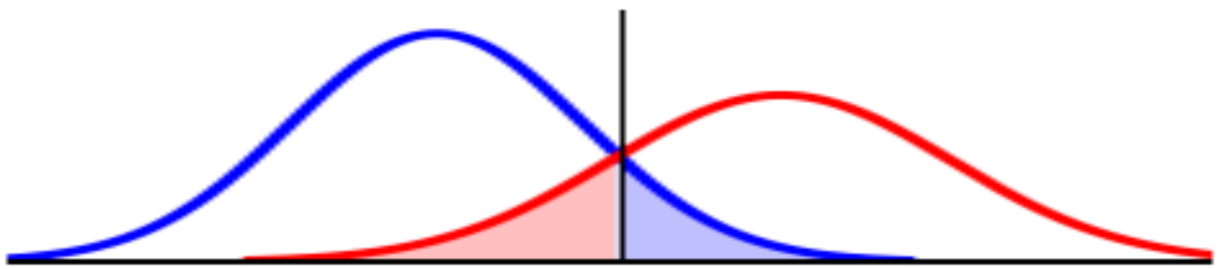
CS273A:
Machine Learning & Data Mining



Prof. Alexander Ihler

Fall 2024

- **Masters** Student Reception: Tues 10/8 @ 5:00-6:30pm (*RSVP!*)
- **PhD** Student Reception: Wed 10/9 @ 5:00-6:30pm (*RSVP!*)

Outline

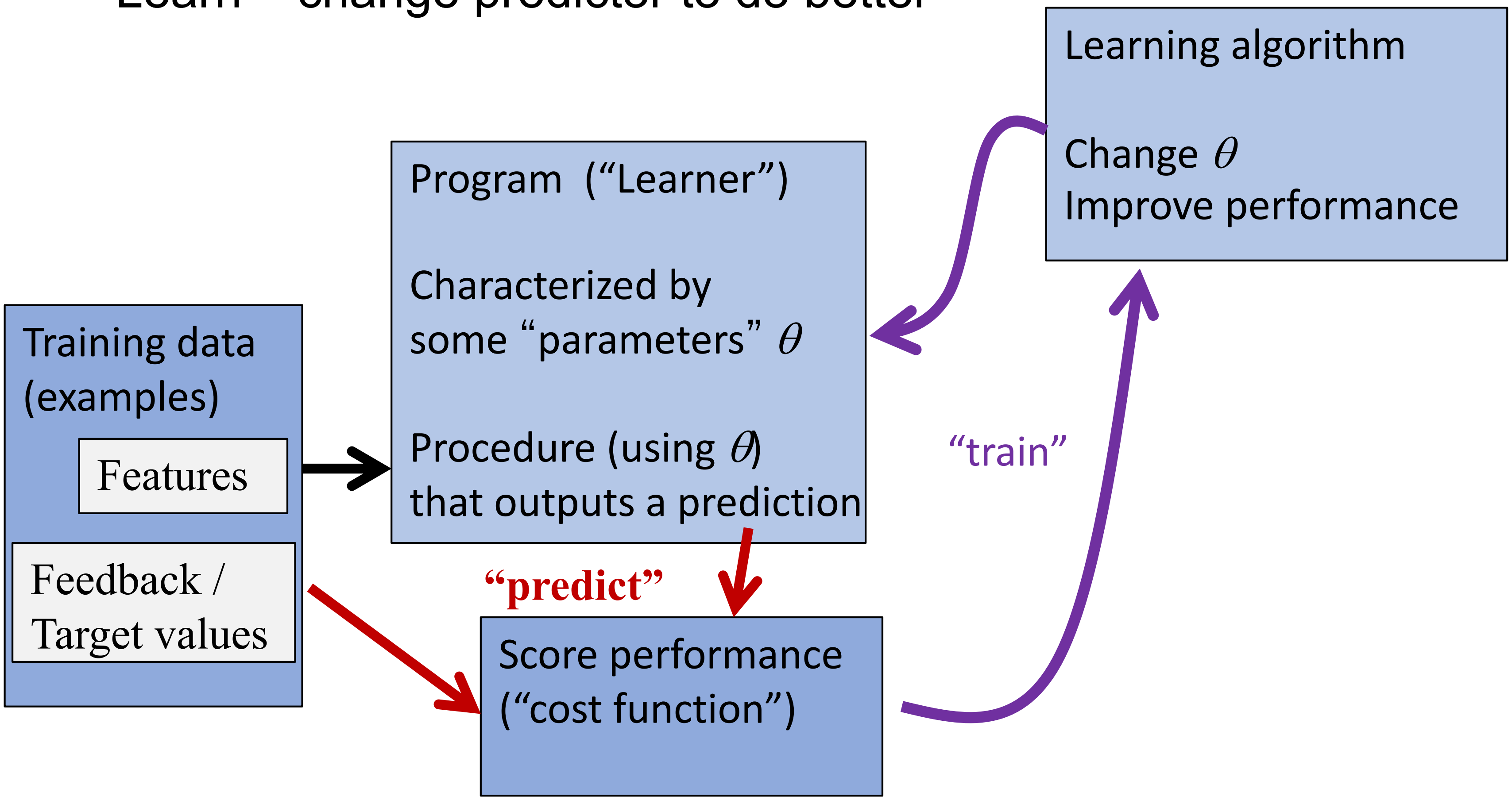
- How does ML work?
- Ex: Centroid Classifier
- Optimal Decisions (in theory)
- Bayes Classifiers
- Types of Errors

Outline

- How does ML work?
- Ex: Centroid Classifier
- Optimal Decisions (in theory)
- Bayes Classifiers
- Types of Errors

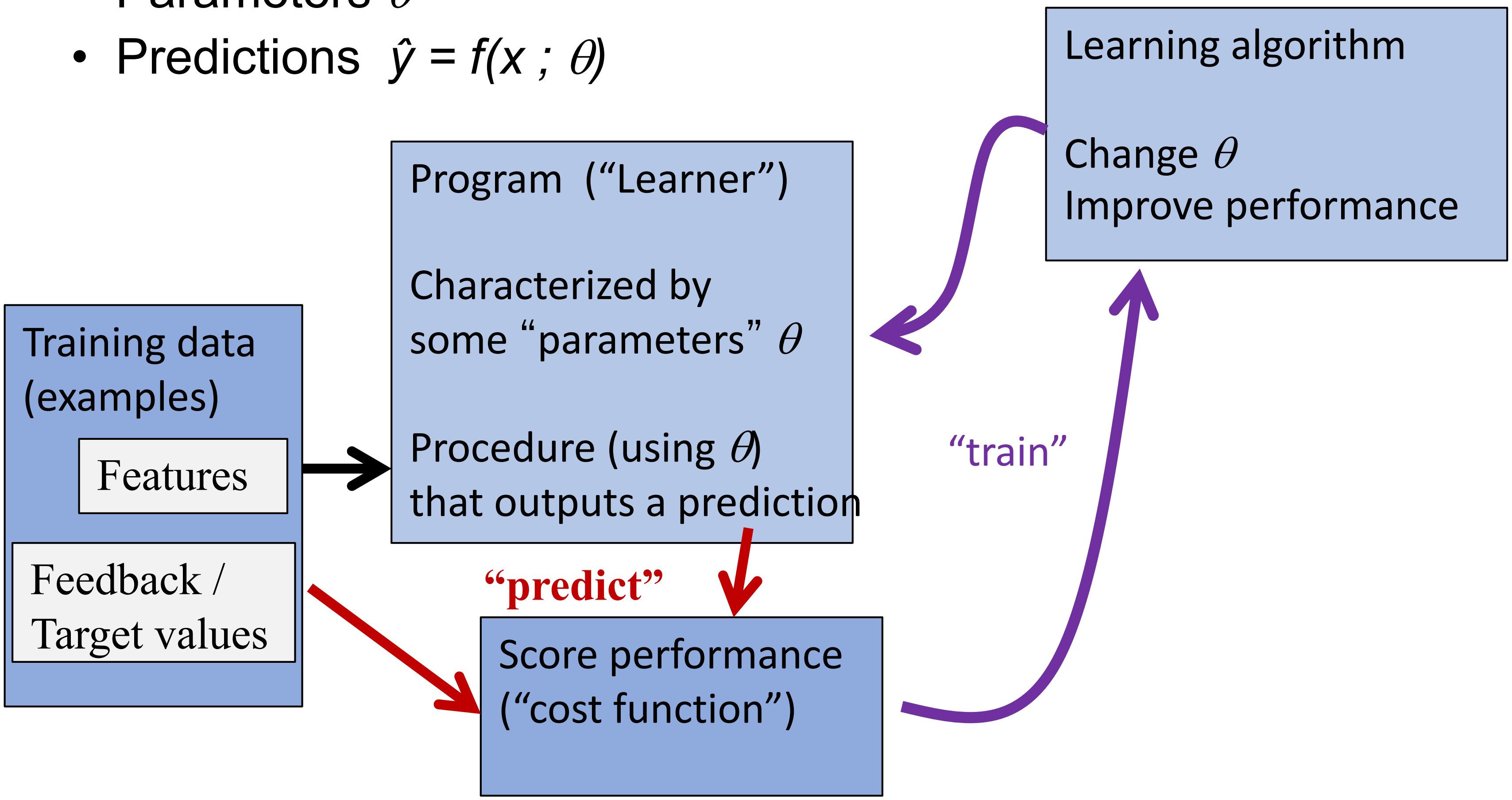
How does machine learning work?

- "Meta-programming"
  - Predict – apply rules to examples
  - Score – get feedback on performance
  - Learn – change predictor to do better

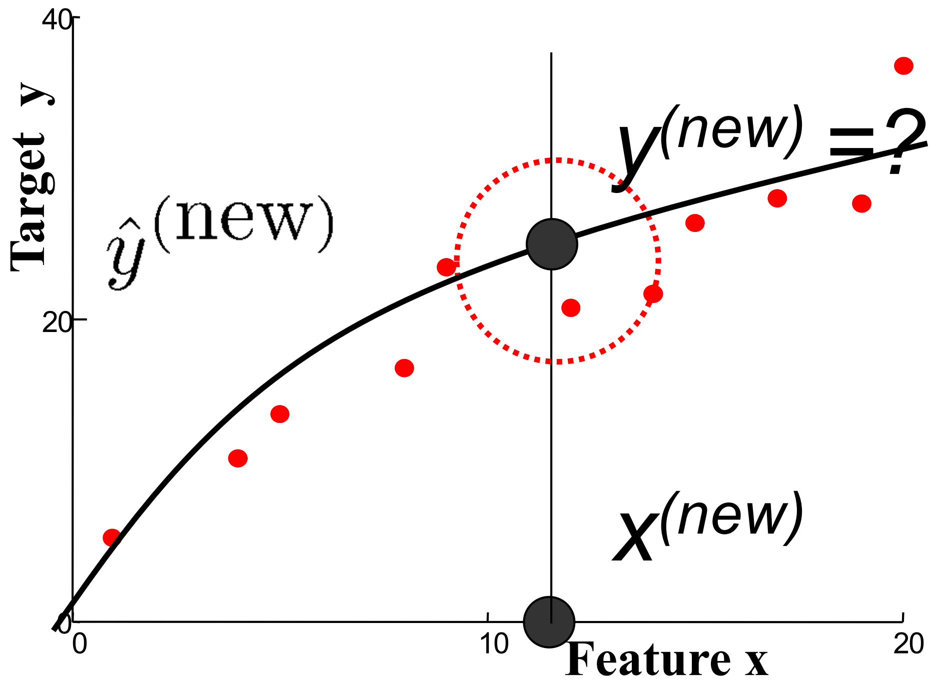


Supervised Learning

- Notation
  - Features x
  - Targets y
  - Parameters θ
  - Predictions $\hat{y} = f(x ; \theta)$

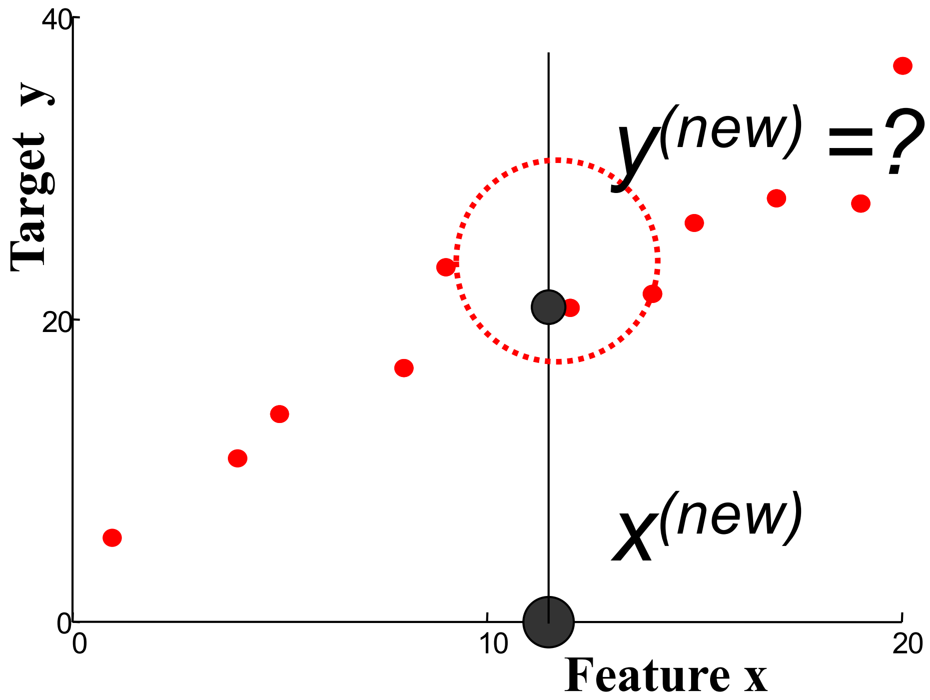


Regression: scatter plots



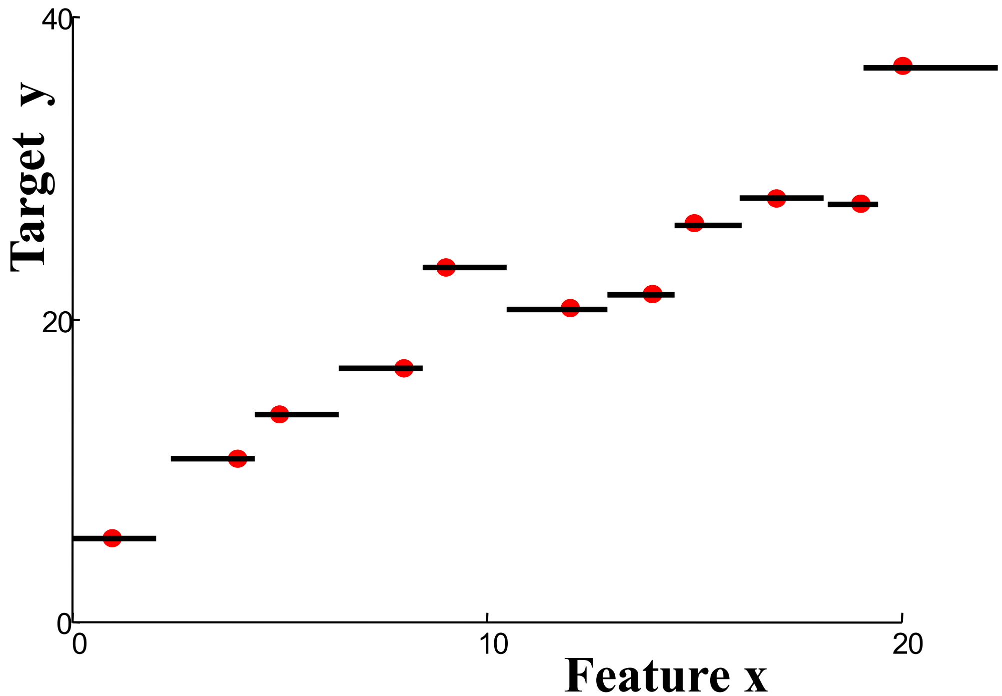
- Data suggest a relationship between x and y
- *Prediction*: new x, what is y?

Regression: nearest neighbor



- Find training datum $x^{(i)}$ closest to $x^{(new)}$; predict $y^{(i)}$

Regression: nearest neighbor



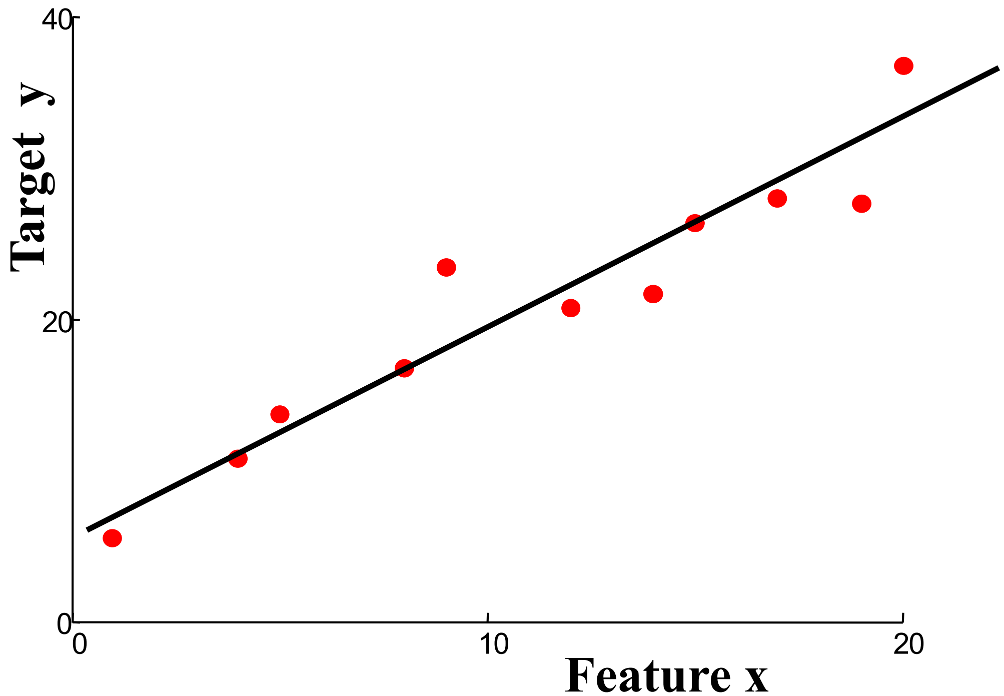
"Predictor":
Given new features:
Find nearest example
Return its value

Parameters? Saved examples

Train on data X? Just save X

- Defines a function f(x) implicitly
- "Form" is piecewise constant

Regression: linear regression



"Predictor":
Evaluate line:

$$r = \theta_0 + \theta_1 x_1$$

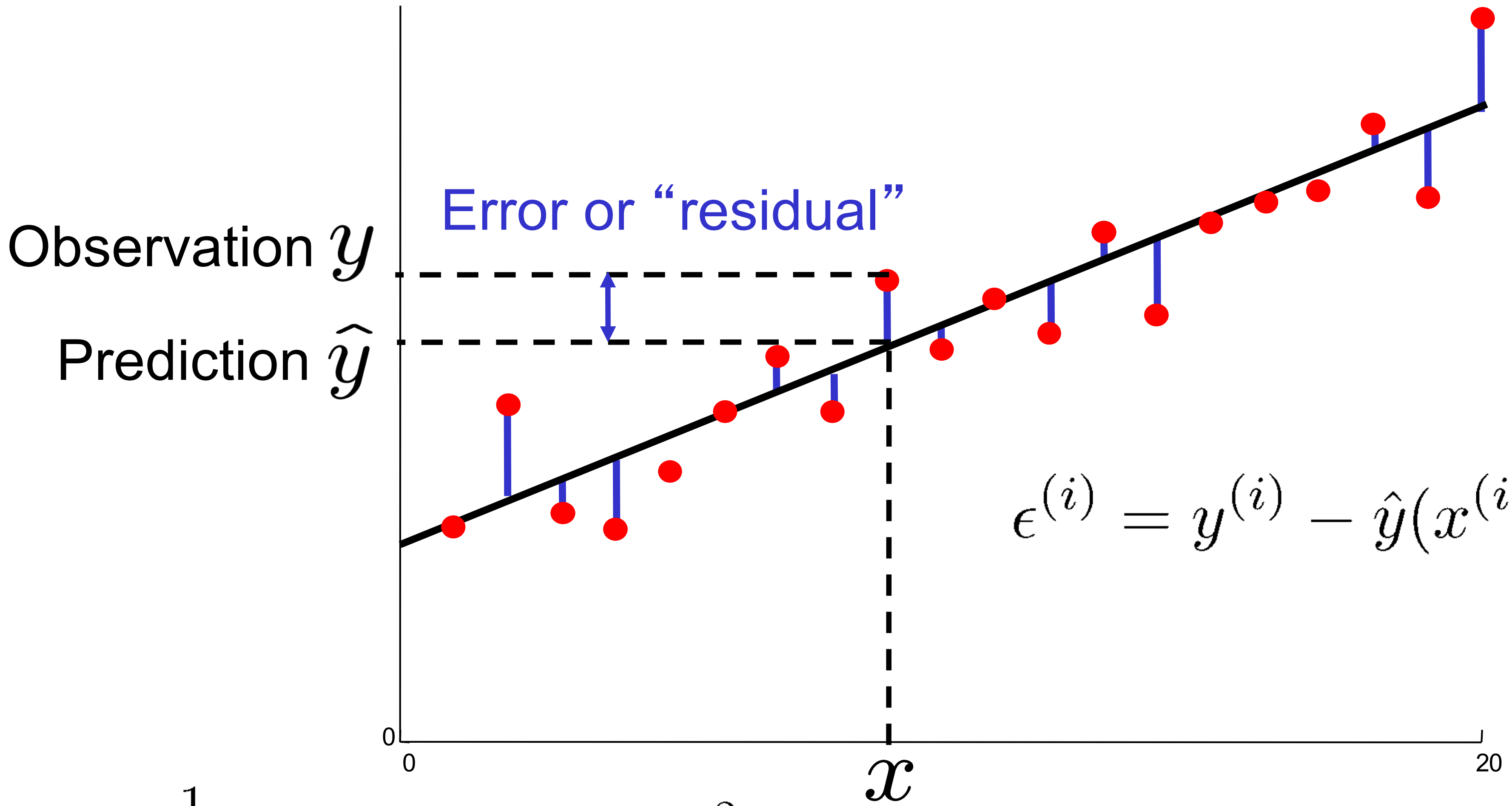
return r

Parameters? Slope, intercept

Train on data X? Find a "close" l

- Define form of function f(x) explicitly
- Find a good f(x) within that family

Measuring error

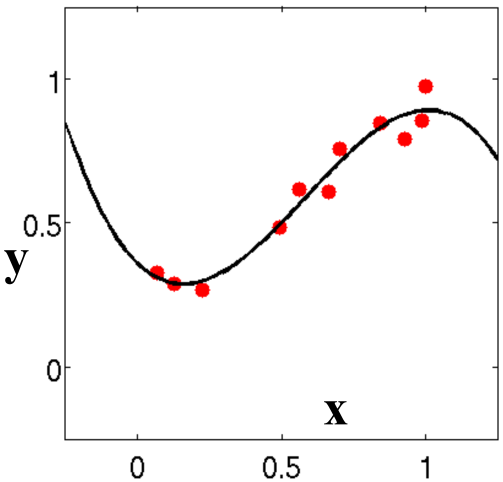


$$\epsilon^{(i)} = y^{(i)} - \hat{y}(x^{(i)})$$

$$\text{MSE} = \frac{1}{m}\sum_i \left(y^{(i)} - \hat{y}(x^{(i)}) \right)^2$$

Regression vs Classification

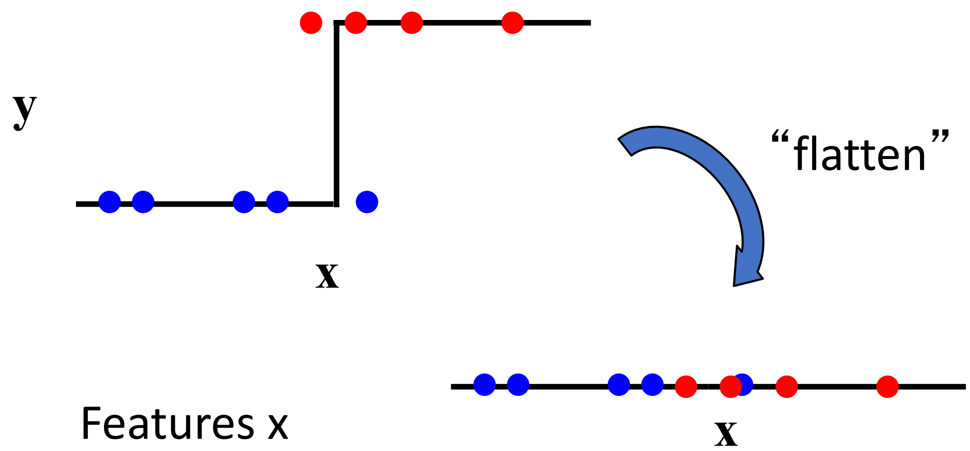
Regression



Features x
Real-valued target y

Predict continuous function ŷ(x)

Classification



Features x
Discrete class c
(usually 0/1 or +1/-1)
Predict discrete function ŷ(x)

Feature Vectors

$$x = (x_1, x_2, \ldots, x_n)$$

A component of the vector, corresponding to the value of "feature 2"

n = dimensionality of the vector

Example 1:

Feature vector for a medical patient: $\mathbf{x} = (\underset{\text{age}}{21.4},\ \underset{\text{height}}{6.1},\ \underset{\text{weight}}{200})$

Example 2:

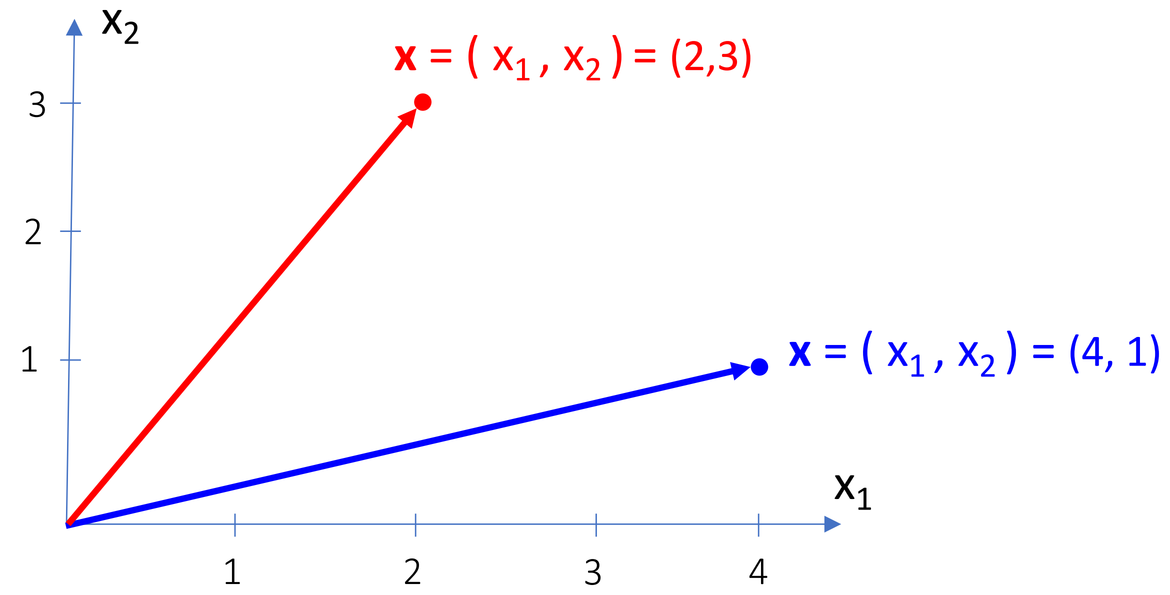
Feature vector for a loan applicant: $\mathbf{x} = (\underset{\text{age}}{21.4},\ \underset{\text{zipcode}}{92697},\ \underset{\text{income}}{65k},\ \underset{\text{debt}}{7.5k})$

Feature Vectors as “Data Points”

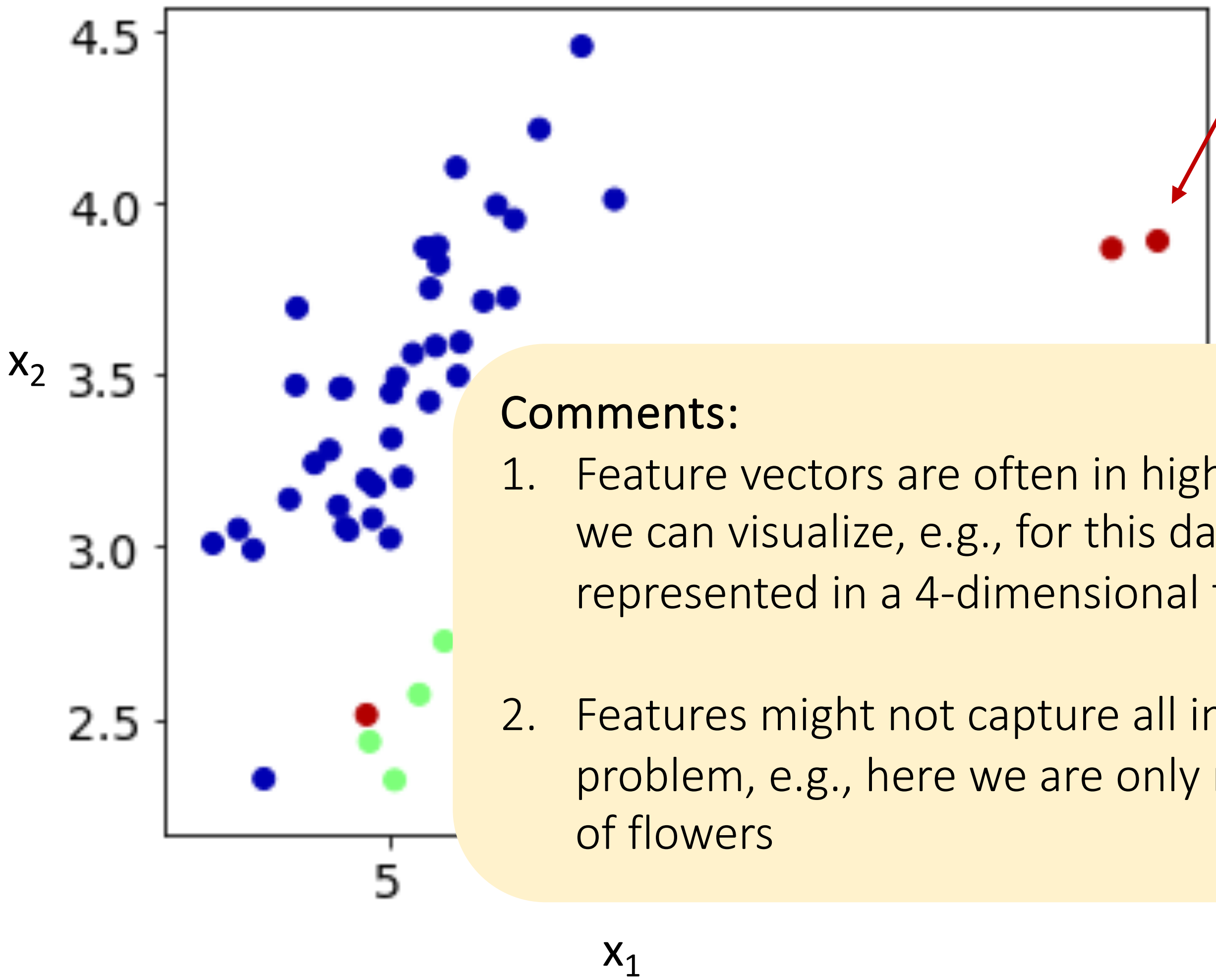
When we say “feature vector” we are referring to a point (in some d-dimensional space)

For example, if n = 2, we have $\mathbf{x} = (x_1, x_2)$

Here are two examples of vectors (red and blue)
representing two different datapoints in this 2-dimensional space



2-Dim Feature Space for Flowers



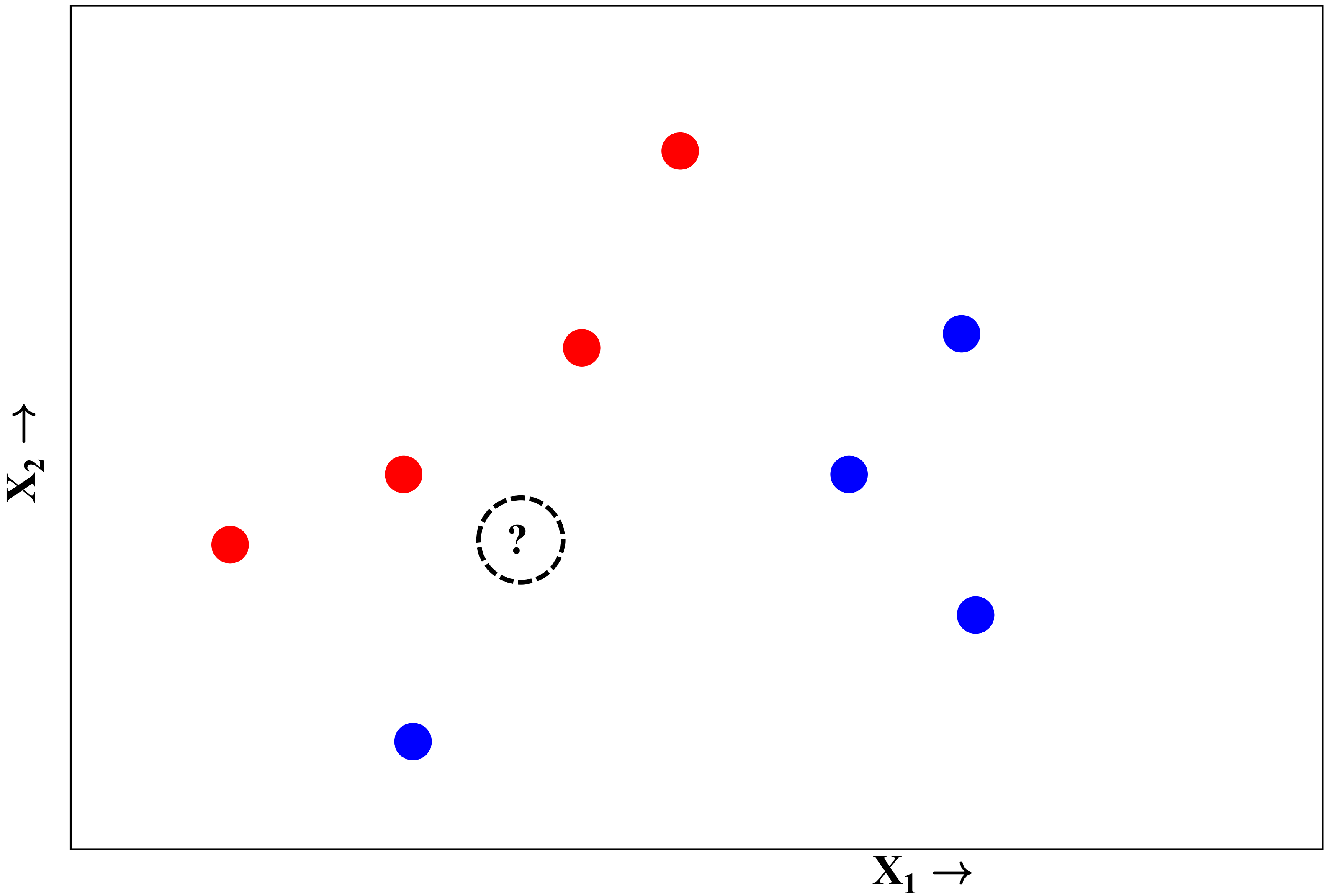
This feature vector, with coordinates (7.95, 3.89), represents a particular datapoint (a flower)

Comments:

1. Feature vectors are often in higher-dimensional spaces than we can visualize, e.g., for this dataset the flowers are represented in a 4-dimensional feature space

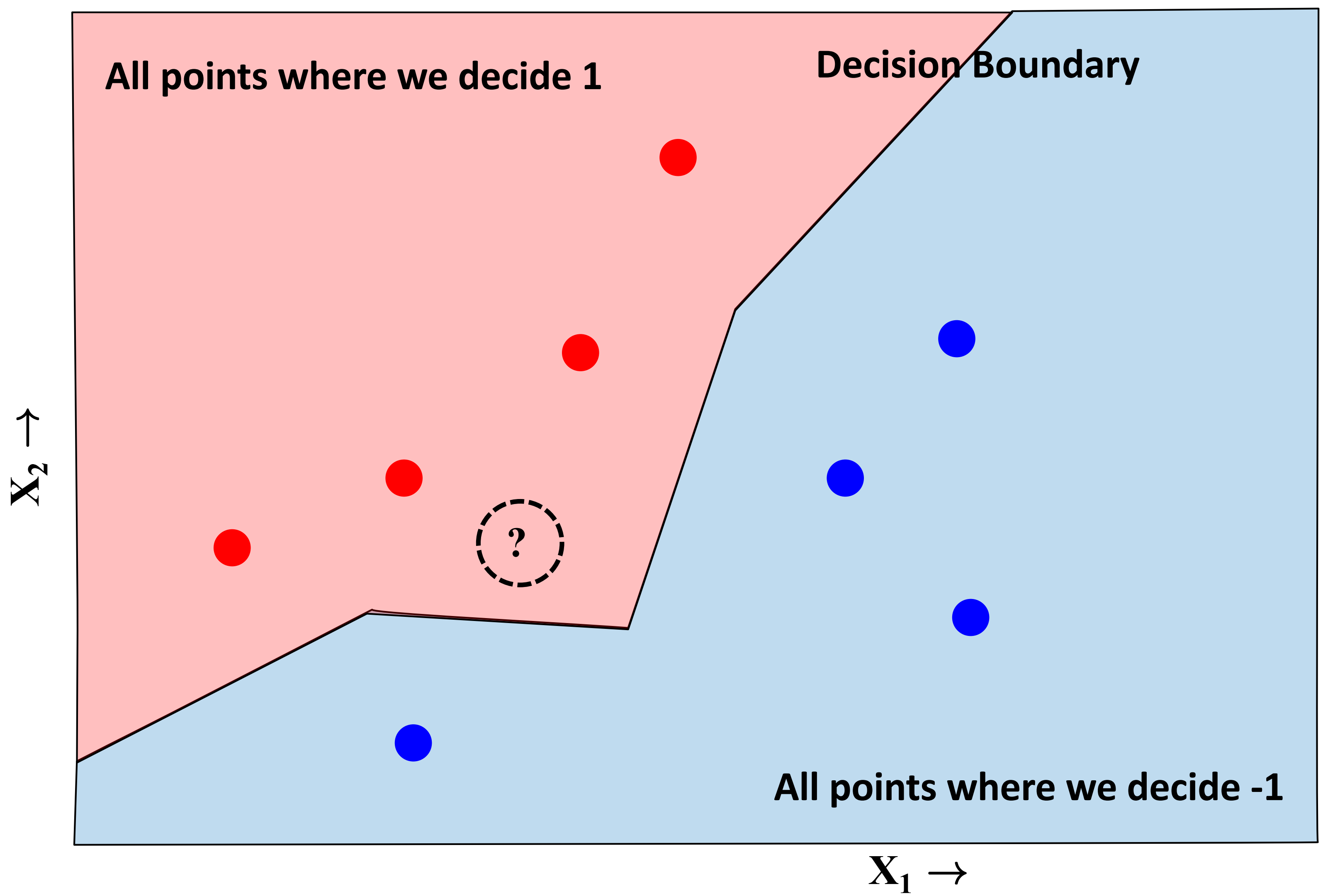
2. Features might not capture all important aspects of a problem, e.g., here we are only measuring certain properties of flowers

Classification



Classification

$$\text{ERR} = \frac{1}{m}\sum_i \left[\, y^{(i)} \neq \hat{y}(x^{(i)}) \,\right]$$



Outline

How does ML work?

Ex: Centroid Classifier

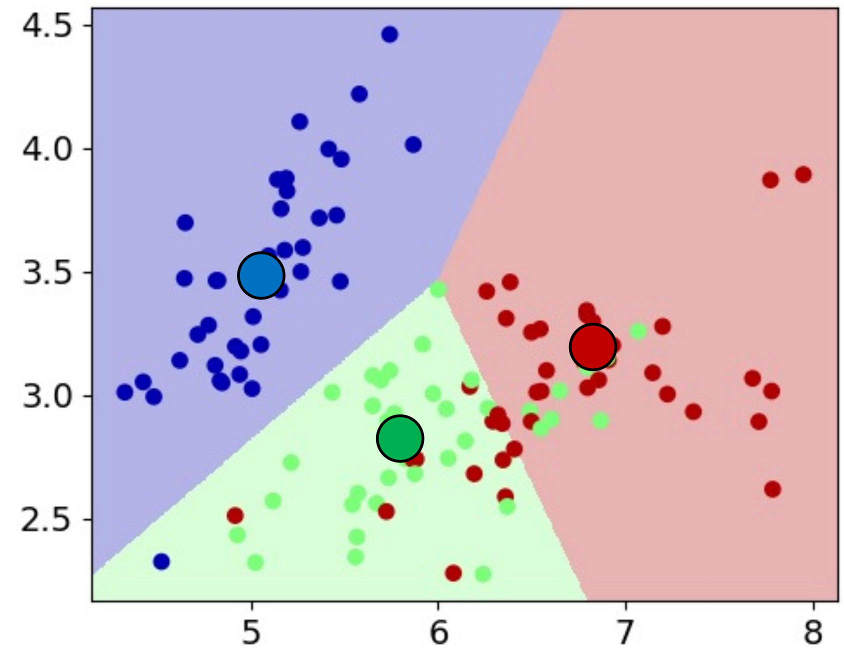
Optimal Decisions (in theory)

Bayes Classifiers

Types of Errors

Ex: Centroid Classifier

- A simple, classical predictor
  - Train: decide what a “typical example” of each class y looks like
    - Identify the possible classes
    - For each class: “typical example” = the centroid (average) of those examples
  - Predict: which does the test point x look most like?
    - “most like” = closest in Euclidean distance



Mean Values of Individual Features

Visually, to compute the mean value of feature 1
…. we sum all values in 1st col and divide by m

$$\underline{X} = \begin{bmatrix} \boxed{\begin{matrix} x_1^{(1)} \\ \vdots \\ x_1^{(m)} \end{matrix}} & \begin{matrix} \dots \\ \ddots \\ \dots \end{matrix} & \begin{matrix} x_n^{(1)} \\ \vdots \\ x_n^{(m)} \end{matrix} \end{bmatrix}$$

Mathematically,
mean value for feature j (e.g., j=2)

$$\boxed{\mu_j = \frac{1}{m} \sum_{i=1}^{m} x_j^{(i)}}$$

Sum over each row of matrix **X** (index i is for rows)

Only sum the elements of column j in matrix **X**

Means and Mean Vectors

Mean value for feature j

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)}$$

average over rows for column j (feature j) in X

Mean feature vector
(will also be referred to as the "centroid")

$$\mu = (\mu_1, \ldots, \mu_n)$$
$$= \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

average of the n vectors in X

Simple numerical example

$$X = \begin{pmatrix} x^{(1)} \\ x^{(2)} \end{pmatrix} = \begin{pmatrix} 21.4 & 6.1 & 200 \\ 28.1 & 5.5 & 145 \end{pmatrix}$$

Mean value for feature j = 1

$$\mu_1 = (21.4 + 28.1)/2 = 24.75$$

Mean feature vector (or centroid)

$$\mu = \begin{pmatrix} 24.75 & 5.8 & 172.5 \end{pmatrix}$$

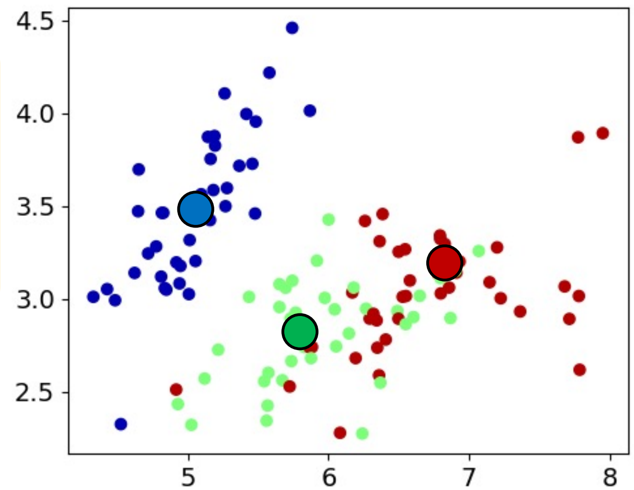
(in Python can use np.mean(X,axis=0),
where X is a 2 x 3 array)

Ex: Centroid Classifier

- A simple, classical predictor

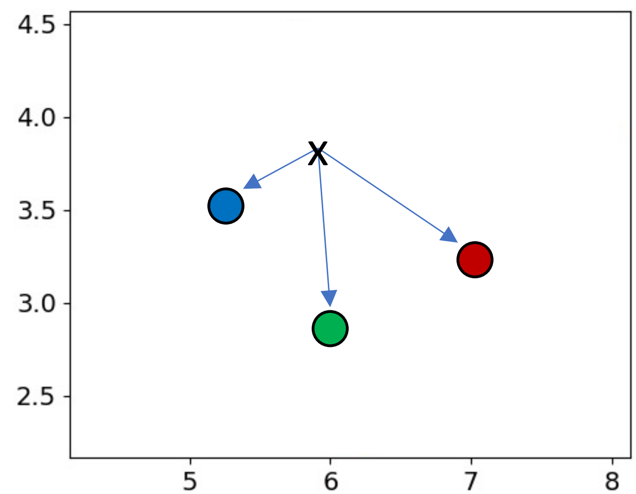
Training ("fit"):

```
train(X,y):
  for each possible class c:
    identify index of data points with y==c
    compute centroid (mean) mu_c of those data
```



Prediction:

```
predict(X):   # no known label y!
 for data point x:
   for each possible class c:
     find distance of x to mu_c
   pick the class c with smallest distance
```



Outline

- How does ML work?
- Ex: Centroid Classifier
- **Optimal Decisions (in theory)**
- Bayes Classifiers
- Types of Errors

A simple, optimal classifier

- Classifier f(x ; θ)
  - maps observations x to predicted target values
- Simple example
  - Discrete feature x: f(x ; θ) is a contingency table
  - Ex: spam filtering: observe just X_1 = sender in contact list?
- Suppose we knew the true conditional probabilities:
- Best prediction is the most likely target!

"Bayes error rate"

Pr[X=0] * Pr[wrong | X=0] + Pr[X=1] * Pr[wrong | X=1]

= Pr[X=0] * (1- Pr[Y=S | X=0]) + Pr[X=1] * (1-Pr[Y=K | X=1])

Can't do better than this without more information:
e.g., more features (email header, body text, etc.)

Feature	spam	keep
X=0	0.6	0.4
X=1	0.1	0.9

A simple classifier from data

- Training data $D=\{x^{(i)},y^{(i)}\}$, Classifier $f(x ; D)$
  - Discrete feature vector x
  - f(x ; D) is a contingency table
- Ex: Fisher Iris data, one feature
  - X_1 = sepal length (different ranges)
  - How should we make our predictions?
  - One method: just estimate the probabilities?

Sepal length	Iris setosa	Iris virsicolor	Iris virginica
X < 5	21	30	5
5 < X < 6	23	21	30
6 < X < 7	0	16	35
7 < X	0	1	10

$\Rightarrow$

Sepal length	Iris setosa	Iris virsicolor	Iris virginica
X < 5	0.375	0.536	0.089
5 < X < 6	0.311	0.284	0.405
6 < X < 7	0.	0.314	0.686
7 < X	0.	0.091	0.909

(empirically estimated)

Estimating p(y|X=x): "probabilistic" learning
Gives a prediction *and* an (estimated) notion of confidence in that prediction

A simple classifier from data

- Training data D={$x^{(i)}$,$y^{(i)}$}, Classifier f(x ; D)
  - Discrete feature vector x
  - f(x ; D) is a contingency table
- Ex: Fisher Iris data, one feature
  - What if we bin the data more finely?
  - Find data falling within each range:

Two sources of error!

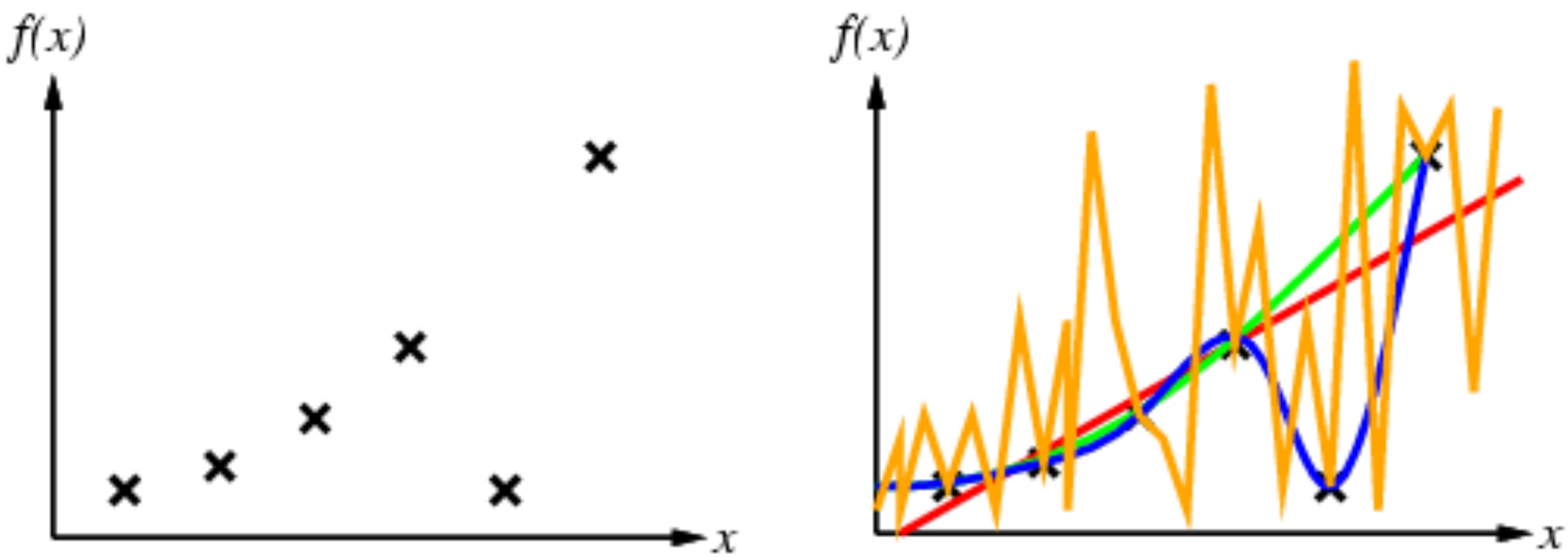
- Bayes error rate (improve with more info in X)
- Mis-estimating probability (improve with more data)

Sepal length	Iris setosa	Iris virsicolor	Iris virginica
…			
5.25	0.57	0.07	0.36
5.5	0.09	0.48	0.43
5.75	0.08	0.38	0.54
…			

Sepal length	Iris setosa	Iris virsicolor	Iris virginica
…			
5.48	1	0	0
5.5	0	0	1
5.52	0	0	0
…			

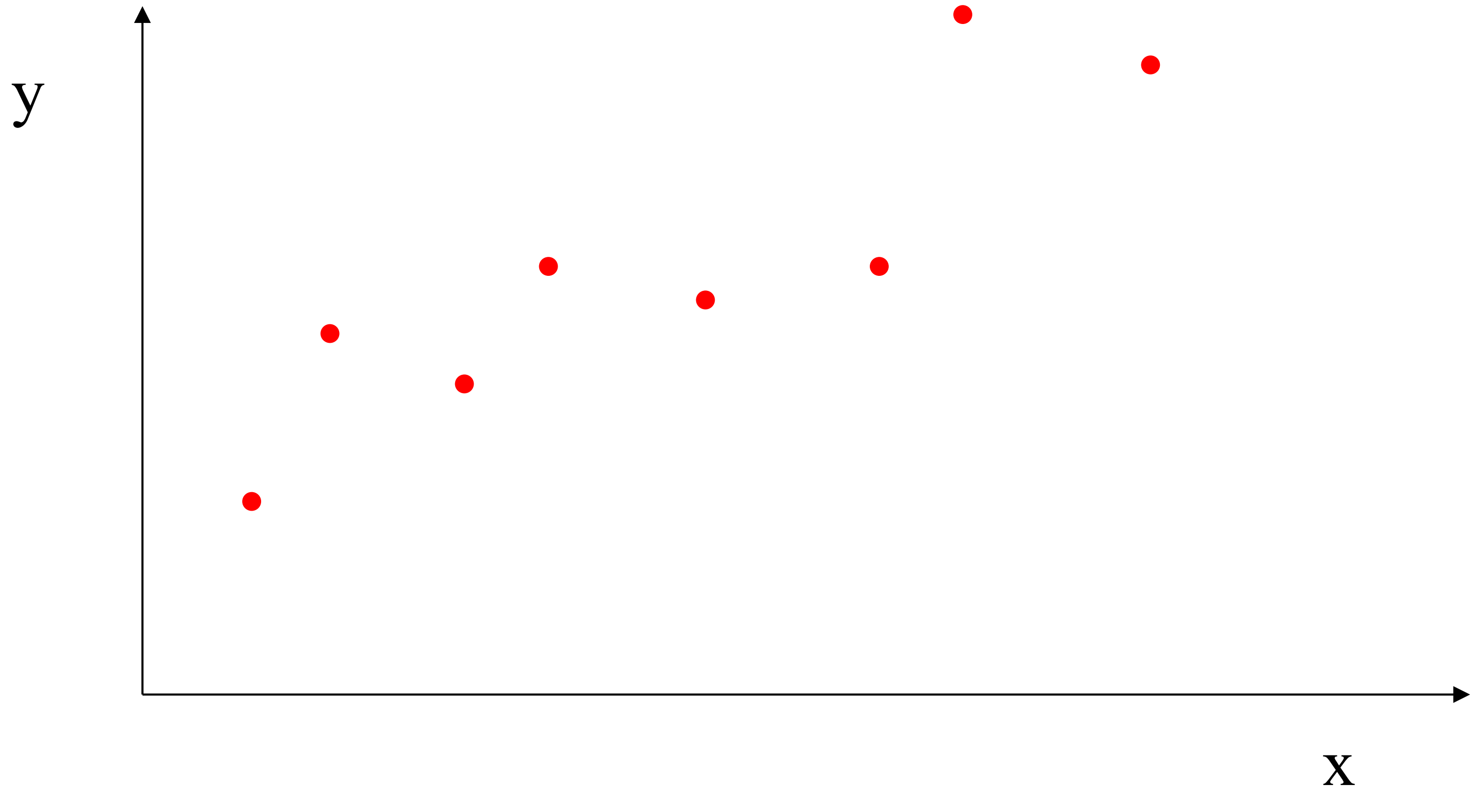
Inductive bias

- Allow us to extend observed data to unobserved ones
  - Interpolation / extrapolation
- What relationships do we expect in the data?
  - A (perhaps *the*) key question in ML models
  - Usually, data pull us away from assumptions only with evidence!

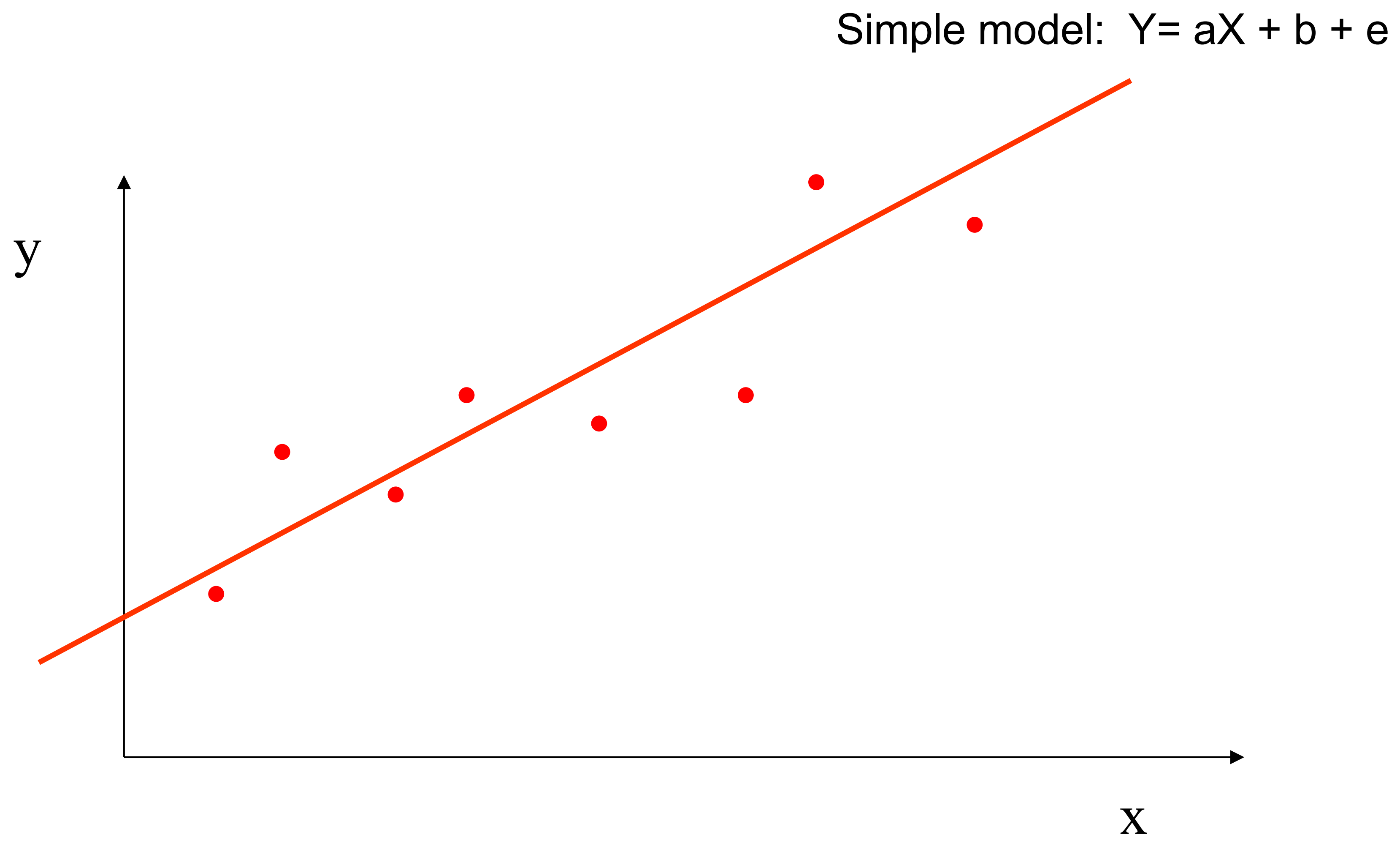


All of these explain the data in some way!

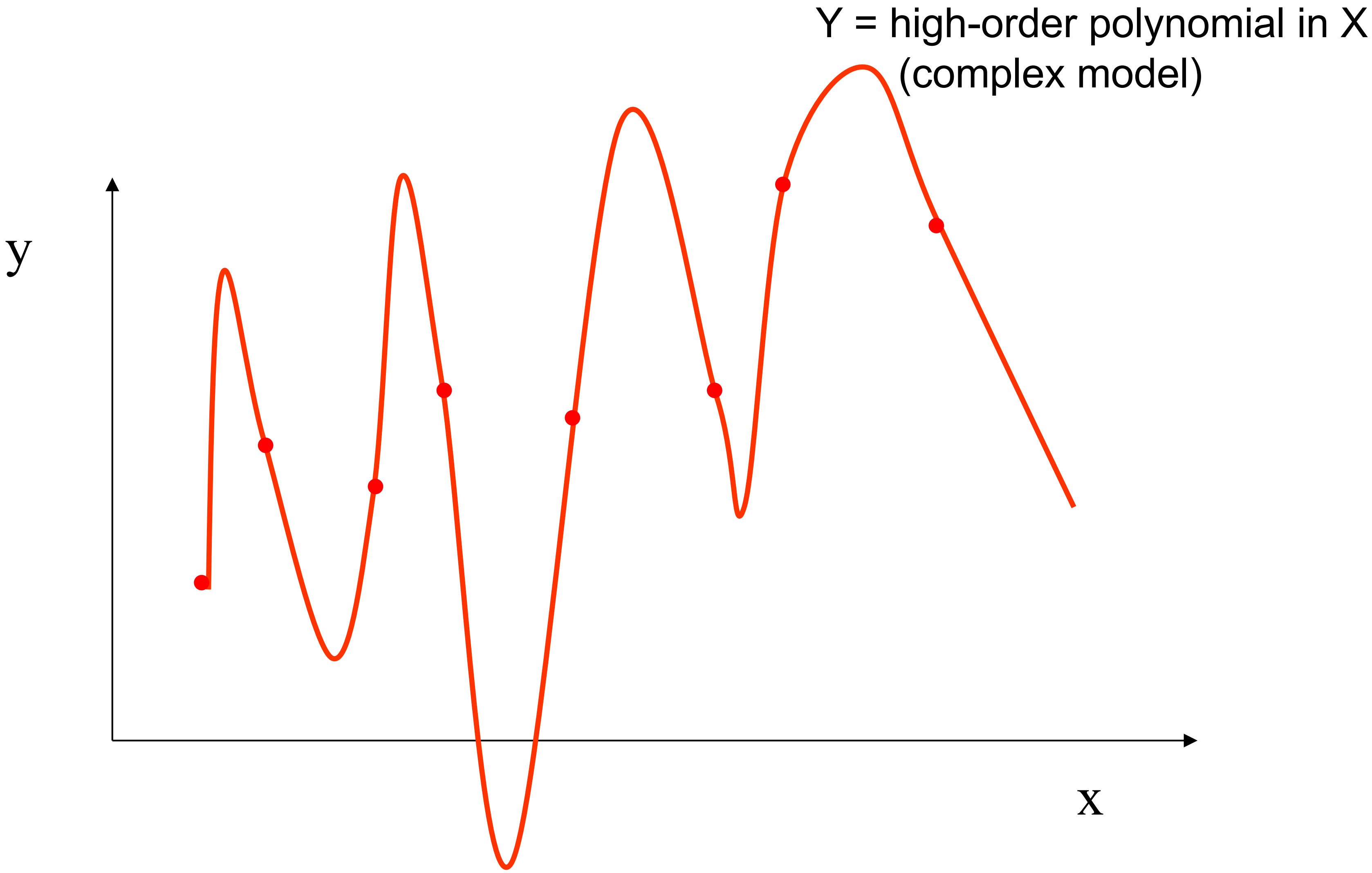
Overfitting & Complexity



Overfitting & Complexity

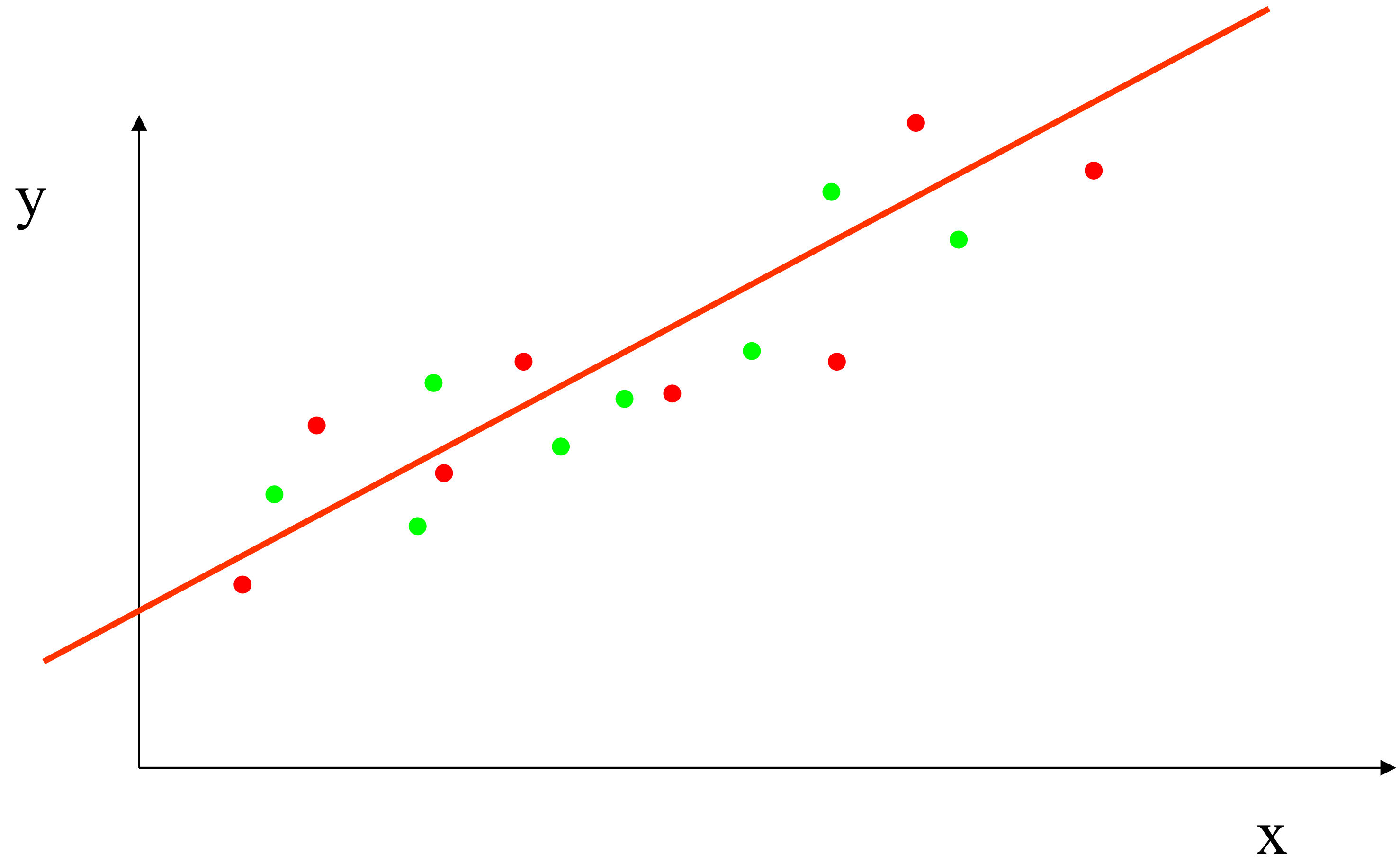


Overfitting & Complexity

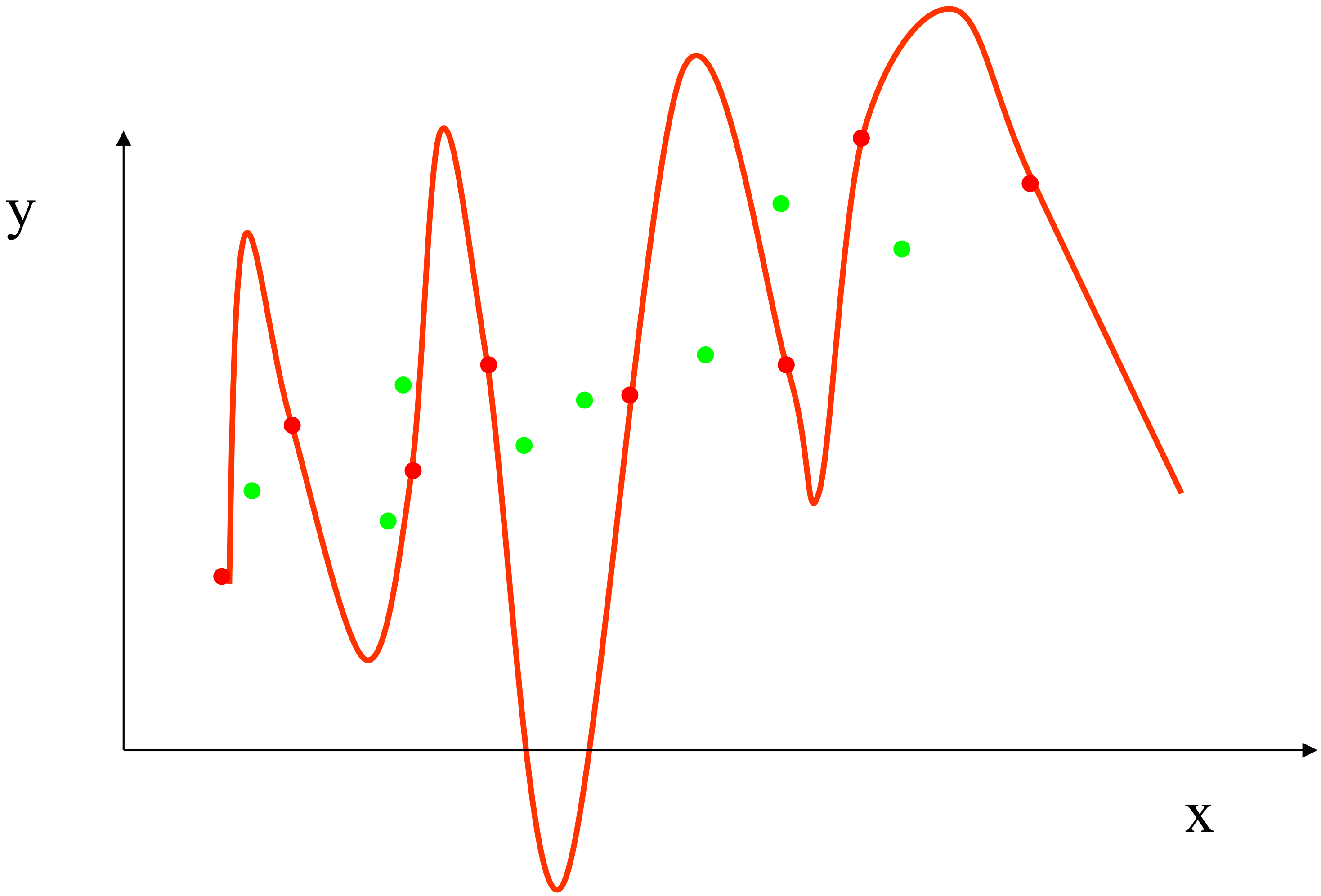


Overfitting & Complexity

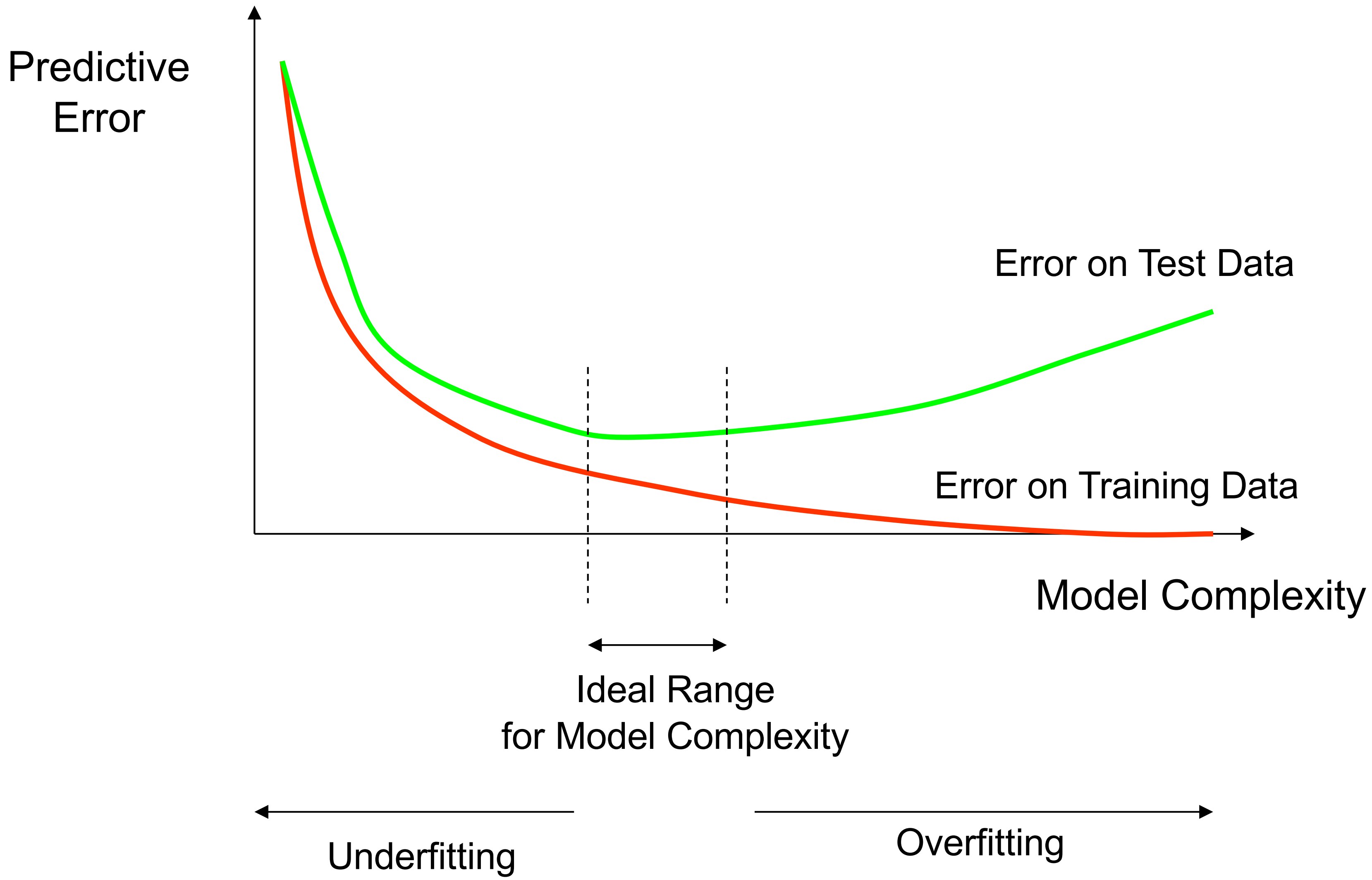
Simple model: $Y = aX + b + e$



Overfitting & Complexity



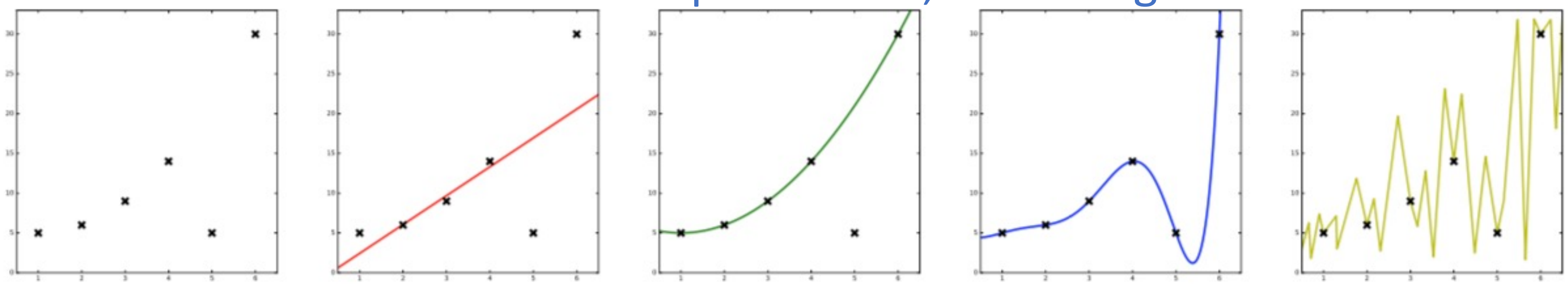
How Overfitting Affects Prediction



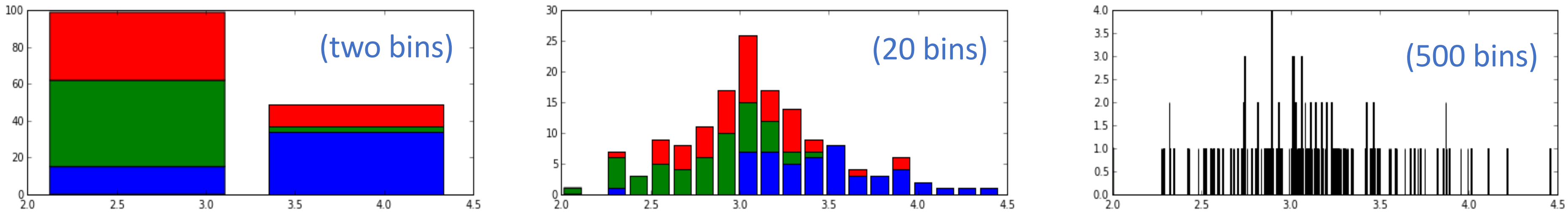
Recall: Inductive bias

- How can we transfer observations to other, unobserved values?

Data Models that “explain” data, extending to other values



- For p(x,y)? One option: discretize (histograms)



  - Binning “transfers” data density to nearby feature values
  - Too few bins = lose information; too many = noisy, no estimates at many locations

Fundamental issue of ML: How can we transfer information from “similar” examples?

Outline

How does ML work?

Ex: Centroid Classifier

Optimal Decisions (in theory)

Bayes Classifiers

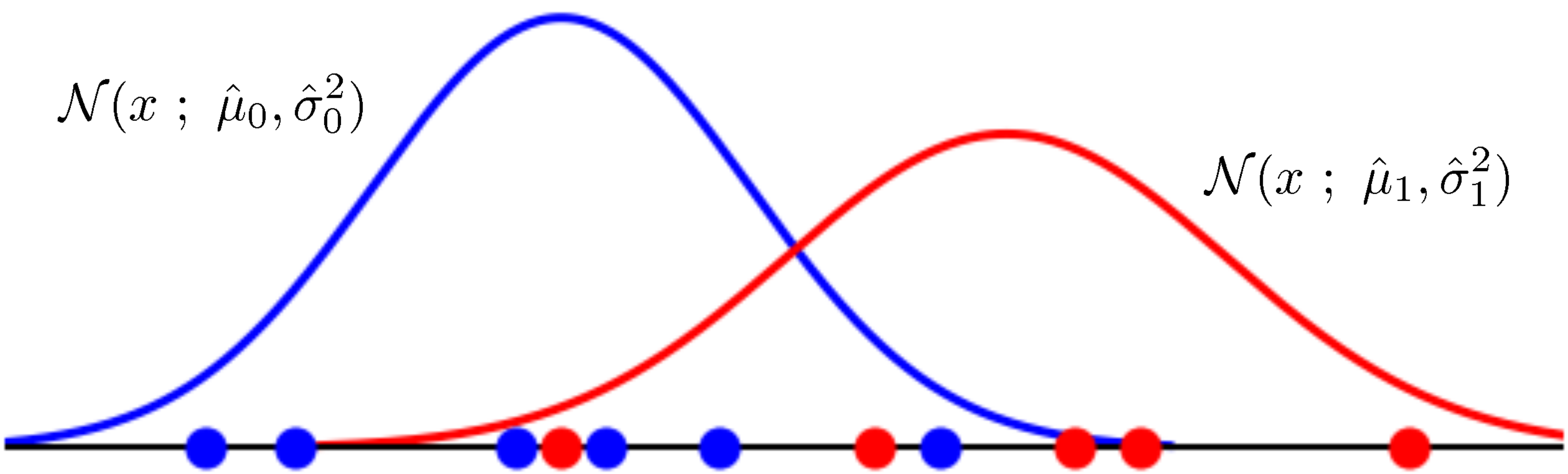
Types of Errors

Gaussian probability models

- Estimate parameters of a Gaussian distribution from data
  - Gaussian dist: $\mathcal{N}(x\,;\,\mu_c,\sigma_c^2) = \left(2\pi\sigma_c^2\right)^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(x-\mu_c)^2/\sigma_c^2\right]$
  - Empirical (and maximum likelihood) parameter estimates:

$$\hat{p}(Y=1) = \frac{m_1}{m} \qquad \hat{\mu}_1 = \frac{1}{m_1}\sum_{i:y^{(i)}=1} x^{(i)} \qquad \hat{\sigma}_1^2 = \frac{1}{m_1}\sum_{i:y^{(i)}=1} (x^{(i)}-\hat{\mu}_1)^2$$

(and similarly for class 0)



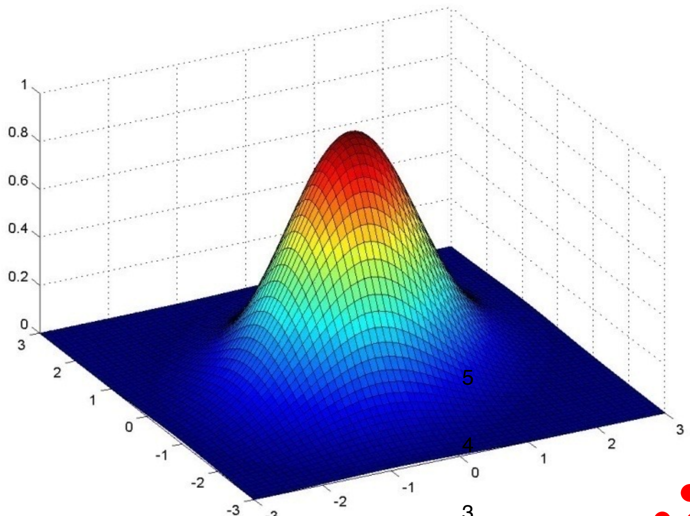
Multivariate Gaussian models

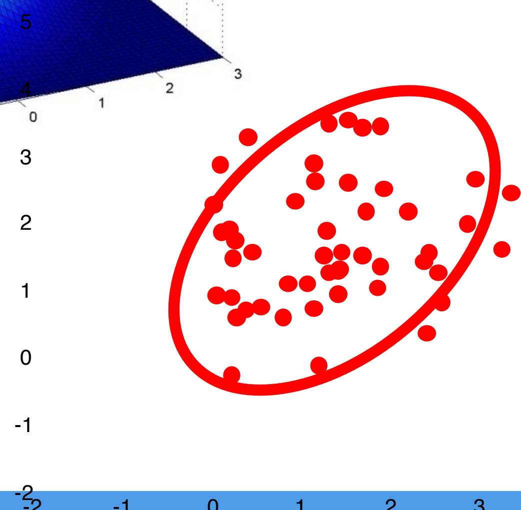
- Similar to univariate case

$$\mathcal{N}(x\,;\,\mu,\Sigma) = \frac{1}{(2\pi)^{n/2}}|\Sigma|^{-\frac{1}{2}}\exp\left\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\}$$

$$\mu = n \times 1 \text{ mean vector}$$

$$\Sigma = n \times n \text{ covariance matrix}$$





Maximum likelihood estimate:

$$\hat{\mu} = \frac{1}{m}\sum_j x^{(j)}$$

$$\hat{\Sigma} = \frac{1}{m}\sum_j (x^{(j)} - \hat{\mu})^T(x^{(j)} - \hat{\mu})$$

Bayes rule

- How to compute the probability of a hidden "cause" Y, after observing some evidence "effect" X:

How probable is the hidden cause?

$$p(Y|X)\; p(X) = p(X,Y) = p(X|Y)\; p(Y)$$

How often does Y cause X?

$$\Rightarrow \quad p(Y|X) = \frac{p(X|Y)\; p(Y)}{p(X)}$$

"Bayes rule"

- Example: flu
  - P(F), P(H|F)
  - P(F=1 | H=1) = ?

$$= \frac{0.50 * 0.05}{0.50 * 0.05 + 0.20 * 0.95} = 0.116$$

F	P(F)
0	0.95
1	0.05

F	H	P(H|F)
0	0	0.80
0	1	0.20
1	0	0.50
1	1	0.50

Bayes Classifiers from Data

- Estimate prior probability of each class, p(y)
  - E.g., how common is each type of Iris?
- Distribution of features given the class, p(x | y=c)
  - How likely are we to see "x" in each type of iris?
- Joint distribution $p(y|x)p(x) = p(x,y) = p(x|y)p(y)$
- Bayes Rule: $\Rightarrow \quad p(y|x) = p(x|y)p(y)/p(x)$

$$= \frac{p(x|y)p(y)}{\sum_c p(x|y=c)p(y=c)}$$

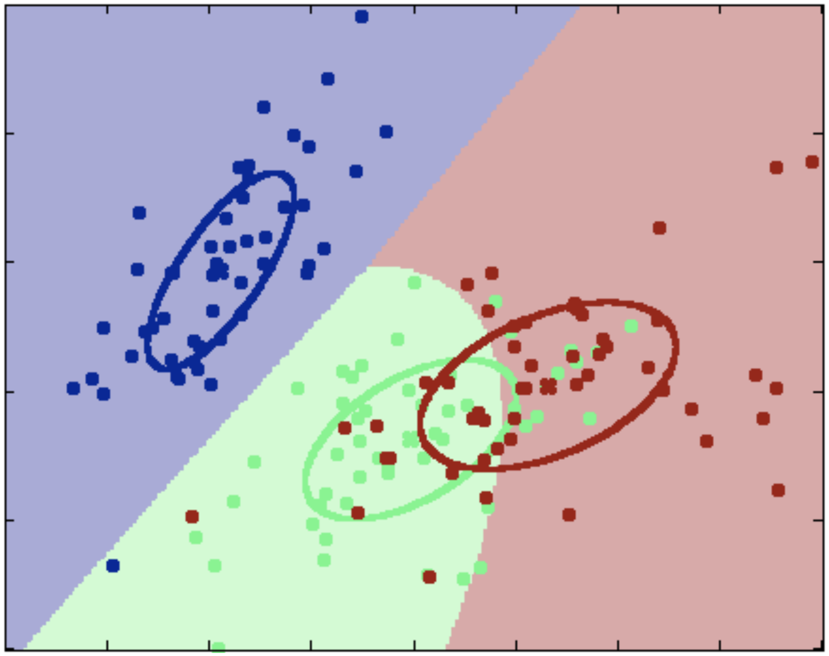
(Use the rule of total probability to calculate the denominator!)

Example: Gaussian Bayes, Iris Data

- Fit Gaussian distribution to each class {0,1,2}

$$p(y) = \text{Discrete}(\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3})$$

$$p(x_1, x_2 | y = 0) = \mathcal{N}(x\,;\, \mu_0, \Sigma_0)$$
$$p(x_1, x_2 | y = 1) = \mathcal{N}(x\,;\, \mu_1, \Sigma_1)$$
$$p(x_1, x_2 | y = 2) = \mathcal{N}(x\,;\, \mu_2, \Sigma_2)$$



Then, Bayes rule:

(How well does Y=blue explain x?)

(How well do Y=green or Y=red explain x?)

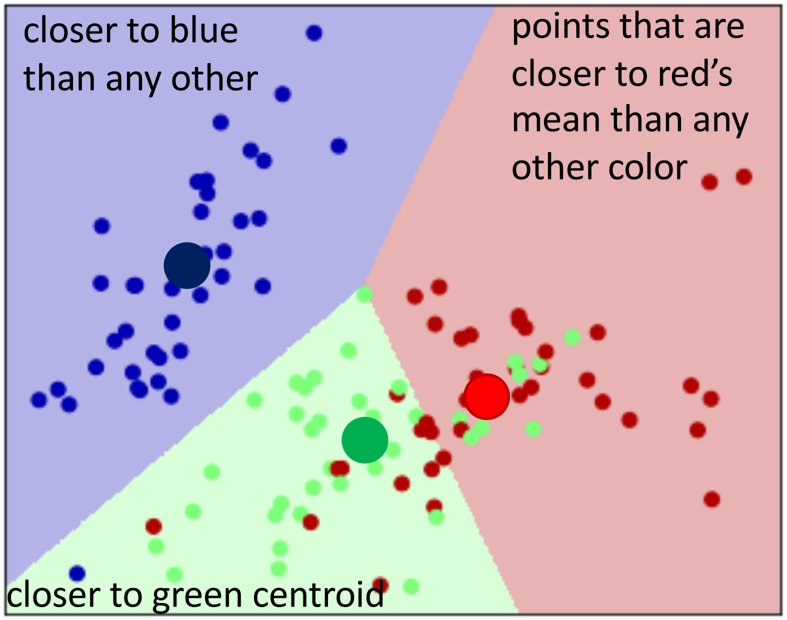
$$p(Y=b|x) = \frac{p(Y=b)p(x|Y=b)}{p(Y=b)p(x|Y=b) + p(Y=g)p(x|Y=g) + p(Y=r)p(x|Y=r)}$$

Homework: Centroid Classifier

- Simple, special case of Gaussian Bayes classifier
- Estimate just the mean (centroid) of each data class
  - Then, rule is simply:

  predict class y by:

$$\hat{y}(x) = \arg\min_c \|x - \mu_c\|^2$$



Typically, use Euclidean distance:

$$\|x - \mu\|^2 = \sum_j (x_j - \mu_j)^2$$

though other distances also possible (more later…)

What about discrete features?

- Estimate joint probability for each class
  - E.g., how many times (what fraction) did each outcome occur?

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

- m data $\ll$ 2^n parameters?

- What about the zeros?
  - We learn that certain combinations are impossible?
  - What if we see these later in test data?

- Overfitting!

What about discrete features?

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

- Estimate joint probability for each class
  - E.g., how many times (what fraction) did each outcome occur?
- m data << 2^n parameters?
- What about the zeros?
  - We learn that certain combinations are impossible?
  - What if we see these later in test data?
- One option: regularize $\hat{p}(a,b,c) \propto (M_{abc} + \alpha)$
- Normalize to make sure values sum to one…

Naïve Bayes Classifiers

- Another option: reduce the model complexity by assuming the features are (conditionally) independent of one another
- Independence: $p(a,b) = p(a)\, p(b)$
- $p(x_1, x_2, \ldots x_N \mid y=1) = p(x_1 \mid y=1)\, p(x_2 \mid y=1) \ldots p(x_N \mid y=1)$
- Only need to estimate each 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

$\Rightarrow$

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	

Example: Naïve Bayes

Observed Data:

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

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

$$\hat{p}(x_1, x_2 | y=0) = \hat{p}(x_1 | y=0)\, \hat{p}(x_2 | y=0)$$

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

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

Prediction given some observation x?

$$\hat{p}(y=1)\hat{p}(x=11|y=1) \quad \substack{<\\>} \quad \hat{p}(y=0)\hat{p}(x=11|y=0)$$

$$\frac{4}{8} \times \frac{2}{4} \times \frac{1}{4} \qquad\qquad \frac{4}{8} \times \frac{3}{4} \times \frac{2}{4}$$

Decide class 0

Example: Naïve Bayes

Observed Data:

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

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

$$\hat{p}(x_1, x_2 | y=0) = \hat{p}(x_1 | y=0)\, \hat{p}(x_2 | y=0)$$

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

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

$$\hat{p}(y=1 | x_1 = 1, x_2 = 1) = \frac{\frac{4}{8} \times \frac{2}{4} \times \frac{1}{4}}{\frac{3}{4} \times \frac{2}{4} \times \frac{4}{8} + \frac{2}{4} \times \frac{1}{4} \times \frac{4}{8}}$$

$$= \frac{1}{4}$$

Example: Joint Bayes

Observed Data:

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

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

$$\hat{p}(x_1, x_2 | y=0) =$$

x_1	x_2	p(x | y=0)
0	0	1/4
0	1	0/4
1	0	1/4
1	1	2/4

$$\hat{p}(x_1, x_2 | y=1) =$$

x_1	x_2	p(x | y=1)
0	0	1/4
0	1	1/4
1	0	2/4
1	1	0/4

$$\hat{p}(y=1 | x_1=1, x_2=1) = \frac{\frac{4}{8} \times 0}{\frac{2}{4} \times \frac{4}{8} + 0 \times \frac{4}{8}}$$

$$= 0$$

Naïve Bayes Models

- Variable y to predict, e.g. "auto accident in next year?"
- *Many* co-observed variables $x=[x_1 \ldots x_n]$
  - Age, income, education, zip code, …
- Learn $p(y \mid x_1 \ldots x_n)$, to predict y?
  - Arbitrary distribution: $O(d^n)$ values!
- Naïve Bayes:

Now only 2*n*d parameters!

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

Bayes Rule

$$p(x|y) = \prod_j p(x_j|y)$$

"Naïve" : conditional independence

- Note: may not be a good model of the data
  - Doesn't capture correlations in features
  - Can't capture some dependencies
- But in practice it often does quite well!

Outline

- How does ML work?
- Ex: Centroid Classifier
- Optimal Decisions (in theory)
- Bayes Classifiers
- **Types of Errors**

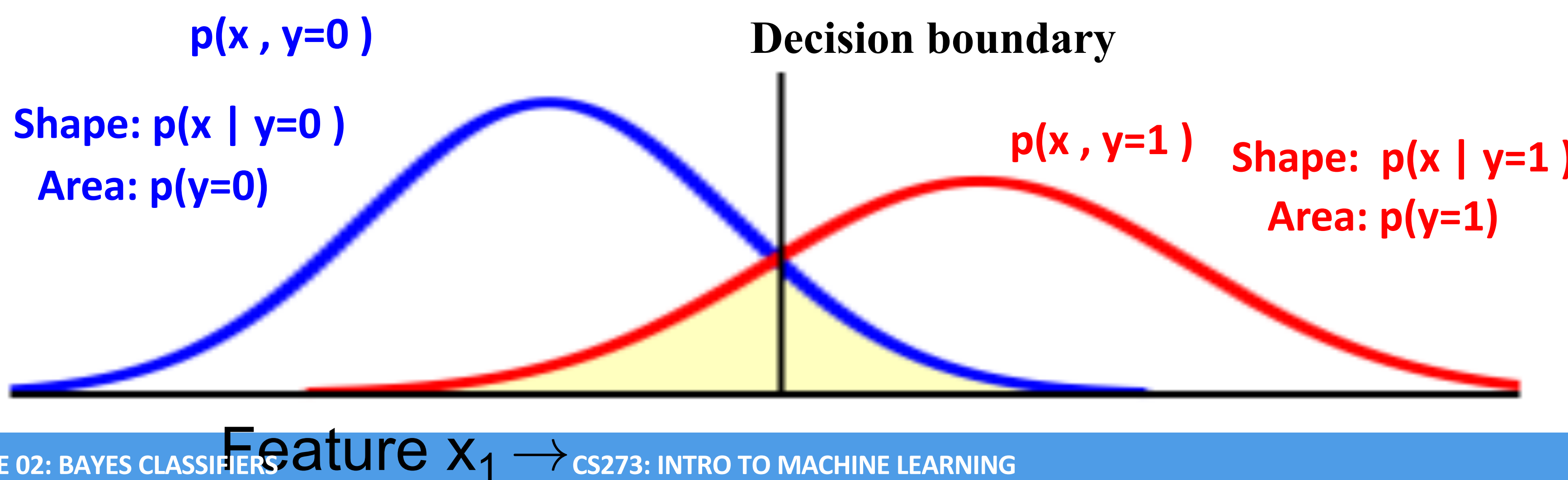
Bayes Classifiers

- Bayes classification decision rule compares probabilities:

$$p(y=0|x) \underset{>}{<} p(y=1|x)$$

$$= p(y=0,x) \underset{>}{<} p(y=1,x)$$

- Can visualize this nicely if x is a scalar:

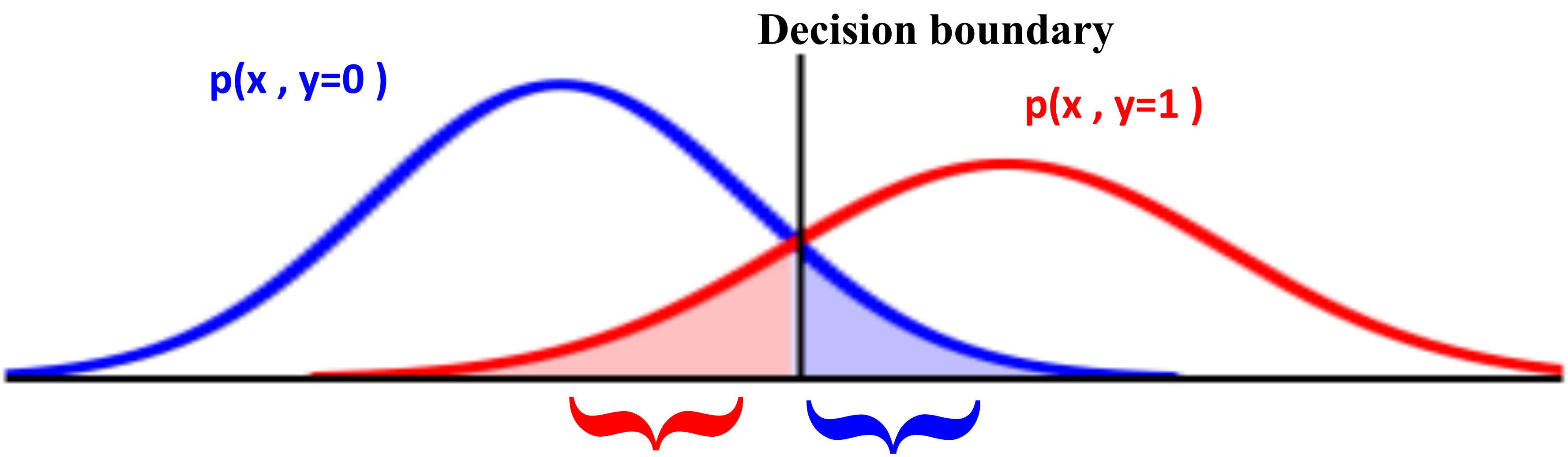


Bayes Classifiers

- Not all errors are created equally…
- Risk associated with each outcome?

Add multiplier alpha:

$$\alpha \ p(y=0,x) \ \substack{<\\>} \ p(y=1,x)$$



Type 1 errors: false positives

Type 2 errors: false negatives

False positive rate: (# y=0, ŷ=1) / (#y=0)

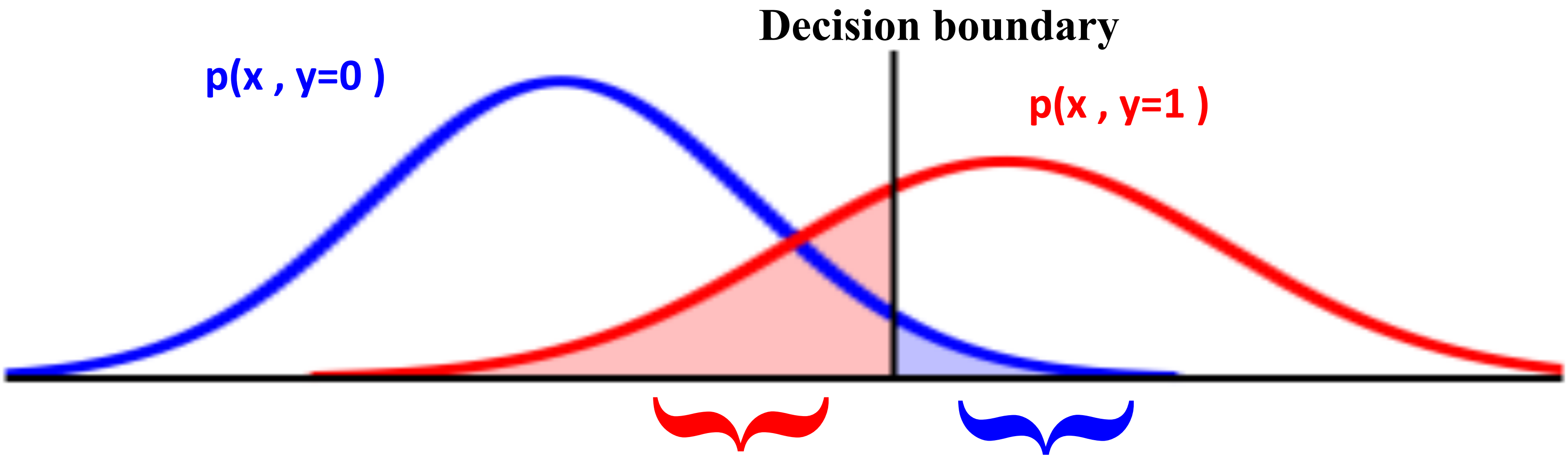
False negative rate: (# y=1, ŷ=0) / (#y=1)

Bayes Classifiers

- Increase alpha: prefer class 0
- Spam detection

Add multiplier alpha:

$$\alpha\ p(y=0,x) \ \overset{<}{>}\ p(y=1,x)$$



Type 1 errors: false positives

Type 2 errors: false negatives

False positive rate: (# y=0, ŷ=1) / (#y=0)

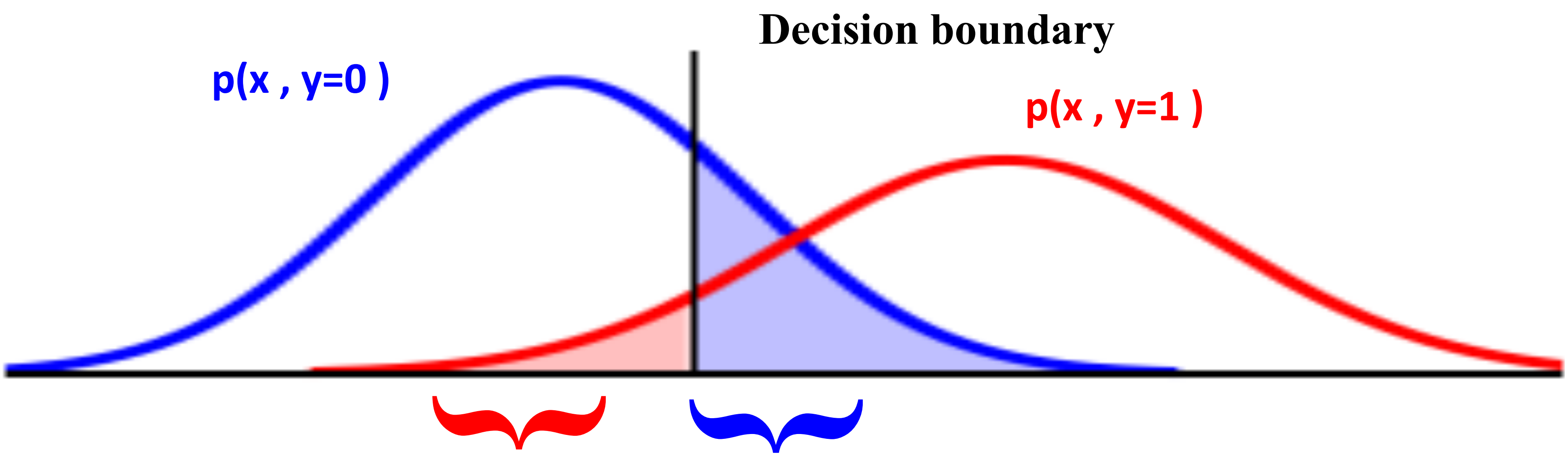
False negative rate: (# y=1, ŷ=0) / (#y=1)

Bayes Classifiers

- Decrease alpha: prefer class 1
- Cancer detection

Add multiplier alpha:

$$\alpha \; p(y=0,x) \;\underset{>}{<}\; p(y=1,x)$$



Type 1 errors: false positives

Type 2 errors: false negatives

False positive rate: (# y=0, ŷ=1) / (#y=0)

False negative rate: (# y=1, ŷ=0) / (#y=1)

Measuring Errors

- Confusion matrix
- Can extend to more classes

	Predict 0	Predict 1
Y=0	380	5
Y=1	338	3

- True positive rate: #(y=1 , ŷ=1) / #(y=1) -- "sensitivity"
- False negative rate: #(y=1 , ŷ=0) / #(y=1)
- False positive rate: #(y=0 , ŷ=1) / #(y=0)
- True negative rate: #(y=0 , ŷ=0) / #(y=0) -- "specificity"

Likelihood Ratio Tests

- Connection to classical, statistical decision theory:

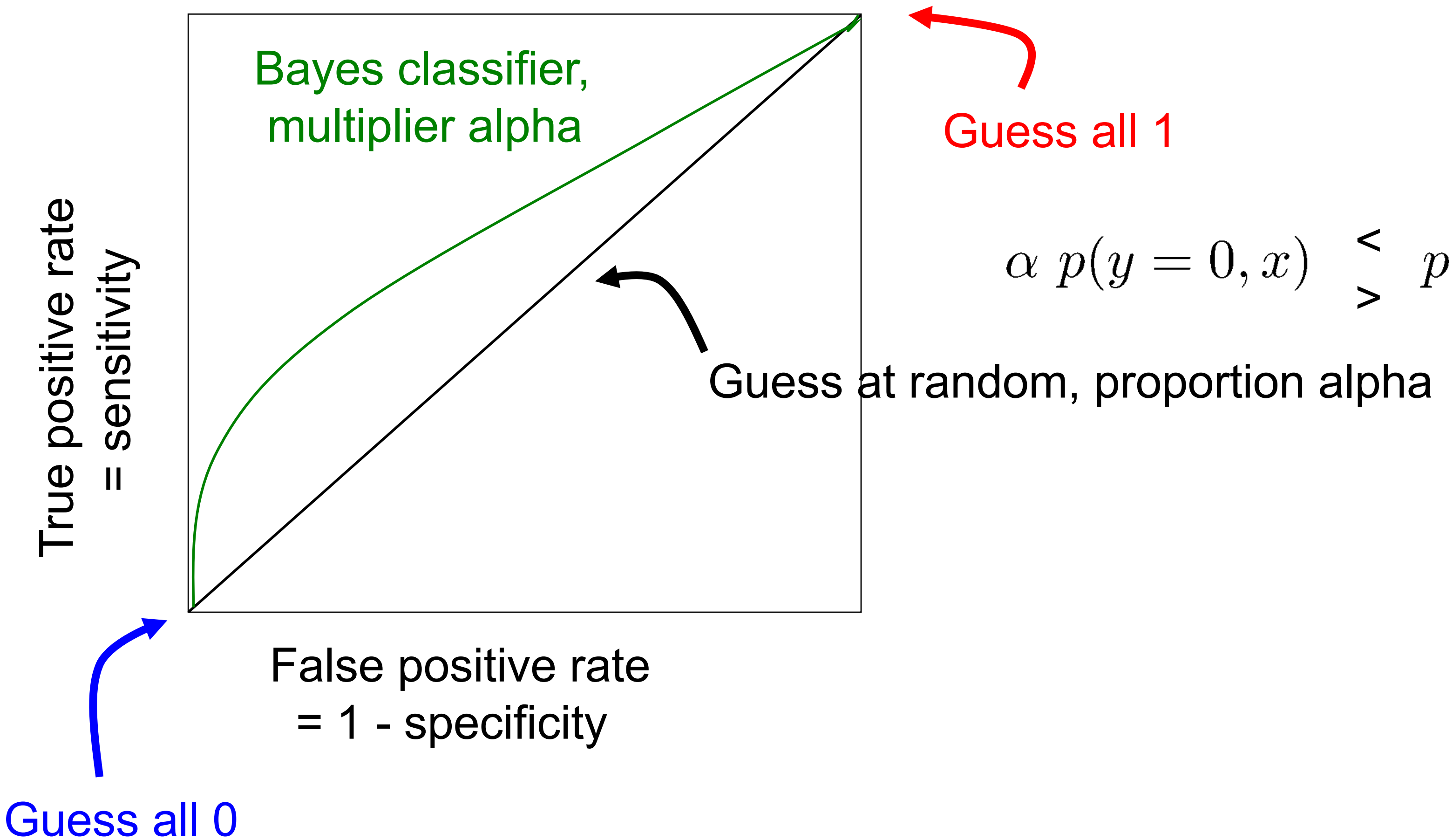
$$p(y=0,x) \underset{>}{<} p(y=1,x) \quad = \quad \log \frac{p(y=0)}{p(y=1)} \underset{>}{<} \log \frac{p(x|y=1)}{p(x|y=0)}$$

"log likelihood ratio"

- Likelihood ratio: relative support for observation "x" under "alternative hypothesis" y=1, compared to "null hypothesis" y=0
- Can vary the decision threshold: $\gamma \underset{>}{<} \log \frac{p(x|y=1)}{p(x|y=0)}$
- Classical testing:
  - Choose gamma so that FPR is fixed ("p-value")
  - Given that y=0 is true, what's the probability we decide y=1?

ROC Curves

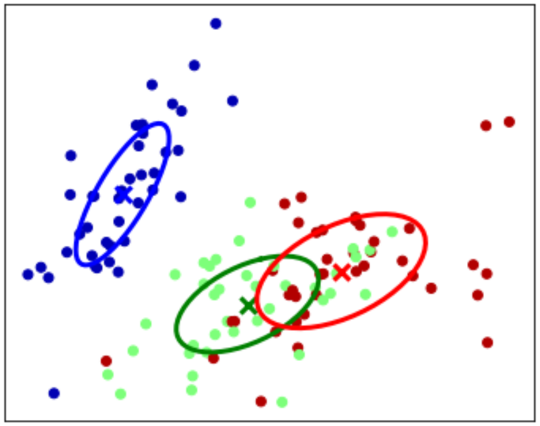
- Characterize performance as we vary the decision threshold?



$$\alpha\ p(y=0,x) \ \lessgtr\ p(y=1,x)$$

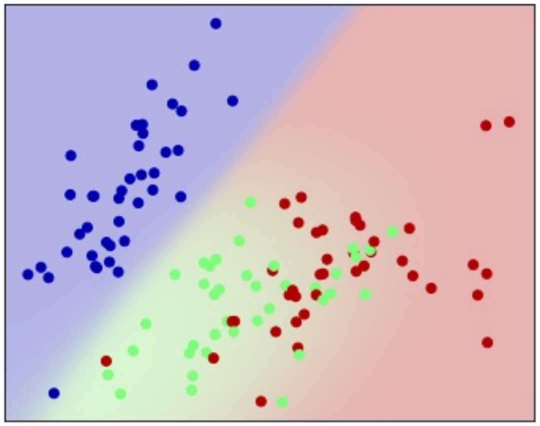
Types of Supervised Learning

Probabilistic
Generative Learning



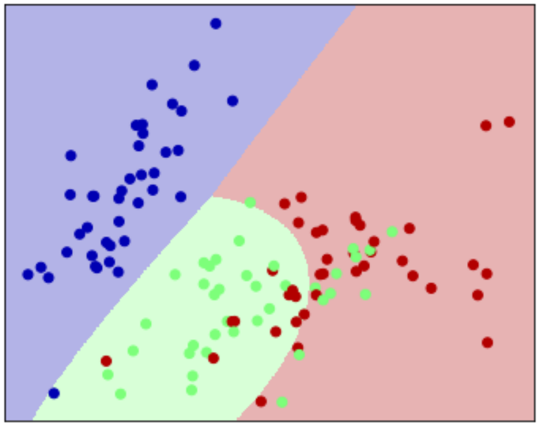
Full "generative" model
Also explain features,
e.g., p(y,x)

Probabilistic
Discriminative Learning



"Soft" predictions
Probability / confidence,
e.g., p(y|x)

Discriminative Learning



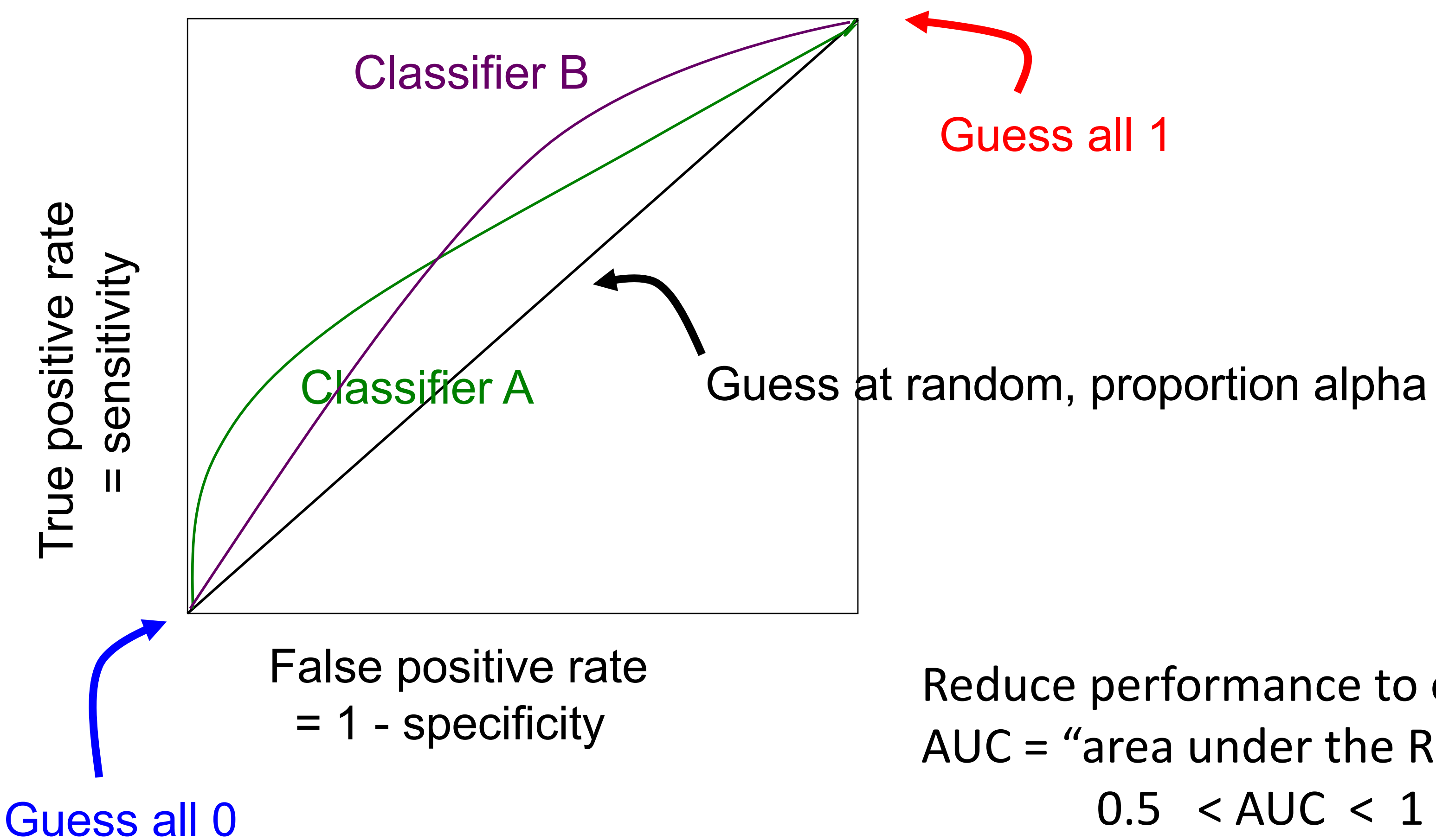
"Hard" (discrete) predictions
Minimize loss, e.g., error rate

Confidence predictions allow us to change our desired loss "after" training:

- Care more about one type of error than another?
- Expect more of one class than the other?
- (Easier to) combine different predictions? (see: ensembles)

ROC Curves

- Characterize performance as we vary our confidence threshold?



Reduce performance to one number?

AUC = "area under the ROC curve"

$0.5 < AUC < 1$

Questions?