decision trees → each tree has low bias but high variance

Random Forests (RF)

Why do random forests work? Tldr; bias-variance decomposition

$$\underbrace{\mathbb{E}[(y - \hat{f}(x))^2]}_{\text{Population MSE}} = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \underbrace{\sigma^2}_{\text{Irreproducible Error}}$$

- First, train *very deep* decision trees → each tree has low bias but high variance
- Second, inject *randomness* into the training process through bootstrapping and feature subsampling → trees become more independent of each other

Random Forests (RF)

Why do random forests work? Tldr; bias-variance decomposition

$$\underbrace{\mathbb{E}[(y - \hat{f}(x))^2]}_{\text{Population MSE}} = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \underbrace{\sigma^2}_{\text{Irreproducible Error}}$$

- First, train *very deep* decision trees → each tree has low bias but high variance
- Second, inject *randomness* into the training process through bootstrapping and feature subsampling → trees become more independent of each other
  - Averaging across somewhat independent trees reduces the variance of the ensembled model

Random Forests (RF)

Why do random forests work? Tldr; bias-variance decomposition

$$\underbrace{\mathbb{E}[(y - \hat{f}(x))^2]}_{\text{Population MSE}} = \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) + \underbrace{\sigma^2}_{\text{Irreproducible Error}}$$

- First, train *very deep* decision trees → each tree has low bias but high variance
- Second, inject *randomness* into the training process through bootstrapping and feature subsampling → trees become more independent of each other
  - Averaging across somewhat independent trees reduces the variance of the ensembled model
  - Why is this?

Recap + Next Time

- The **#1 goal in supervised learning** is to **generalize** well to **new, unseen data**
- Generalizability is closely connected to the **bias-variance decomposition**
- In an ideal world, a good model has **low bias** and **low variance**

Next Time:

- How do we truthfully assess generalizability → data splitting

Final Project

Mid-Semester Feedback Survey

Please fill out anonymous mid-semester feedback survey:
https://forms.gle/KoXjt3CbY91GKk7V8