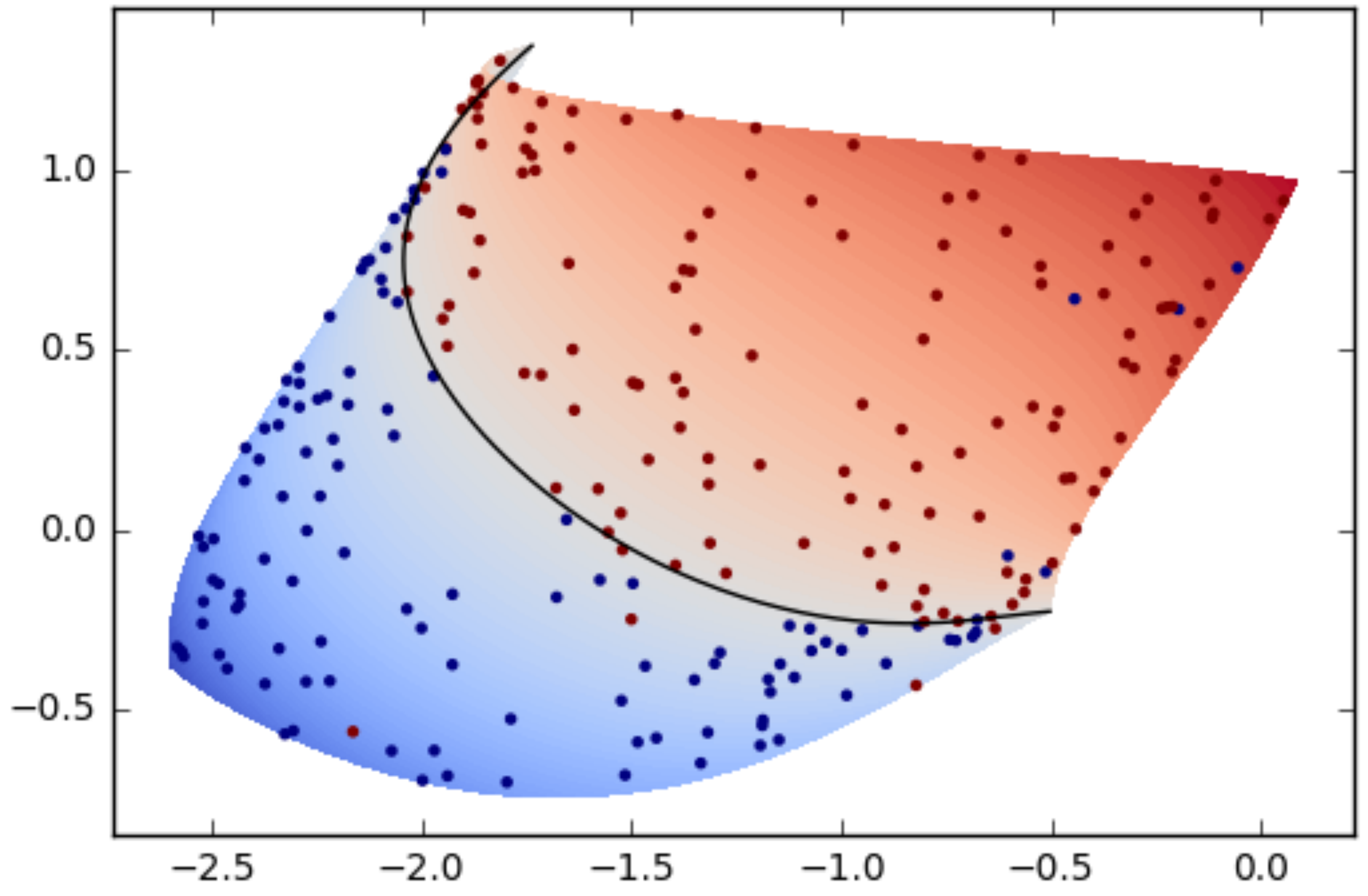
CS 273A: Machine Learning

Fall 2021

Lecture 4: Linear Regression

Roy Fox
Department of Computer Science
Bren School of Information and Computer Sciences
University of California, Irvine

All slides in this course adapted from Alex Ihler & Sameer Singh



Logistics

assignments

- Assignment 1 due Thursday
- Assignment 2 to be published later this week

Today's lecture

Naïve Bayes Classifiers

Bayes error

ROC curves

Linear regression

Representing joint distributions

- Assume data with binary features
- How to represent $p(x \mid y)$?
- Create a truth table of all x values

A	B	C
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Representing joint distributions

- Assume data with binary features
- How to represent $p(x \mid y)$?
- Create a truth table of all x values
- Specify $p(x \mid y)$ for each cell
- How many parameters?
  - $2^n - 1$

A	B	C	p(A,B,C | y=1)
0	0	0	0.50
0	0	1	0.05
0	1	0	0.01
0	1	1	0.10
1	0	0	0.04
1	0	1	0.15
1	1	0	0.05
1	1	1	0.10

Estimating joint distributions

- Can we estimate $p(x \mid y)$ from data?
- Count how many data points for each x?
  - If $m \ll 2^n$, most instances never occur
    - Do we predict that missing instances are impossible?
      - What if they occur in test data?
- Difficulty to represent and estimate go hand in hand
  - Model complexity → overfitting!

A	B	C	p(A,B,C | y=1)
0	0	0	4/10
0	0	1	1/10
0	1	0	0/10
0	1	1	0/10
1	0	0	1/10
1	0	1	2/10
1	1	0	1/10
1	1	1	1/10

Regularization

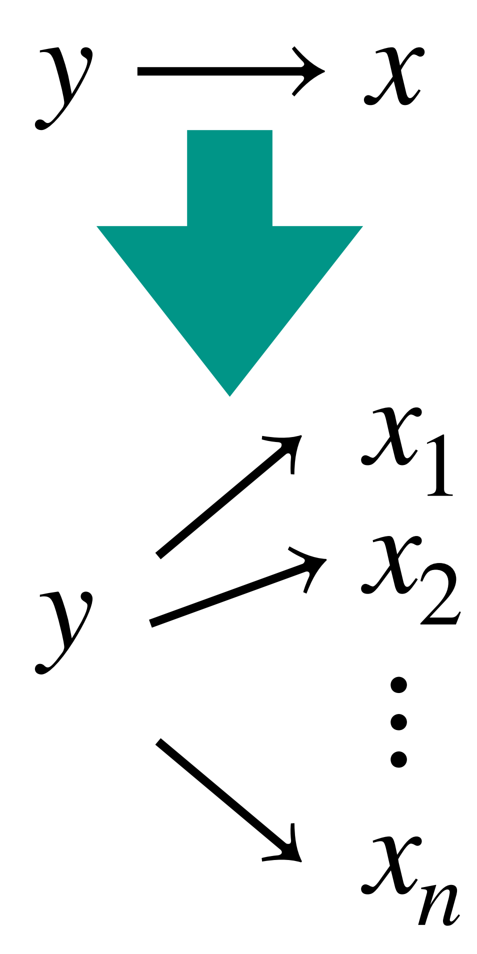
- Reduce effective size of model class
  - Hope to avoid overfitting
- One way: make the model more "regular", less sensitive to data quirks
- Example: add small "pseudo-count" to the counts (before normalizing)
  - $\hat{p}(x \mid y = c) = \dfrac{\#_c(x) + \alpha}{m_c + \alpha \cdot 2^n}$
  - Not a huge help here, most cells will be uninformative $\dfrac{\alpha}{m_c + \alpha \cdot 2^n}$

Simplifying the model

- Another way: reduce model complexity

- Example: assume features are independent of one another (in each class)
  - $p(x_1, x_2, \ldots, x_n \mid y) = p(x_1 \mid y)p(x_2 \mid y)\cdots p(x_n \mid y)$

- Now we only need to represent / estimate each $p(x_i \mid y)$ individually



A	p(A |y=1)
0	.4
1	.6

B	p(B |y=1)
0	.7
1	.3

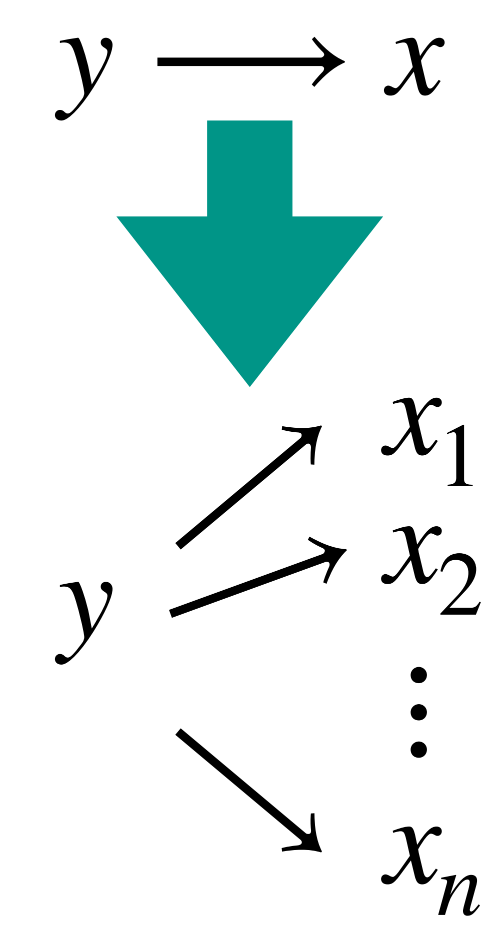
C	p(C |y=1)
0	.1
1	.9

$\Longrightarrow$

A	B	C	p(A,B,C | y=1)
0	0	0	.4 * .7 * .1
0	0	1	.4 * .7 * .9
0	1	0	.4 * .3 * .1
0	1	1	…
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Naïve Bayes models

- We want to predict some value y, e.g. auto accident next year
- We have many known indicators for y (covariates) $x = x_1, \ldots, x_n$
  - E.g., age, income, education, zip code, ...
  - Learn $p(y \mid x_1, \ldots, x_n)$ — but cannot represent / estimate $O(2^n)$ values
- Naïve Bayes
  - Estimate prior distribution $\hat{p}(y)$
  - Assume $p(x_1, \ldots, x_n \mid y) = \prod_i p(x_i \mid y)$, estimate covariates independently $\hat{p}(x_i \mid y)$
  - Model: $\hat{p}(y \mid x) \propto \hat{p}(y) \prod_i \hat{p}(x_i \mid y)$



causal structure wrong!
(but useful...)

Naïve Bayes models: example

- $y \in \{\text{spam}, \text{not spam}\}$
- x = observed words in email
  - E.g., [“the” ... “probabilistic” ... “lottery” ...]
  - $x = [0,1,0,0,\ldots,0,1]$ (1 = word appears; 0 = otherwise)
- Representing $p(x \mid y)$ directly would require $2^{\text{thousands}}$ parameters
- Represent each word indicator as independent (given class)
  - Reducing model complexity to thousands of parameters
- Words more likely in spam pull towards higher $p(\text{spam} \mid x)$, and v.v.

Numeric example

- $\hat{p}(y=1) = \frac{4}{8} \qquad = 1 - \hat{p}(y=0)$

- $\hat{p}(x_1, x_2 \mid y) = \hat{p}(x_1 \mid y)\hat{p}(x_2 \mid y)$

- $\hat{p}(x_1 = 1 \mid y = 0) = \frac{3}{4} \qquad \hat{p}(x_1 = 1 \mid y = 1) = \frac{2}{4}$

- $\hat{p}(x_2 = 1 \mid y = 0) = \frac{2}{4} \qquad \hat{p}(x_2 = 1 \mid y = 1) = \frac{1}{4}$

- What to predict for $x_1, x_2 = 1,1$? **prediction:** $\hat{y} = 0$
  - $\hat{p}(y=0)\hat{p}(x=1,1 \mid y=0) = \frac{4}{8} \cdot \frac{3}{4} \cdot \frac{2}{4} \qquad \hat{p}(y=1)\hat{p}(x=1,1 \mid y=1) = \frac{4}{8} \cdot \frac{2}{4} \cdot \frac{1}{4}$

x_1	x_2	y
1	1	0
1	0	0
1	0	1
0	0	0
0	1	1
1	1	0
0	0	1
1	0	1

Numeric example

- $\hat{p}(y=1) = \dfrac{4}{8} \qquad = 1 - \hat{p}(y=0)$
- $\hat{p}(x_1, x_2 \mid y) = \hat{p}(x_1 \mid y)\hat{p}(x_2 \mid y)$
- $\hat{p}(x_1 = 1 \mid y = 0) = \dfrac{3}{4} \qquad \hat{p}(x_1 = 1 \mid y = 1) = \dfrac{2}{4}$
- $\hat{p}(x_2 = 1 \mid y = 0) = \dfrac{2}{4} \qquad \hat{p}(x_2 = 1 \mid y = 1) = \dfrac{1}{4}$
- What is $\hat{p}(y = 1 \mid x_1 = 1, x_2 = 1)$?

x_1	x_2	y
1	1	0
1	0	0
1	0	1
0	0	0
0	1	1
1	1	0
0	0	1
1	0	1

$$\frac{\hat{p}(y=1)\hat{p}(x=1,1 \mid y=1)}{\hat{p}(x=1,1)} = \frac{\hat{p}(y=1)\hat{p}(x=1,1 \mid y=1)}{\hat{p}(y=0)\hat{p}(x=1,1 \mid y=0) + \hat{p}(y=1)\hat{p}(x=1,1 \mid y=1)} = \frac{\frac{4}{8}\cdot\frac{2}{4}\cdot\frac{1}{4}}{\frac{4}{8}\cdot\frac{3}{4}\cdot\frac{2}{4} + \frac{4}{8}\cdot\frac{2}{4}\cdot\frac{1}{4}} = \frac{1}{4}$$

Recap

- Bayes' rule: $p(y \mid x) = \dfrac{p(y)p(x \mid y)}{p(x)}$
- Bayes classifiers: estimate $p(y)$ and $p(x \mid y)$ from data
- Naïve Bayes classifiers: assume independent features $p(x \mid y) = \prod_i p(x_i \mid y)$
  - Estimate each $p(x_i \mid y)$ individually
- Maximum posterior (MAP): $\hat{y}(x) = \arg\max_y p(y \mid x) = \arg\max_y p(y)p(x \mid y)$
  - Normalizer $p(x)$ not needed

Today's lecture

Naïve Bayes Classifiers

Bayes error

ROC curves

Linear regression

Bayes classification error

- What is the training error of the MAP prediction $\hat{y}(x) = \arg\max_y p(y\,|\,x)$?

Features	# bad	# good	prediction:
X=0	42	15	bad
X=1	338	287	bad
X=2	3	5	good

errors

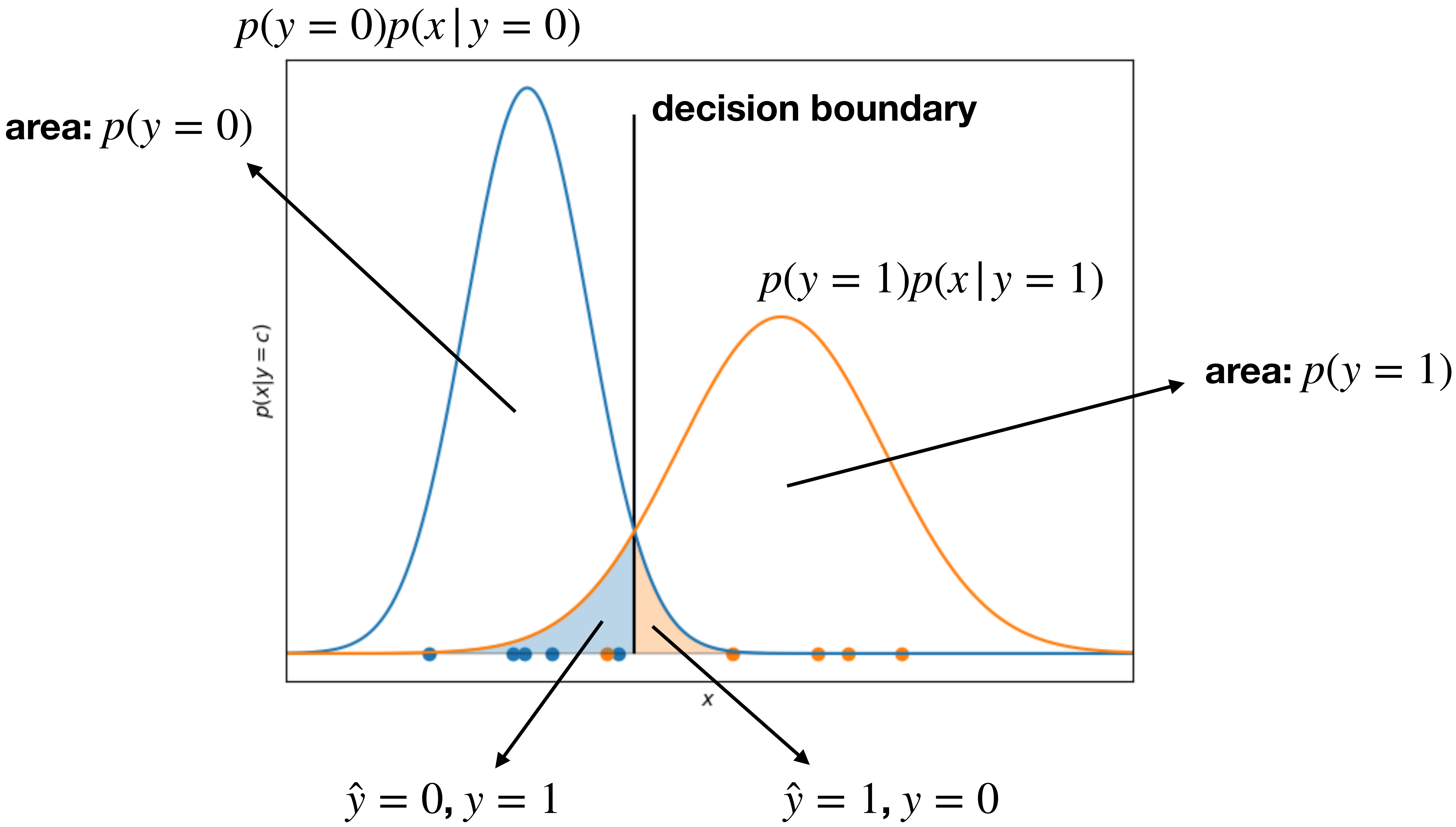
- $p(\hat{y} \neq y) = \dfrac{15 + 287 + 3}{690} = 0.442$

- Bayes error rate: probability of misclassification by MAP of <u>true posterior</u>

Bayes error rate

- Suppose that we know the true probabilities $p(x, y)$
  - And that we can compute prior $p(y)$ and posterior $p(y \mid x)$
- Bayes-optimal decision = MAP: $\hat{y} = \arg\max_y p(y \mid x)$
- Bayes error rate: $\mathbb{E}_{x,y\sim p}[\hat{y} \neq y] = \mathbb{E}_{x\sim p}[1 - \max_y p(y \mid x)]$
  - This is the optimal error rate of any classifier
  - Measures intrinsic hardness of separating y values given only x
    - But may get better with more features
- Normally we cannot estimate the Bayes error rate, only approximate with good classifier

Bayes error rate: Gaussian example



Today's lecture

Naïve Bayes Classifiers

Bayes error

ROC curves

Linear regression

Terminology

- Class prior probabilities: $p(y)$
  - Prior = before seeing any features
- Class-conditional probabilities: $p(x \mid y)$
- Class posterior probabilities: $p(y \mid x)$
- Bayes' rule: $p(y \mid x) = \dfrac{p(y)p(x \mid y)}{p(x)}$
- Law of total probability: $p(x) = \sum_y p(x, y) = \sum_y p(y)p(x \mid y)$

Measuring error

- Confusion matrix: all possible values of $(y, \hat{y})$

- Binary case: **true** / **false** (correct or not) **positive** / **negative** (prediction)

	Predict 0	Predict 1
Y=0	380 **TN**	5 **FP**
Y=1	338 **FN**	3 **TP**

  - Accuracy: $\dfrac{TP + TN}{TP + TN + FP + FN} = 1 -$ error rate

  - True positive rate (TPR): $\hat{p}(\hat{y} = 1 \mid y = 1) = \dfrac{\#(y = 1, \hat{y} = 1)}{\#(y = 1)}$ (aka, sensitivity)

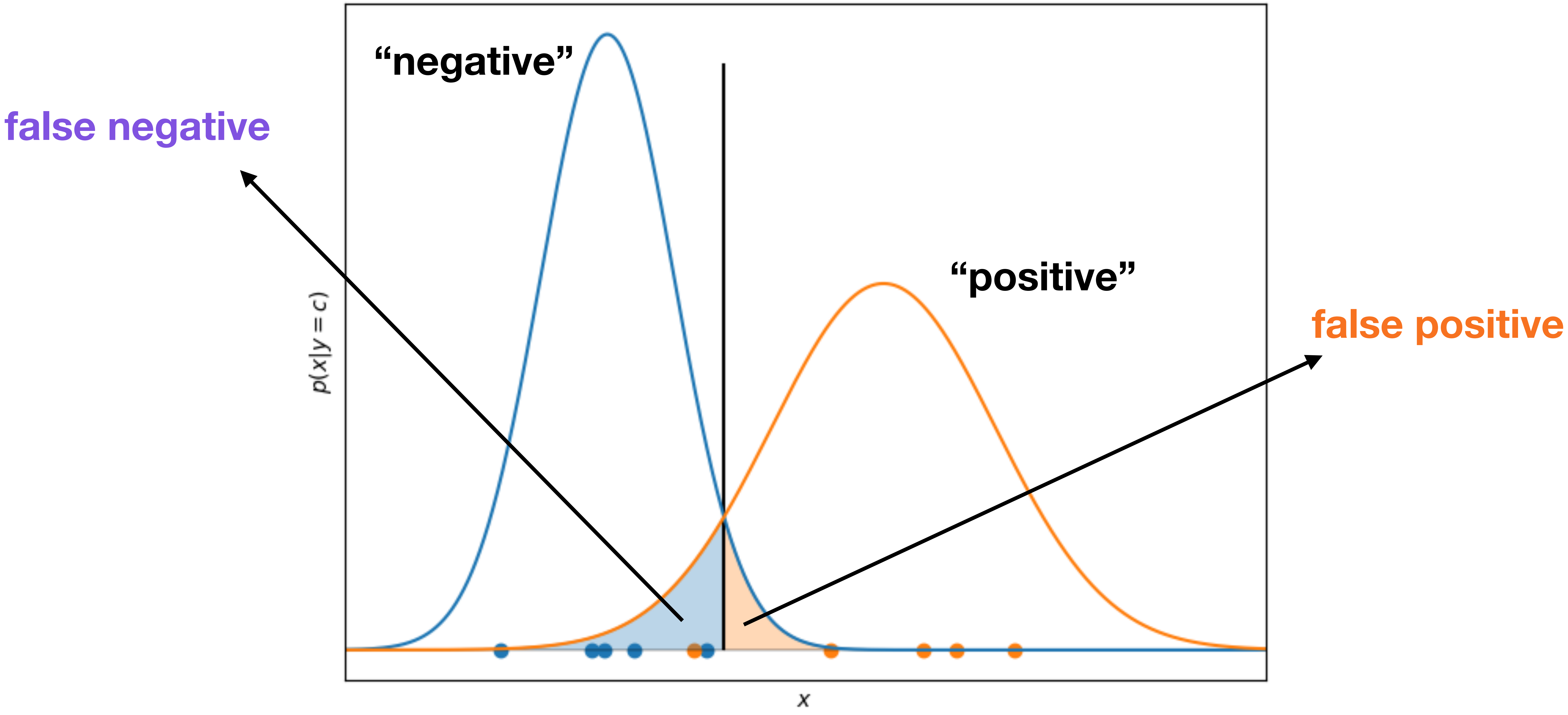
  - False negative rate (FNR): $\hat{p}(\hat{y} = 0 \mid y = 1) = \dfrac{\#(y = 1, \hat{y} = 0)}{\#(y = 1)}$

  - False positive rate (FPR): $\hat{p}(\hat{y} = 1 \mid y = 0) = \dfrac{\#(y = 0, \hat{y} = 1)}{\#(y = 0)}$

  - True negative rate (TNR): $\hat{p}(\hat{y} = 0 \mid y = 0) = \dfrac{\#(y = 0, \hat{y} = 0)}{\#(y = 0)}$ (aka, specificity)

Types of error

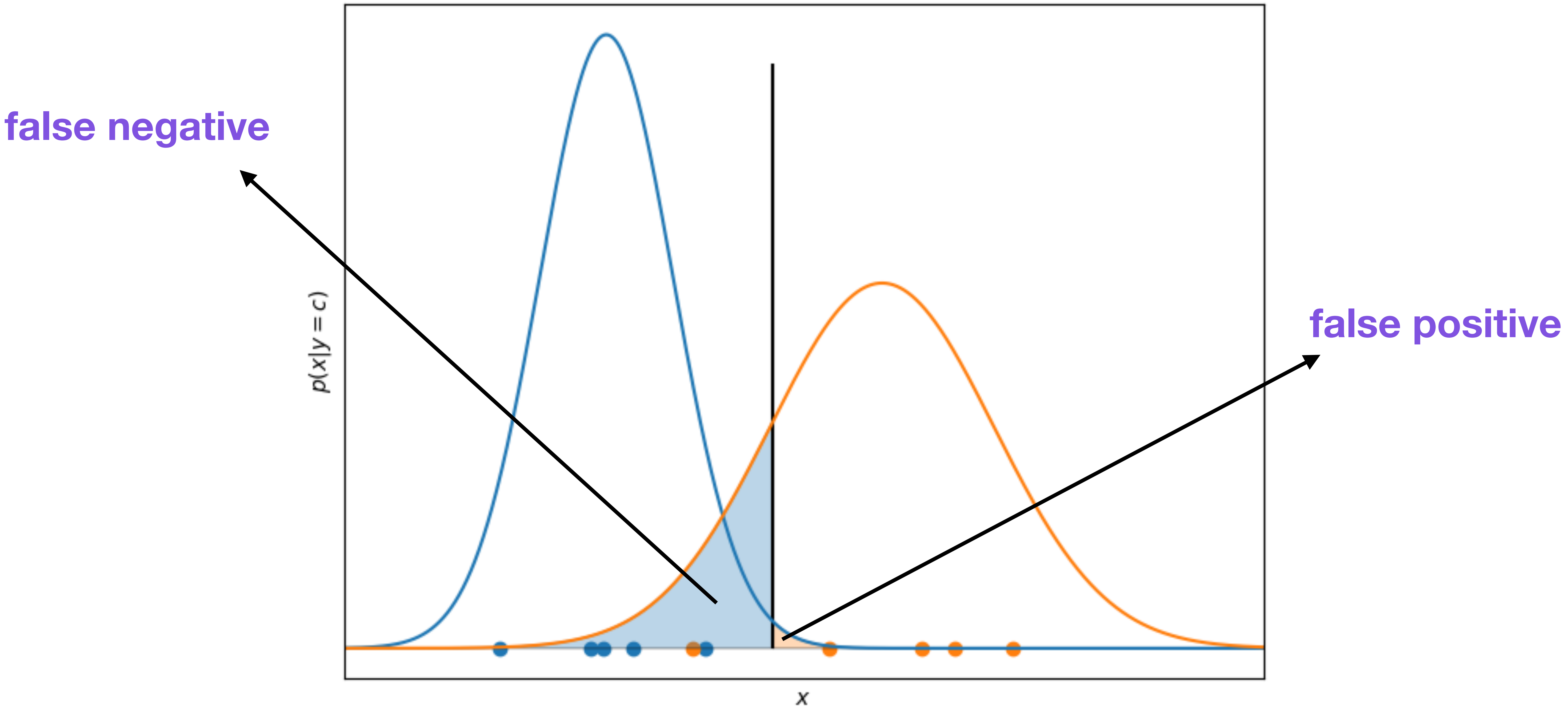
- Not all errors are equally bad
  - Do some cost more? (e.g. red / green light, diseased / healthy)



- False negative rate: $\frac{p(y=1, \hat{y}=0)}{p(y=1)}$; false positive rate: $\frac{p(y=0, \hat{y}=1)}{p(y=0)}$

Cost of error

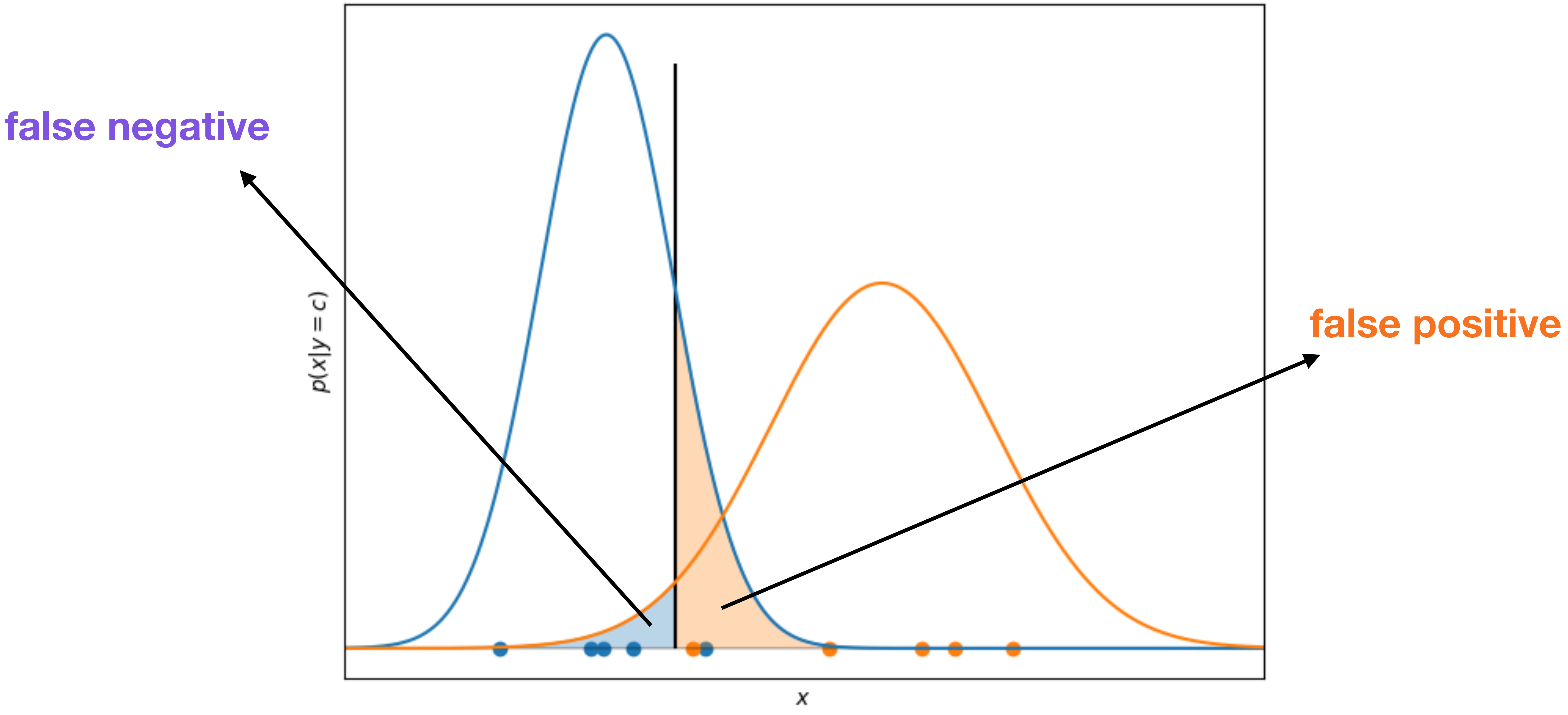
- Weight different costs differently
  - $\alpha \cdot p(y=0)p(x\,|\,y=0) \lessgtr p(y=1)p(x\,|\,y=1)$



- Increase α to prefer class 0 — increase FNR, decrease FPR

Cost of error

- Weight different costs differently
  - $\alpha \cdot p(y=0)p(x \mid y=0) \lessgtr p(y=1)p(x \mid y=1)$



- Decrease α to prefer class 1 — decrease FNR, increase FPR

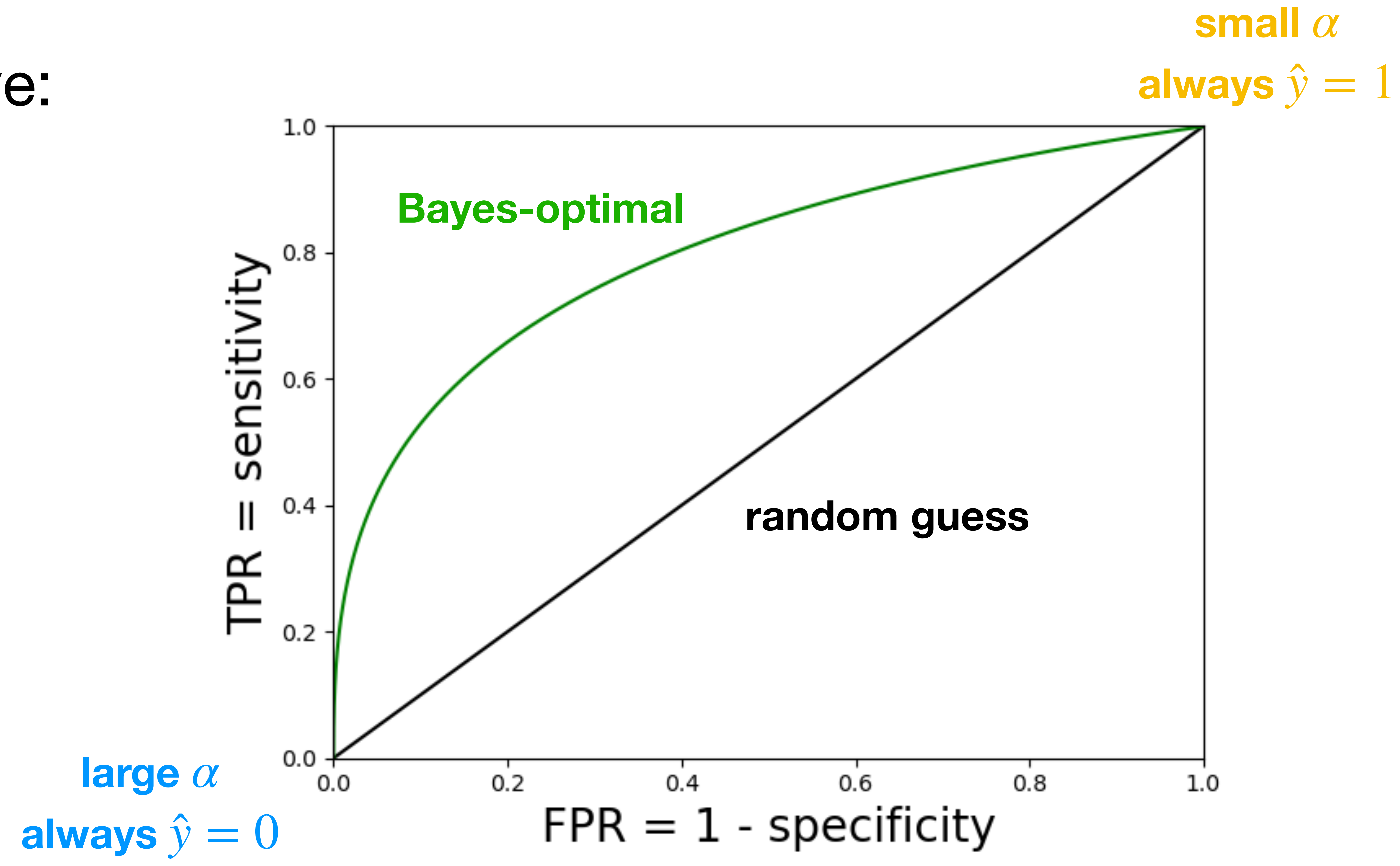
Bayes-optimal decision

- Maximum posterior decision: $\hat{p}(y=0 \mid x) \lessgtr \hat{p}(y=1 \mid x)$
  - Optimal for the error-rate (0–1) loss: $\mathbb{E}_{x,y\sim p}[\hat{y}(x) \neq y]$
- What if we have different cost for different errors? α_{FP}, α_{FN}
  - $\mathscr{L} = \mathbb{E}_{x,y\sim p}[\alpha_{\text{FP}} \cdot \#(y=0,\hat{y}(x)=1) + \alpha_{\text{FN}} \cdot \#(y=1,\hat{y}(x)=0)]$
- Bayes-optimal decision: $\alpha_{\text{FP}} \cdot \hat{p}(y=0 \mid x) \lessgtr \alpha_{\text{FN}} \cdot \hat{p}(y=1 \mid x)$
  - Log probability ratio: $\log \dfrac{\hat{p}(y=1 \mid x)}{\hat{p}(y=0 \mid x)} \lessgtr \log \dfrac{\alpha_{\text{FP}}}{\alpha_{\text{FN}}} = \alpha$

ROC curve

- Often models have a "knob" for tuning preference over classes (e.g. α)
  - Changing the decision boundary to include more instances in preferred class
- Characteristic performance curve:

$$\log \frac{\hat{p}(y=1\,|\,x)}{\hat{p}(y=0\,|\,x)} \lessgtr \alpha$$

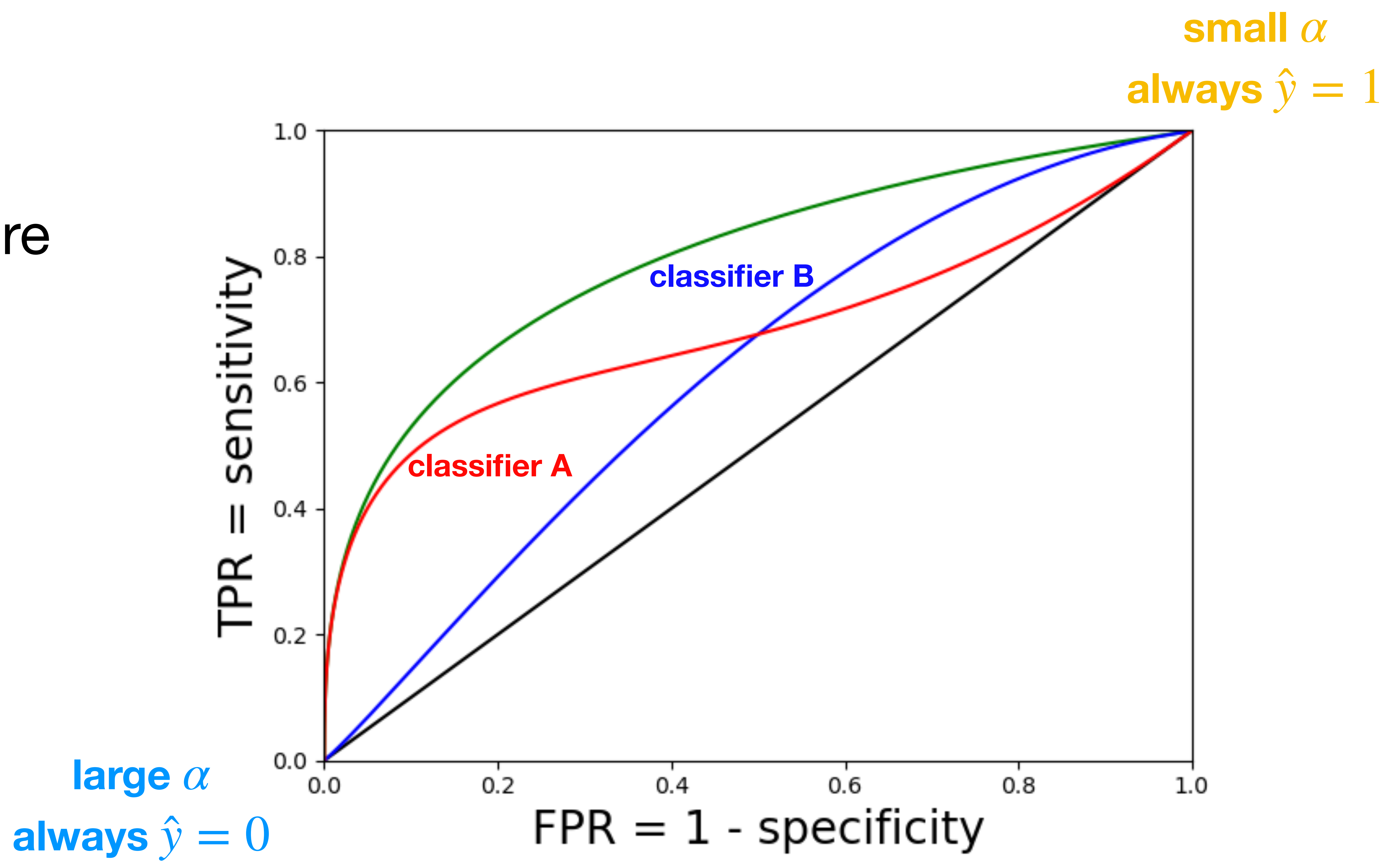


Demonstration

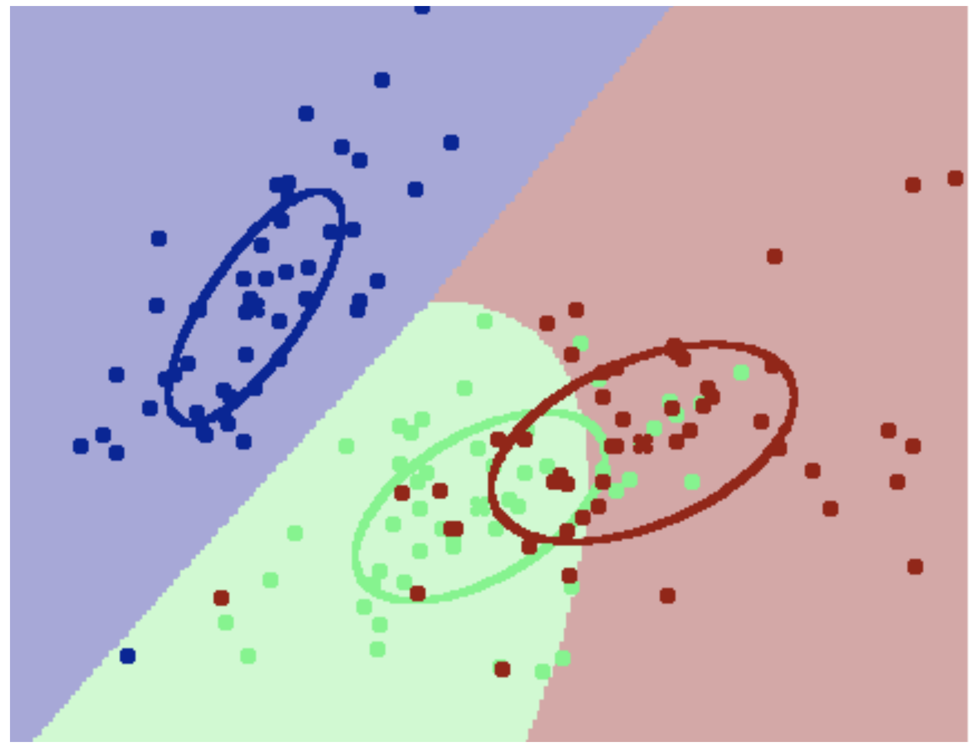
- http://www.navan.name/roc

Comparing classifiers

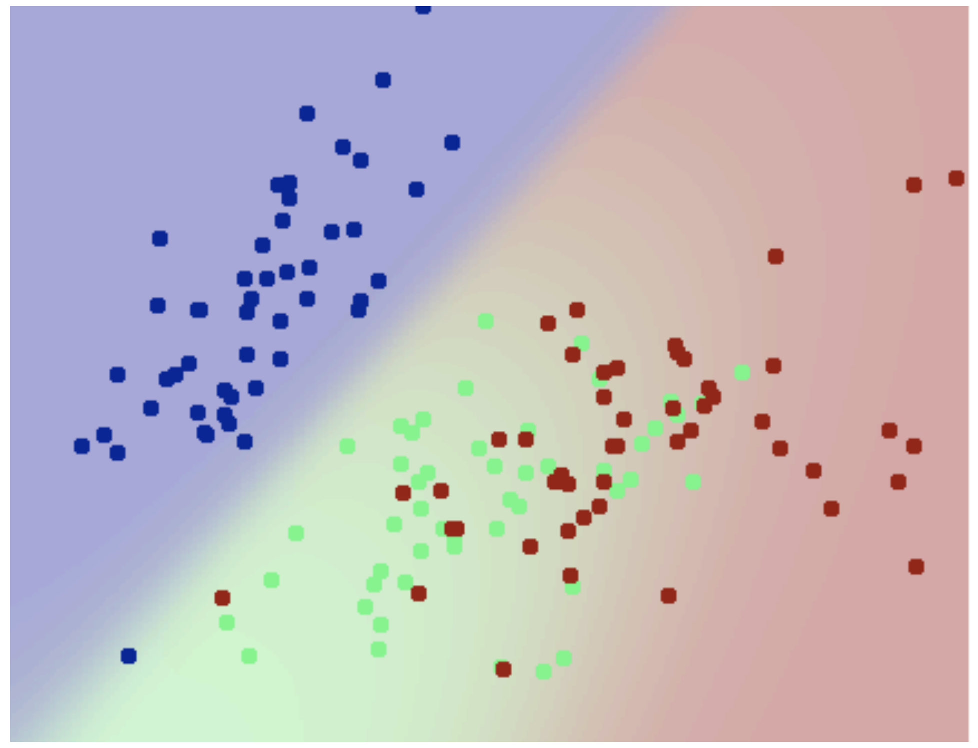
- Which classifier (A or B) performs "better"?
  - A is better for high specificity
  - B is better for high sensitivity
  - Need single performance measure
- Area Under Curve (AUC)
  - 0.5 ≤ AUC ≤ 1
  - AUC = 0.5: random guess
  - AUC = 1: no errors



Discriminative vs. probabilistic predictions



discriminative predictions $\hat{y}(x)$



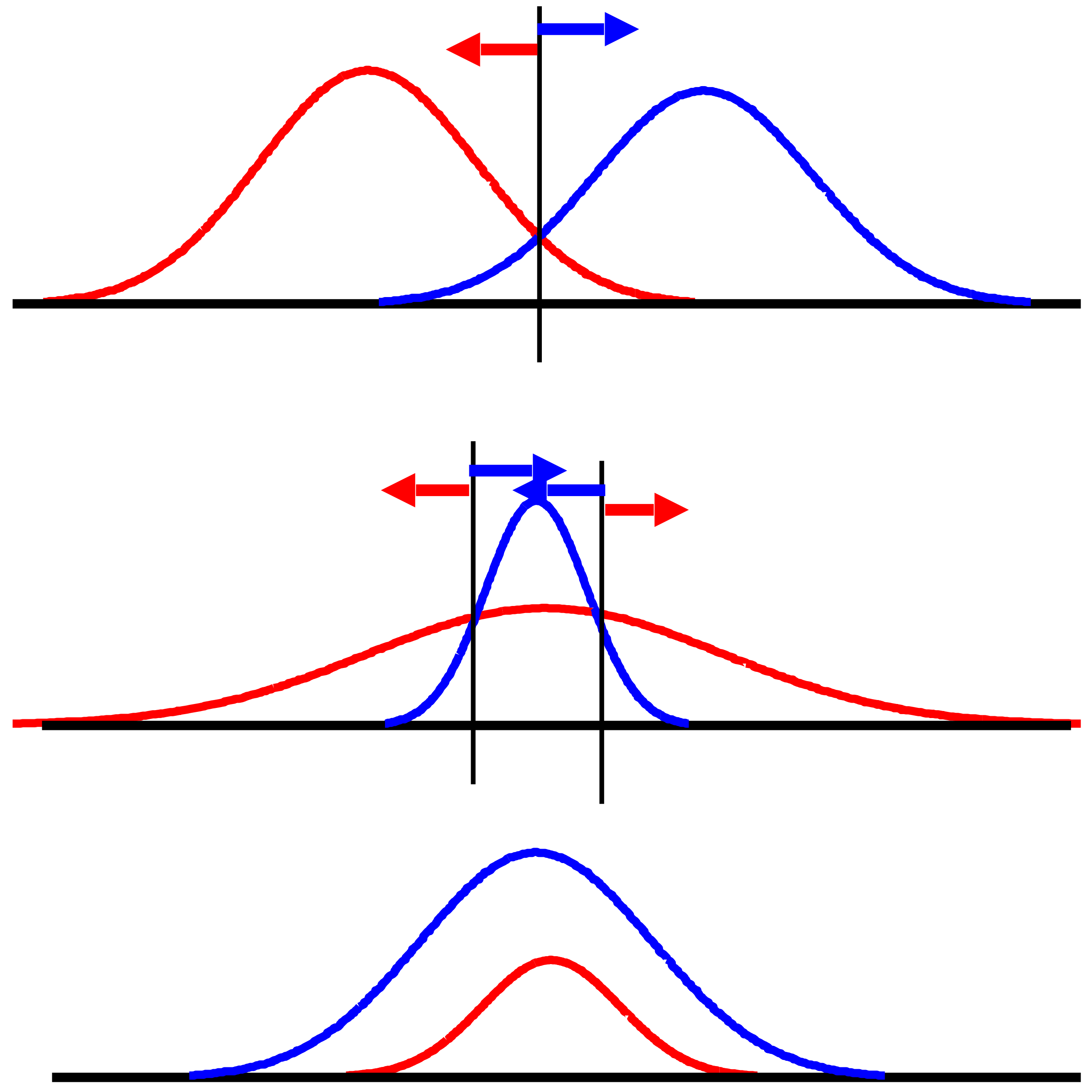
probabilistic predictions $p(y \mid x)$

```
>> learner = gaussianBayesClassify(X,Y)  % build a classifier
>> Ysoft = predictSoft(learner, X)       %  M x C matrix of confidences
>> plotSoftClassify2D(learner,X,Y)       %  shaded confidence plot
```

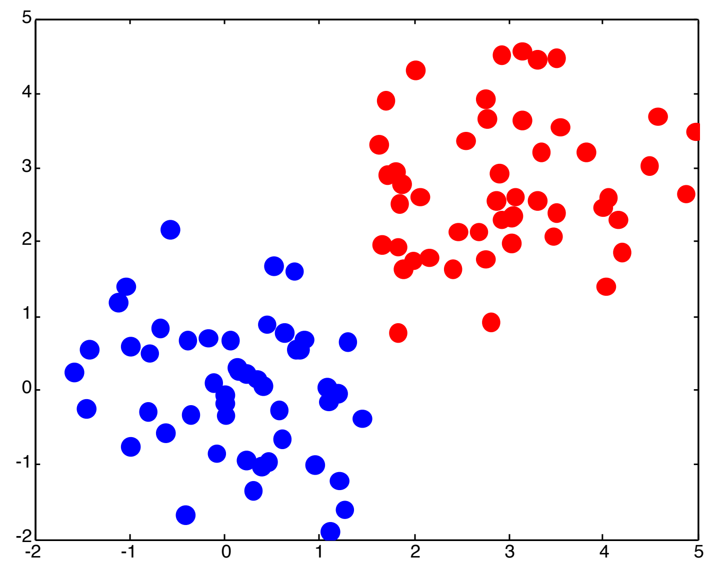
- Probabilistic learning gives more nuanced prediction
  - Can use $p(y \mid x)$ to find $\hat{y}(x) = \arg\max_y p(y \mid x)$ (if argmax is feasible)
  - Express confidence in predicting $\hat{y}$
  - Conditional models: $p(y \mid x)$; vs. generative models: $p(x, y)$
    - Can be used to generate x
    - Bayes classifiers, Naïve Bayes classifiers are generative

Gaussian models

- Bayes-optimal decision:
  - Scale each Gaussian by prior $p(y)$ and relative cost of error
  - Choose the larger scaled probability density
- Decision boundary = where scaled probabilities equal



Gaussian models



- Consider binary classifier with Gaussian conditionals
  - $p(x \mid y=c) = \mathcal{N}(x; \mu_c, \Sigma_c) = (2\pi)^{-\frac{d}{2}} |\Sigma_c|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu_c)^\intercal \Sigma_c^{-1} (x-\mu_c)\right)$
  - Assume same covariance $\Sigma_0 = \Sigma_1$
- What is the shape of the decision boundary $p(y=0 \mid x) = p(y=1 \mid x)$?

$$\alpha \lessgtr \log \frac{p(y=1)p(x \mid y=1)}{p(y=0)p(x \mid y=0)} = \frac{p(y=1)}{p(y=0)} + \text{const}$$

$$+\frac{1}{2}\left(x^\intercal \Sigma^{-1} x - 2\mu_0^\intercal \Sigma^{-1} x + \mu_0^\intercal \Sigma^{-1} \mu_0\right)$$

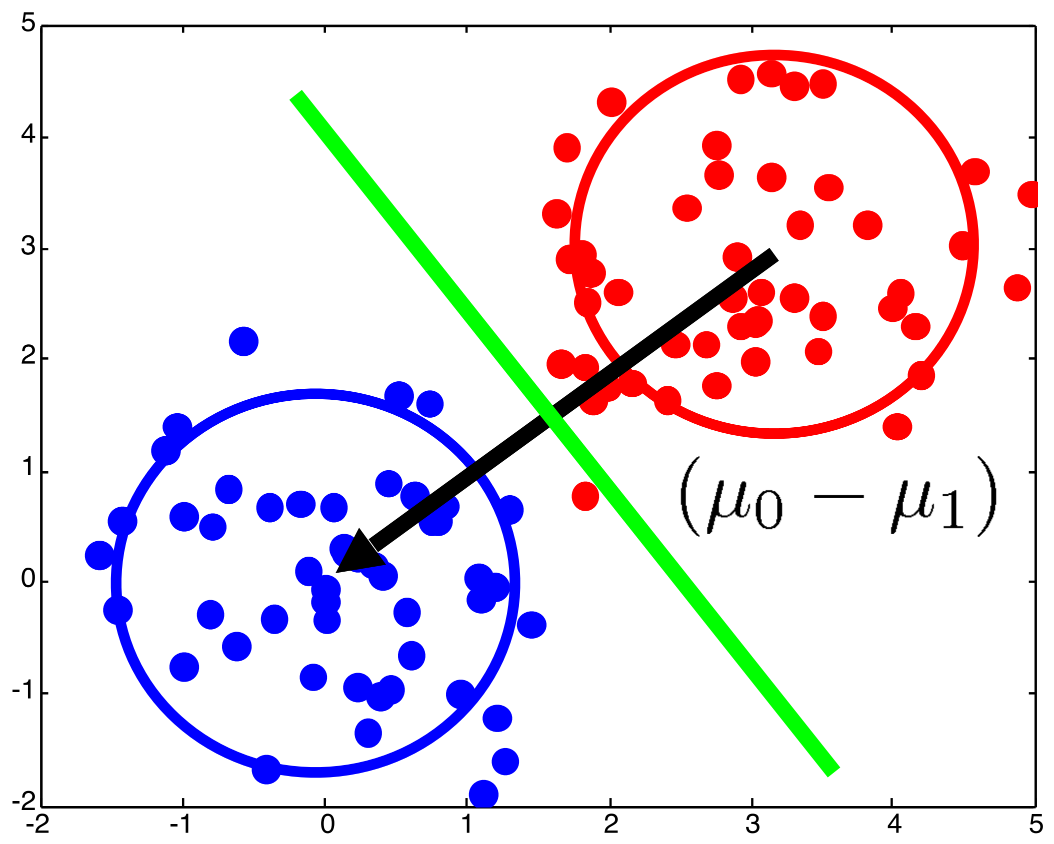
$$-\frac{1}{2}\left(x^\intercal \Sigma^{-1} x - 2\mu_1^\intercal \Sigma^{-1} x + \mu_1^\intercal \Sigma^{-1} \mu_1\right)$$

$$= \frac{1}{2}(\mu_1 - \mu_0)^\intercal \Sigma^{-1} x + \text{const}$$

linear!

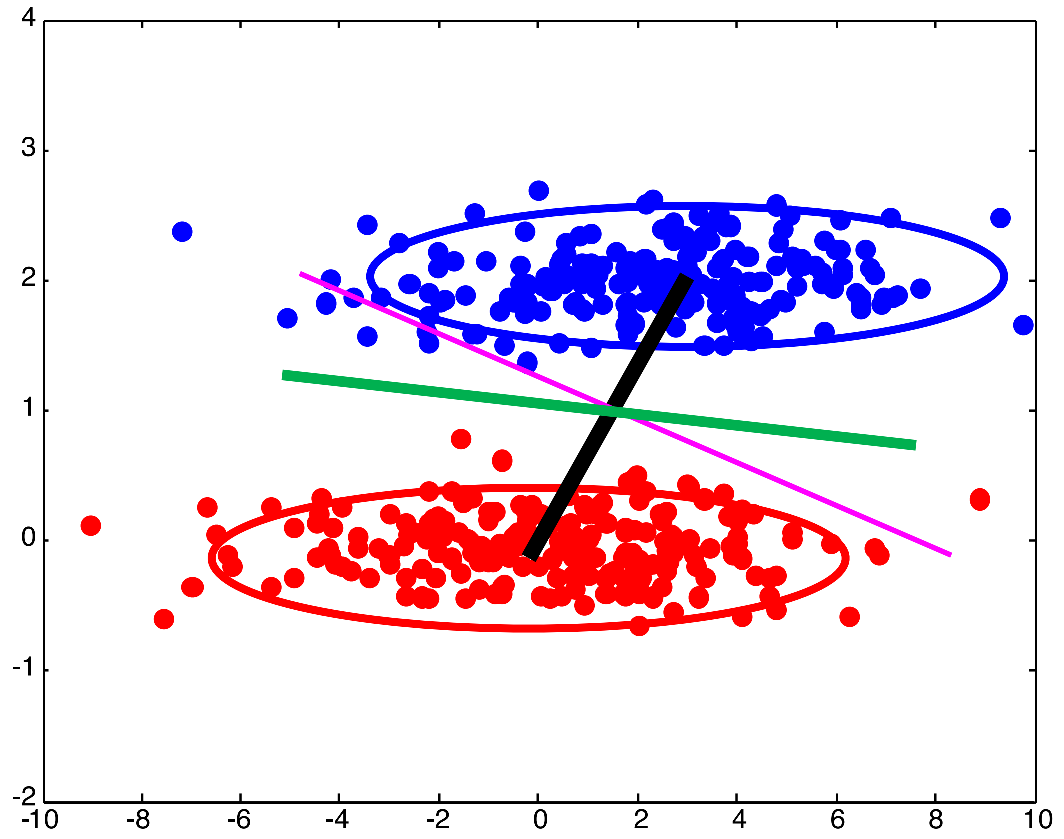
Gaussian models

- Isotropic covariance: $\Sigma = \sigma^2 I_d$
    - Decision: $(\mu_1 - \mu_0)^\intercal x \lessgtr \alpha$
    - Decision boundary perpendicular to segment between means



- General (but equal) covariance:
    - Decision boundary linear, but
        - scaled, if Σ has different eigenvalues
        - rotated, if Σ is not diagonal

$$\Sigma = \begin{bmatrix} 3 & 0 \\ 0 & .25 \end{bmatrix}$$



Today's lecture

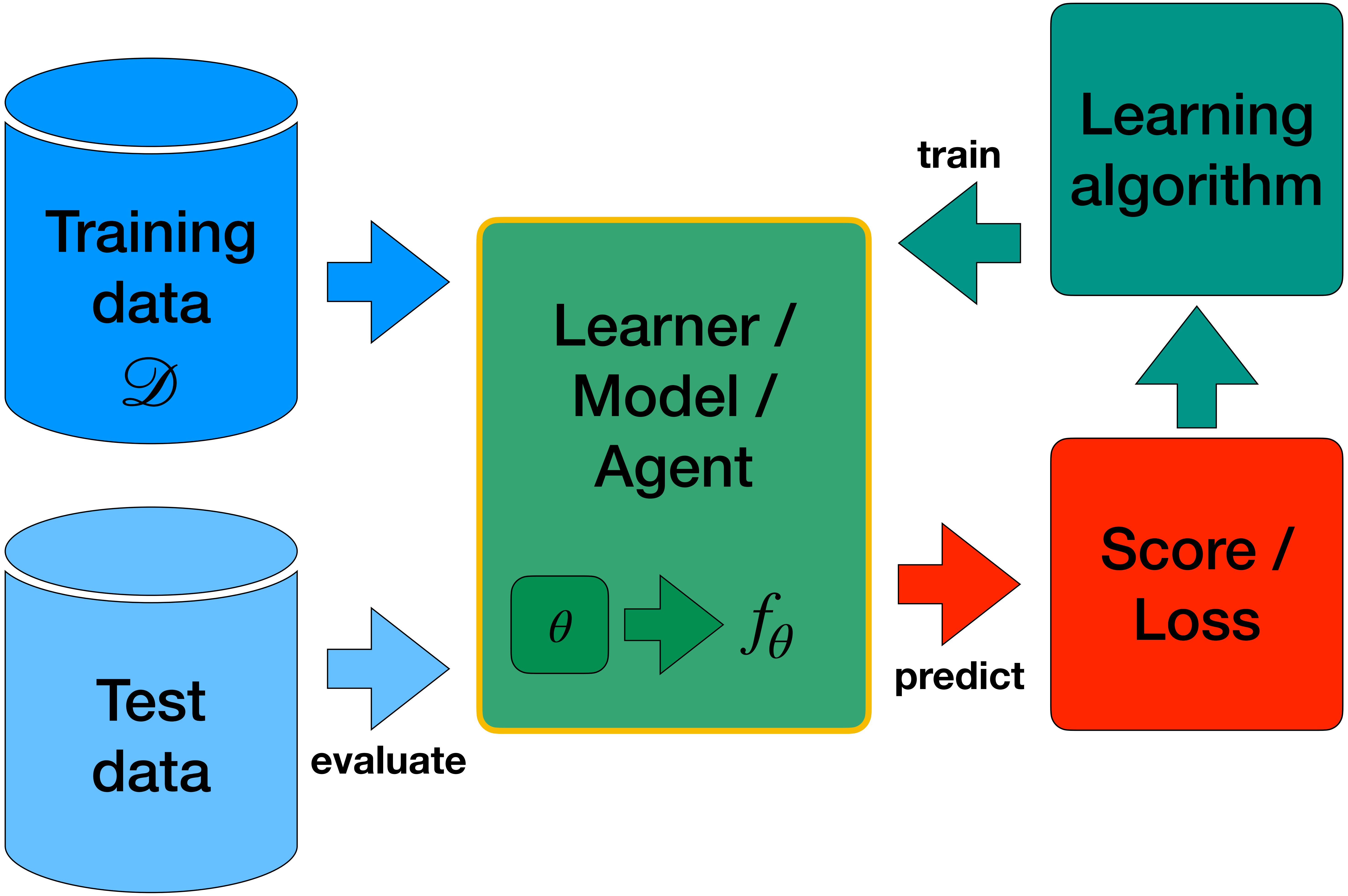
Naïve Bayes Classifiers

Bayes error

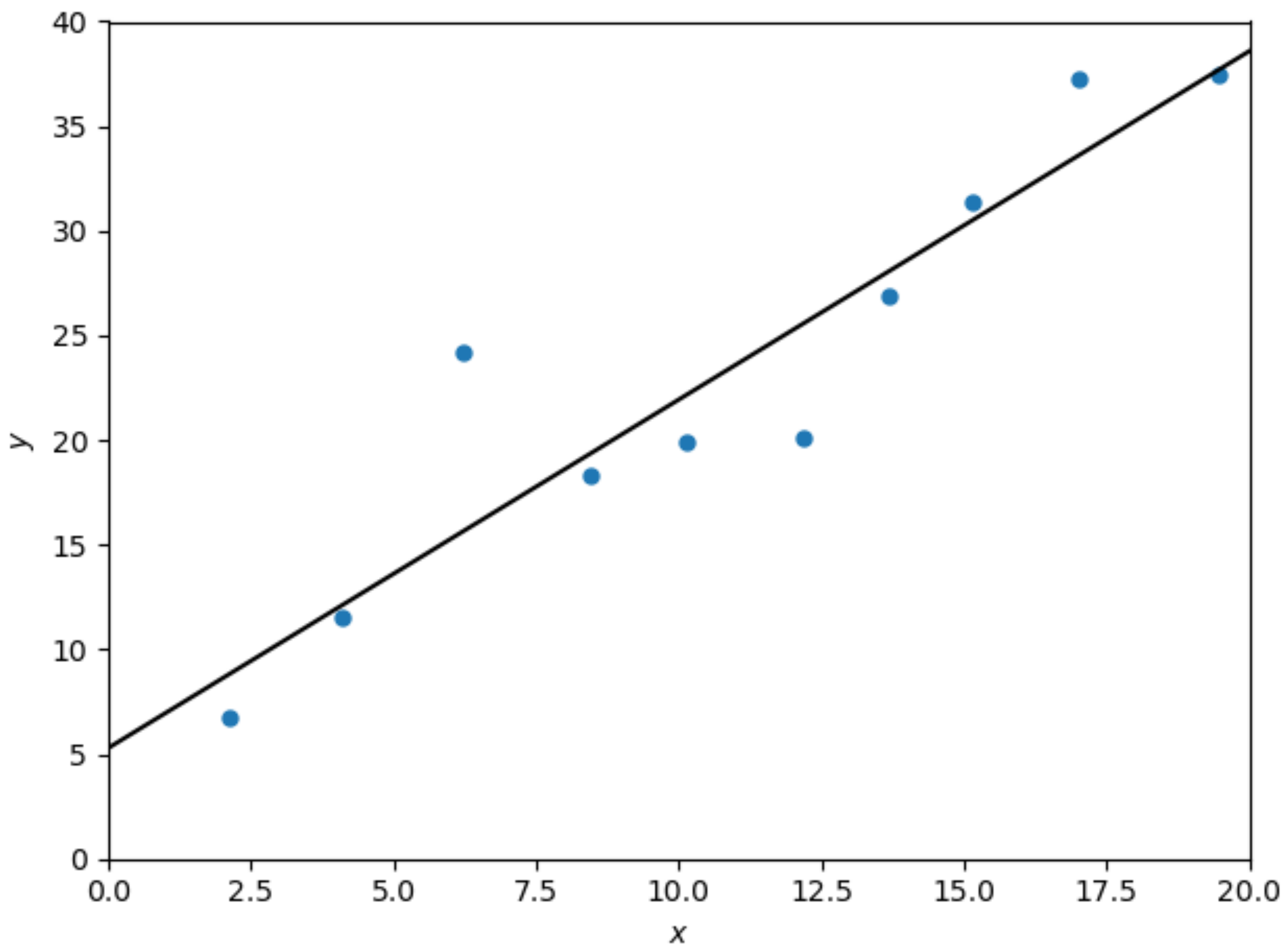
ROC curves

Linear regression

Machine learning



Linear regression



- Decision function $f : x \mapsto y$ is linear, $f(x) = \theta_0 + \theta_1 x$
- f is stored by its parameters $\theta = \begin{bmatrix} \theta_0 & \theta_1 \end{bmatrix}$

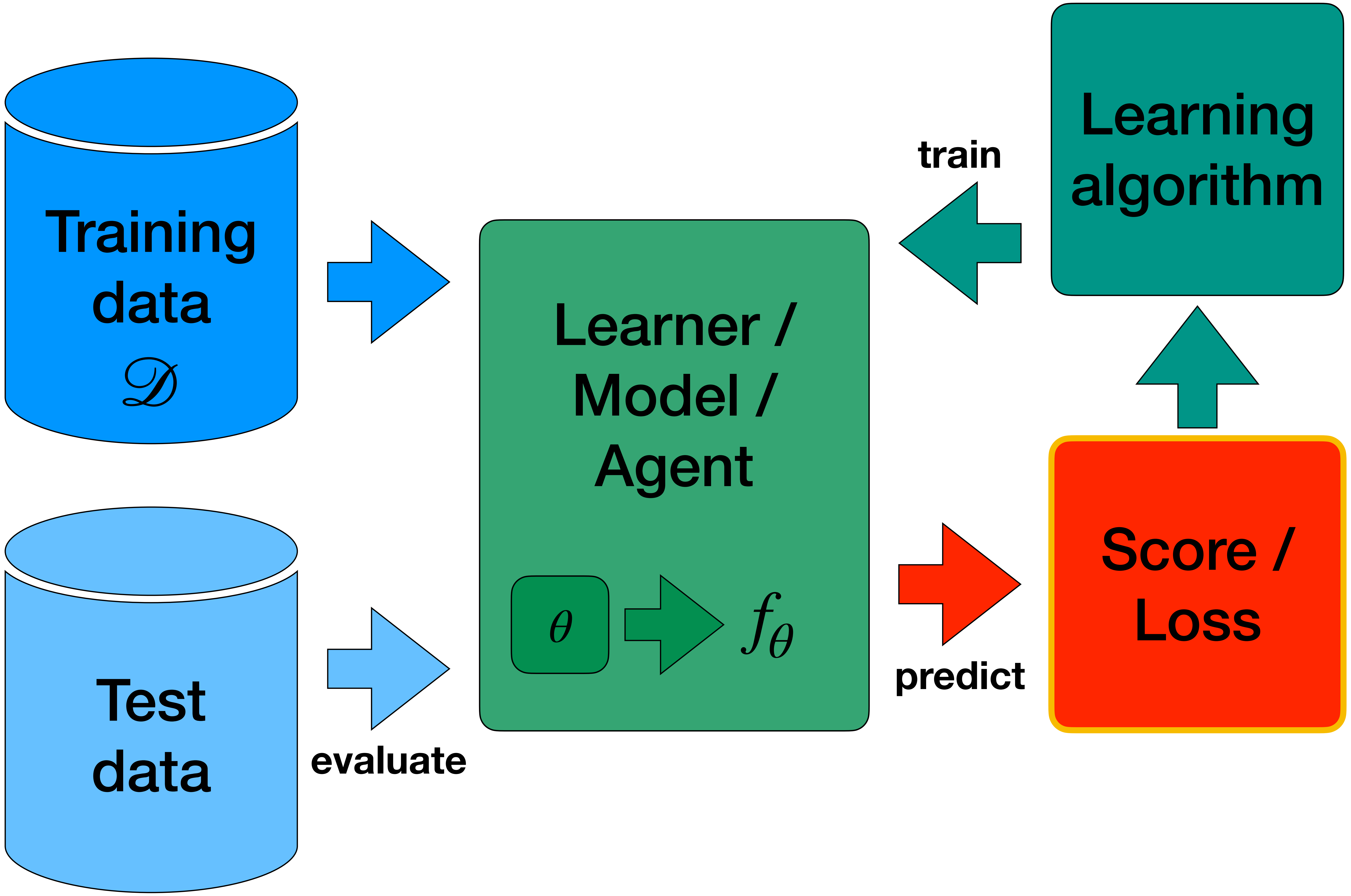
Linear regression

- More generally: $\hat{y}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \theta_n x_n$

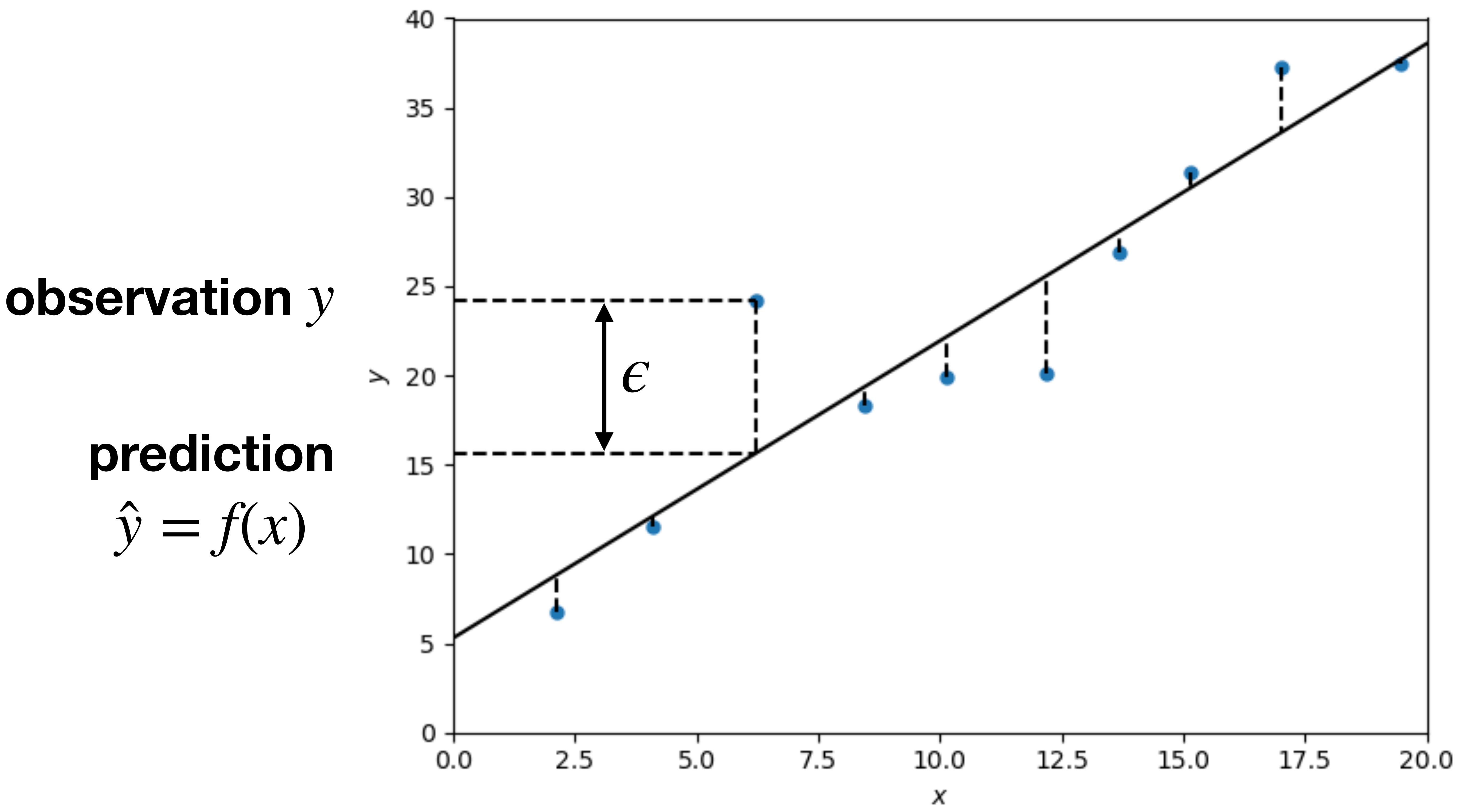
- Define dummy feature $x_0 = 1$ for the shift / bias θ_0

    - $\hat{y}(x) = \theta^\intercal x$; where $\qquad x = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$

Machine learning



Measuring error



- Error / residual: $\epsilon = y - \hat{y}$
- Mean square error (MSE): $\frac{1}{m}\sum_j (\epsilon^{(j)})^2 = \frac{1}{m}\sum_j (y^{(j)} - \hat{y}^{(j)})^2$

Mean square error

- $\mathscr{L}_\theta = \frac{1}{m}\sum_j (y^{(j)} - \hat{y}(x^{(j)}))^2 = \frac{1}{m}\sum_j (y^{(j)} - \theta^\intercal x^{(j)})^2$

- Why MSE?
  - Mathematically and computationally convenient (we'll see why)
  - Estimates the variance of the residuals
  - Corresponds to log-likelihood under Gaussian noise model

$$\log p(y \mid x) = \log \mathcal{N}(y; \theta^\intercal x, \sigma^2) = -\frac{1}{2\sigma^2}(y - \theta^\intercal x)^2 + \text{const}$$

MSE of training data

- Training data matrix: $X = \begin{bmatrix} x_0^{(1)} & \cdots & x_0^{(m)} \\ x_1^{(1)} & \cdots & x_1^{(m)} \\ \vdots & & \vdots \\ x_n^{(1)} & \cdots & x_n^{(m)} \end{bmatrix} \in \mathbb{R}^{(n+1)\times m}$

- Training labels vector: $y = \begin{bmatrix} y^{(1)} & \cdots & y^{(m)} \end{bmatrix}$

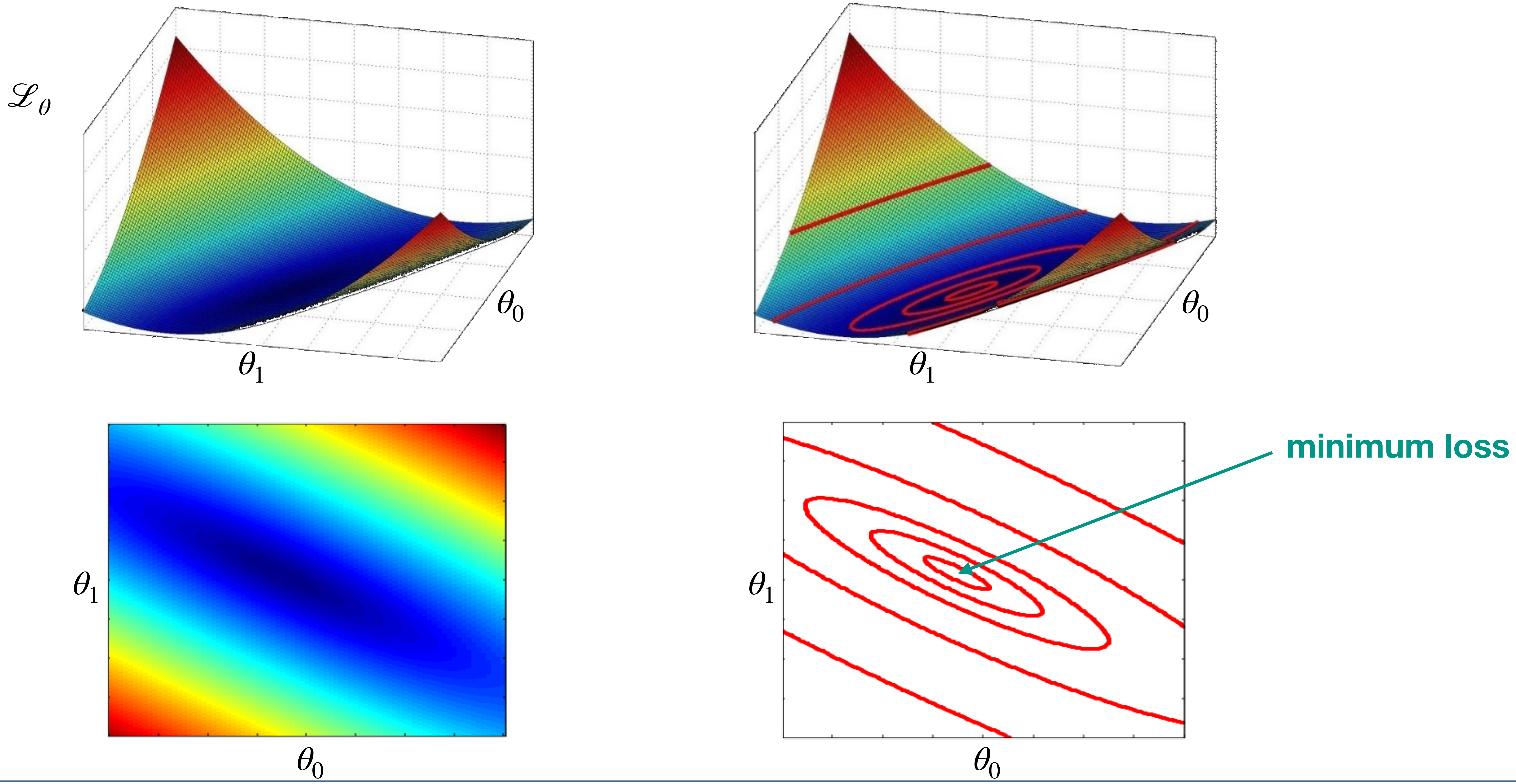
- Prediction: $\hat{y} = \begin{bmatrix} \hat{y}^{(1)} & \cdots & \hat{y}^{(m)} \end{bmatrix} = \theta^\intercal X$

```
# Python / NumPy:
e = y - theta.T @ X
loss = (e @ e.T) / m  # == np.mean( e ** 2 )
```

- Training MSE: $\mathscr{L}_\theta(\mathscr{D}) = \dfrac{1}{m} \sum_j (y^{(j)} - \theta^\intercal x^{(j)})^2 = \dfrac{1}{m}(y - \theta^\intercal X)(y - \theta^\intercal X)^\intercal$

Loss landscape

- $\mathscr{L}_\theta(\mathscr{D}) = \frac{1}{m}(y - \theta^\intercal X)(y - \theta^\intercal X)^\intercal = \frac{1}{m}(\theta^\intercal X X^\intercal \theta - 2yX^\intercal \theta + yy^\intercal)$ ← quadratic!



Logistics

assignments

- Assignment 1 due Thursday
- Assignment 2 to be published later this week