10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Logistic Regression, Nonlinear Features, Regularization

Logistic Regression Readings:
Murphy 8.1-8.3, 8.6
Bishop 4.3.2, 4.3.4
HTF 4.1, 4.4
Mitchell –

"Generative … Logistic Regression" (Mitchell, 2016)
"Maximum … Gradient Training" (Elkan, 2014)

Matt Gormley
Lecture 9
February 15, 2016

Reminders

- **Homework 3: Linear / Logistic Regression**
  - Release: Mon, Feb. 13
  - Due: Wed, Feb. 22 at 11:59pm

Note the change in time.

Outline

- **Motivation:**
  - Choosing the right classifier
  - Example: Image Classification
- **Logistic Regression**
  - Background: Hyperplanes
  - Data, Model, Learning, Prediction
  - Log-odds
  - Bernoulli interpretation
  - Maximum Conditional Likelihood Estimation
- **Gradient descent for Logistic Regression**
  - Stochastic Gradient Descent (SGD)
  - Computing the gradient
  - Details (learning rate, finite differences)
- **Nonlinear Features**

MOTIVATION: LOGISTIC REGRESSION

Classifiers

Which classification method should we use?

1. The one that gives the best predictions…
   – on the training data
   – on the (unseen) test data
   – on the (held-out) validation data
2. The one that is computationally efficient…
   – during training
   – during classification
3. The most interpretable one…
   – in terms of its parameters
   – as a model
4. The one that is easiest to implement…
   – for learning
   – for classification

Classifiers

Which classification method should we use?

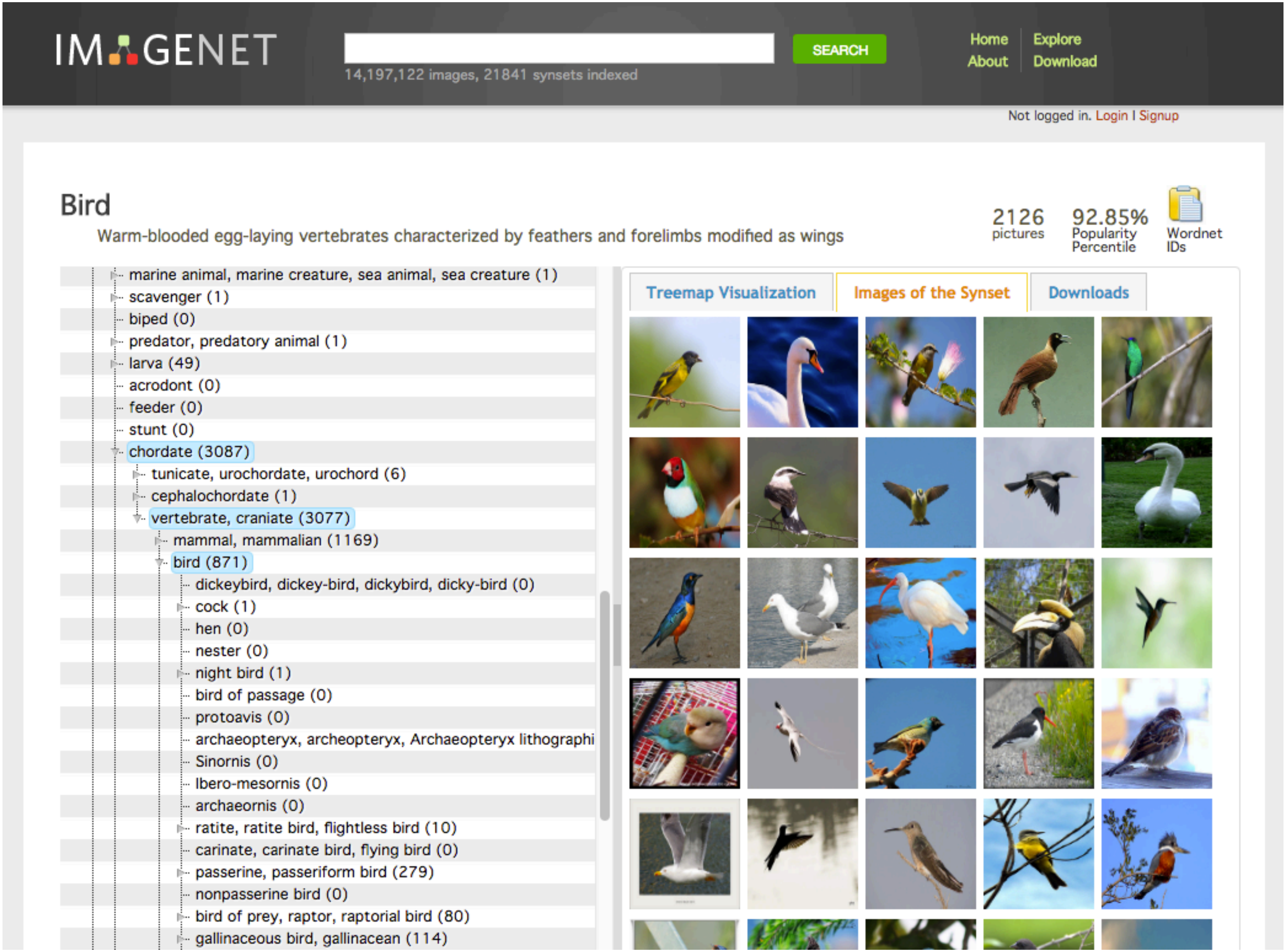
Naïve Bayes defined a generative model $p(\mathbf{x}, y)$ of the features $\mathbf{x}$ and the class y.

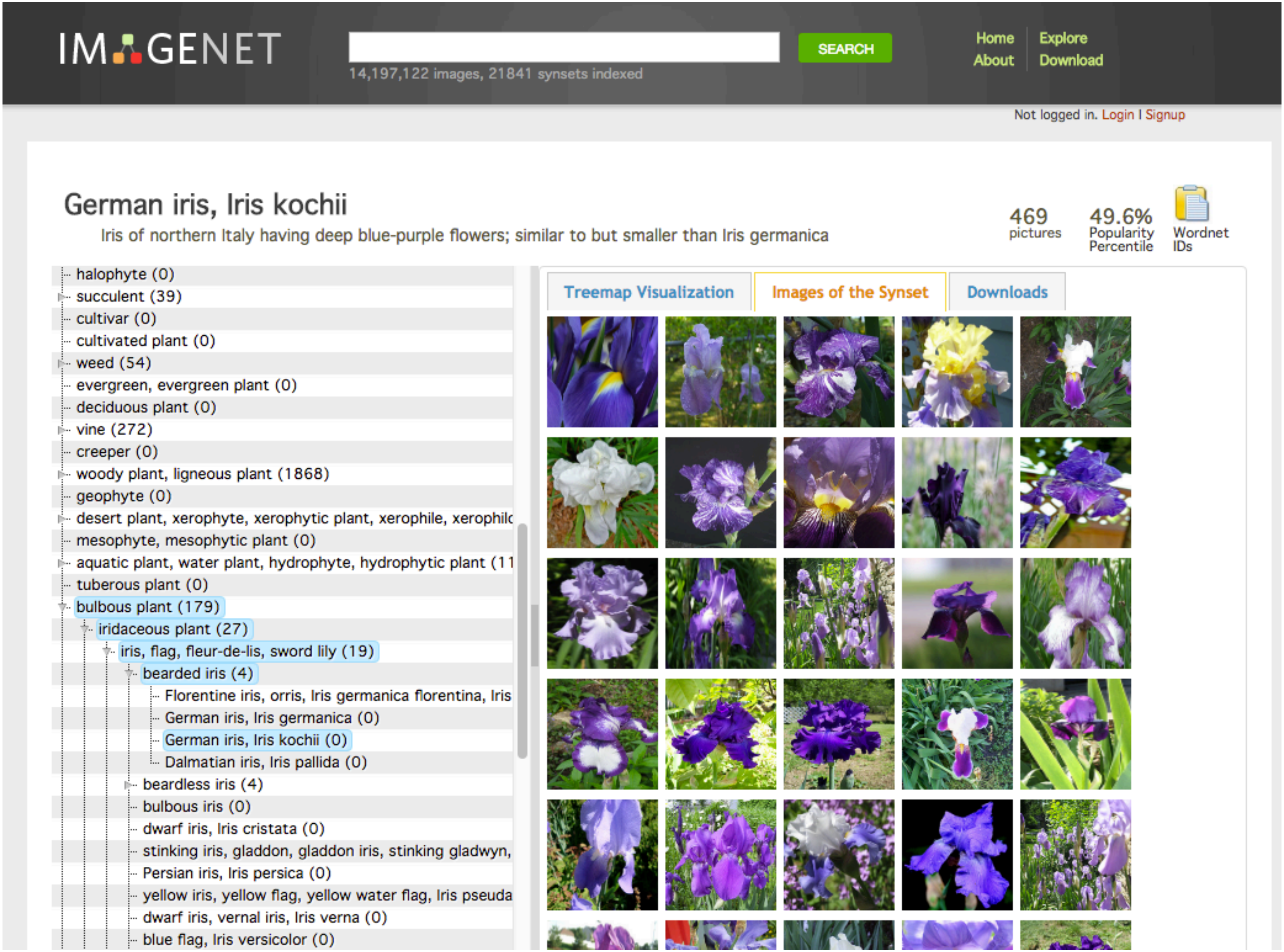
Why should we define a model of $p(\mathbf{x}, y)$ at all?

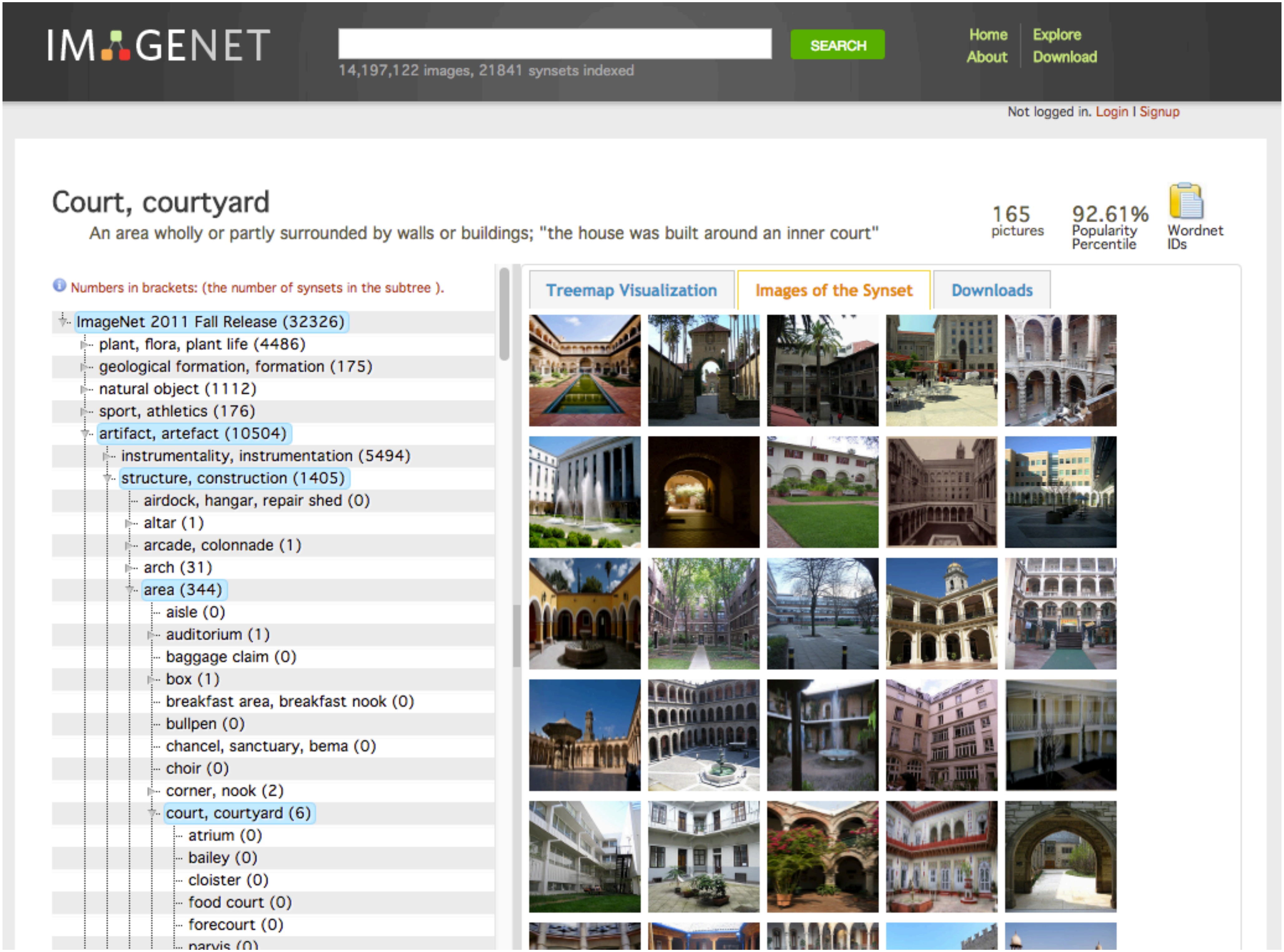
Why not directly model $p(y \mid \mathbf{x})$?

Example: Image Classification

- ImageNet LSVRC-2010 contest:
  - **Dataset:** 1.2 million labeled images, 1000 classes
  - **Task**: Given a new image, label it with the correct class
  - **Multiclass** classification problem
- Examples from http://image-net.org/


IMAGENET
SEARCH
14,197,122 images, 21841 synsets indexed
Home
About
Explore
Download
Not logged in. Login I Signup
Bird
Warm-blooded egg-laying vertebrates characterized by feathers and forelimbs modified as wings
2126 pictures
92.85% Popularity Percentile
Wordnet IDs
marine animal, marine creature, sea animal, sea creature (1)
scavenger (1)
biped (0)
predator, predatory animal (1)
larva (49)
acrodont (0)
feeder (0)
stunt (0)
chordate (3087)
tunicate, urochordate, urochord (6)
cephalochordate (1)
vertebrate, craniate (3077)
mammal, mammalian (1169)
bird (871)
dickeybird, dickey-bird, dickybird, dicky-bird (0)
cock (1)
hen (0)
nester (0)
night bird (1)
bird of passage (0)
protoavis (0)
archaeopteryx, archeopteryx, Archaeopteryx lithographi
Sinornis (0)
Ibero-mesornis (0)
archaeornis (0)
ratite, ratite bird, flightless bird (10)
carinate, carinate bird, flying bird (0)
passerine, passeriform bird (279)
nonpasserine bird (0)
bird of prey, raptor, raptorial bird (80)
gallinaceous bird, gallinacean (114)
Treemap Visualization
Images of the Synset
Downloads


IMAGENET
SEARCH
14,197,122 images, 21841 synsets indexed
Home
About
Explore
Download
Not logged in. Login | Signup
German iris, Iris kochii
Iris of northern Italy having deep blue-purple flowers; similar to but smaller than Iris germanica
469
pictures
49.6%
Popularity
Percentile
Wordnet
IDs
halophyte (0)
succulent (39)
cultivar (0)
cultivated plant (0)
weed (54)
evergreen, evergreen plant (0)
deciduous plant (0)
vine (272)
creeper (0)
woody plant, ligneous plant (1868)
geophyte (0)
desert plant, xerophyte, xerophytic plant, xerophile, xerophilc
mesophyte, mesophytic plant (0)
aquatic plant, water plant, hydrophyte, hydrophytic plant (11
tuberous plant (0)
bulbous plant (179)
iridaceous plant (27)
iris, flag, fleur-de-lis, sword lily (19)
bearded iris (4)
Florentine iris, orris, Iris germanica florentina, Iris
German iris, Iris germanica (0)
German iris, Iris kochii (0)
Dalmatian iris, Iris pallida (0)
beardless iris (4)
bulbous iris (0)
dwarf iris, Iris cristata (0)
stinking iris, gladdon, gladdon iris, stinking gladwyn,
Persian iris, Iris persica (0)
yellow iris, yellow flag, yellow water flag, Iris pseuda
dwarf iris, vernal iris, Iris verna (0)
blue flag, Iris versicolor (0)
Treemap Visualization
Images of the Synset
Downloads


IMAGENET
SEARCH
14,197,122 images, 21841 synsets indexed
Home
About
Explore
Download
Not logged in. Login I Signup
Court, courtyard
An area wholly or partly surrounded by walls or buildings; "the house was built around an inner court"
165
pictures
92.61%
Popularity Percentile
Wordnet IDs
Numbers in brackets: (the number of synsets in the subtree).
ImageNet 2011 Fall Release (32326)
plant, flora, plant life (4486)
geological formation, formation (175)
natural object (1112)
sport, athletics (176)
artifact, artefact (10504)
instrumentality, instrumentation (5494)
structure, construction (1405)
airdock, hangar, repair shed (0)
altar (1)
arcade, colonnade (1)
arch (31)
area (344)
aisle (0)
auditorium (1)
baggage claim (0)
box (1)
breakfast area, breakfast nook (0)
bullpen (0)
chancel, sanctuary, bema (0)
choir (0)
corner, nook (2)
court, courtyard (6)
atrium (0)
bailey (0)
cloister (0)
food court (0)
forecourt (0)
parvis (0)
Treemap Visualization
Images of the Synset
Downloads

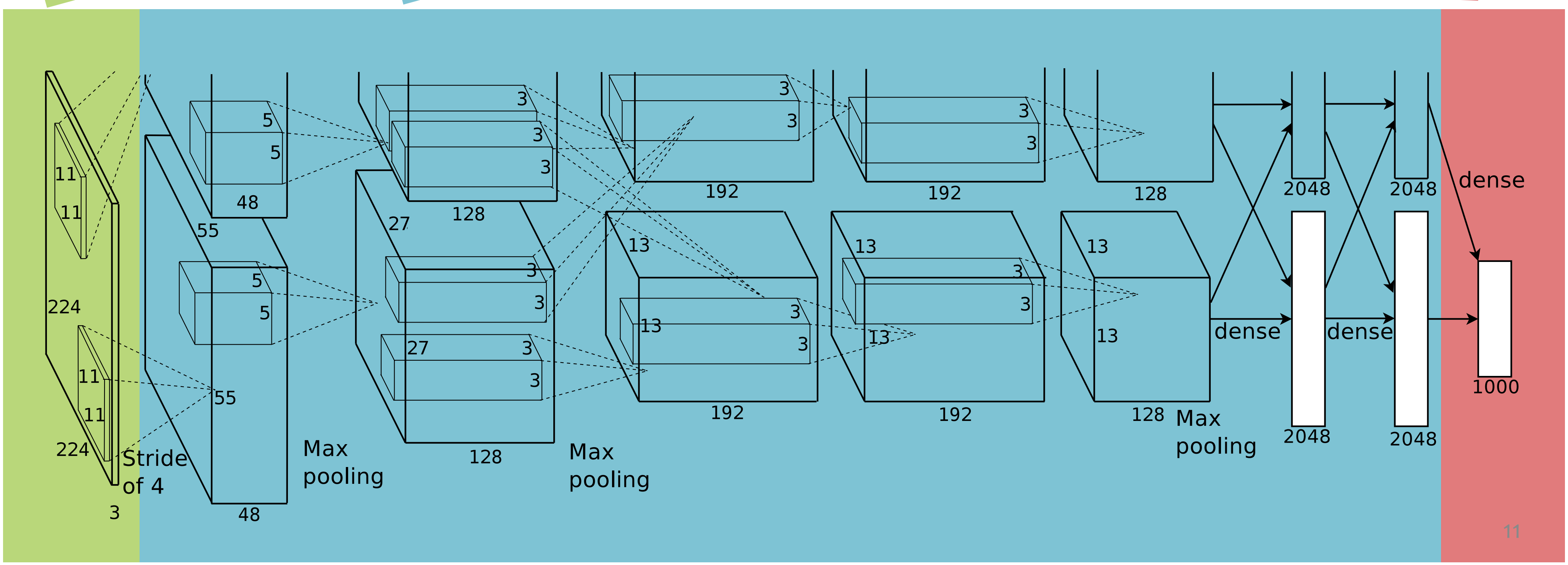
Example: Image Classification

CNN for Image Classification
(Krizhevsky, Sutskever & Hinton, 2011)
17.5% error on ImageNet LSVRC-2010 contest

Input image (pixels)

- Five convolutional layers (w/max-pooling)
- Three fully connected layers

1000-way softmax



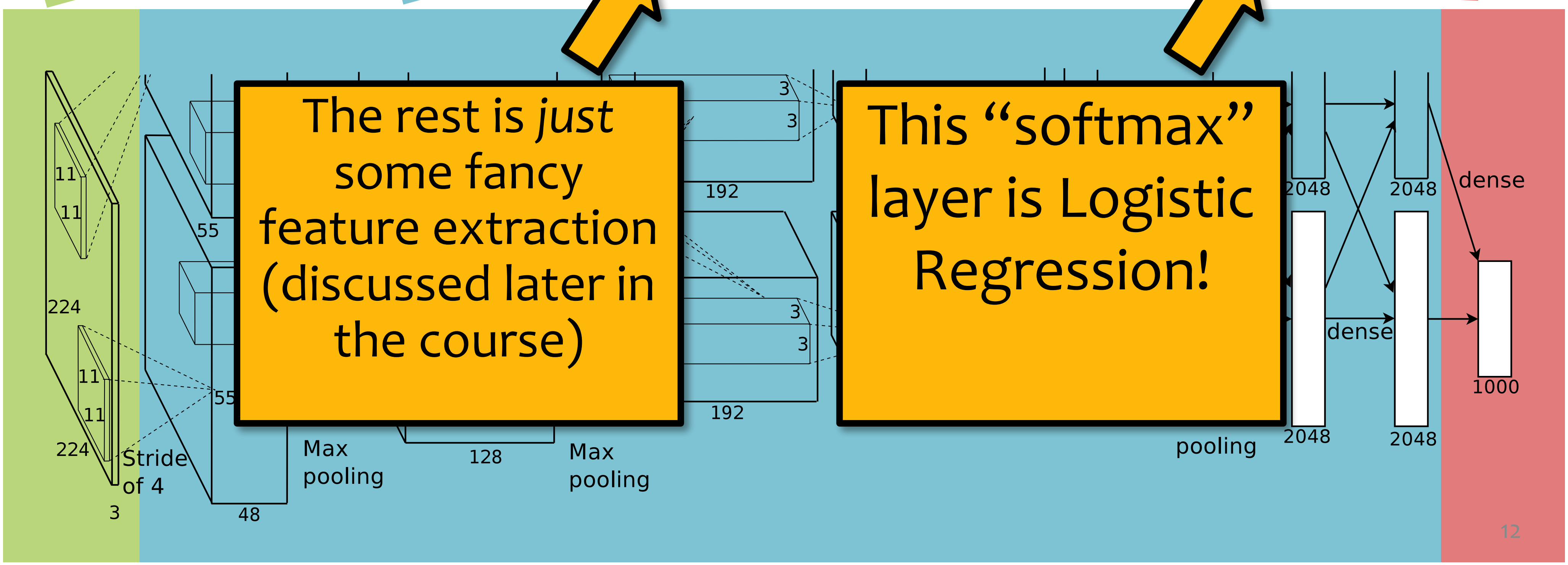
Example: Image Classification

CNN for Image Classification
(Krizhevsky, Sutskever & Hinton, 2011)
17.5% error on ImageNet LSVRC-2010 contest

Input image (pixels)

- Five convolutional layers (w/max-pooling)
- Three fully connected layers

1000-way softmax



The rest is *just* some fancy feature extraction (discussed later in the course)

This "softmax" layer is Logistic Regression!

LOGISTIC REGRESSION

Logistic Regression

Data: Inputs are continuous vectors of length K. Outputs are discrete.

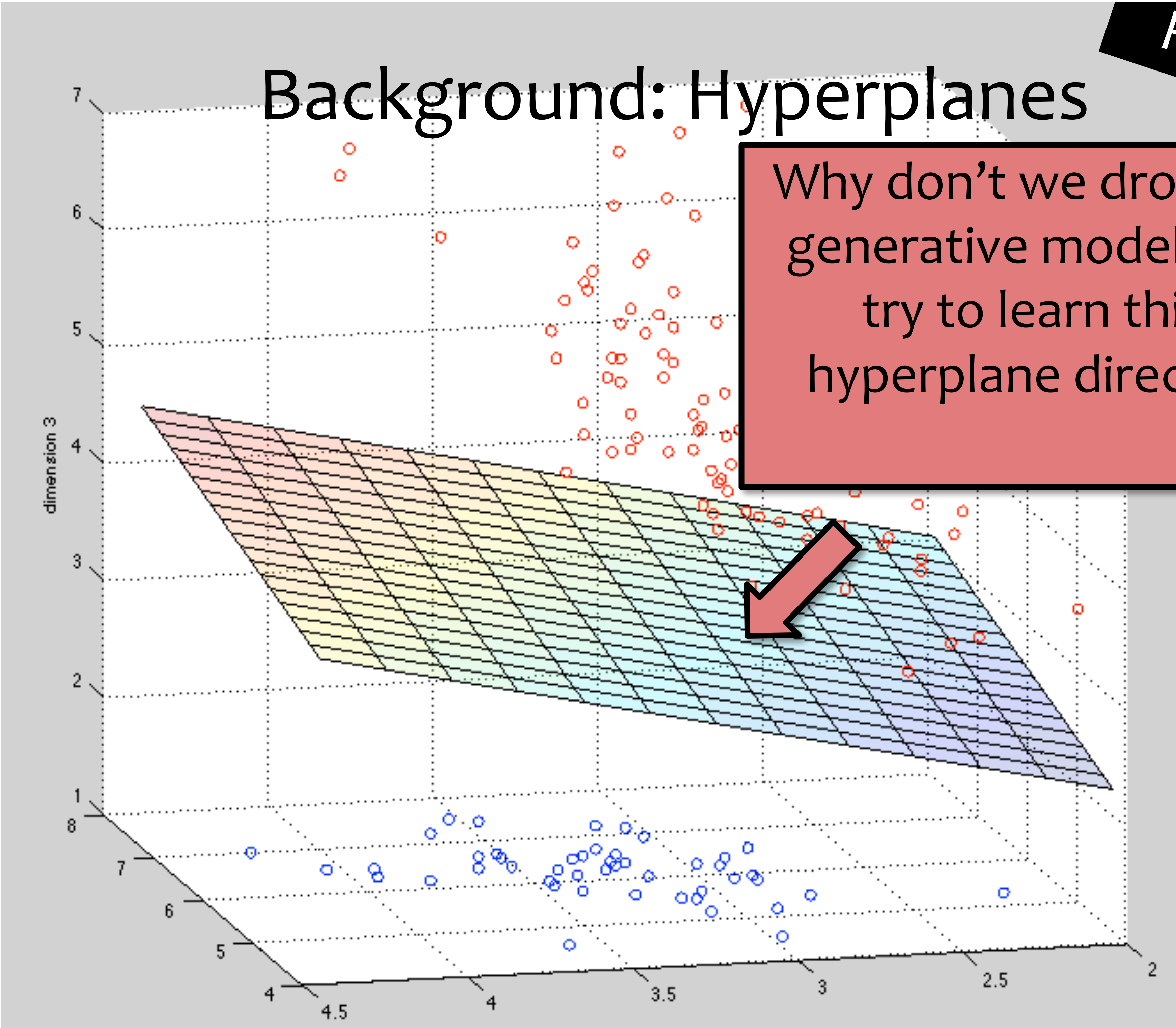
$$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N} \text{ where } \mathbf{x} \in \mathbb{R}^M \text{ and } y \in \{0, 1\}$$

We are back to classification.

Despite the name logistic **regression.**

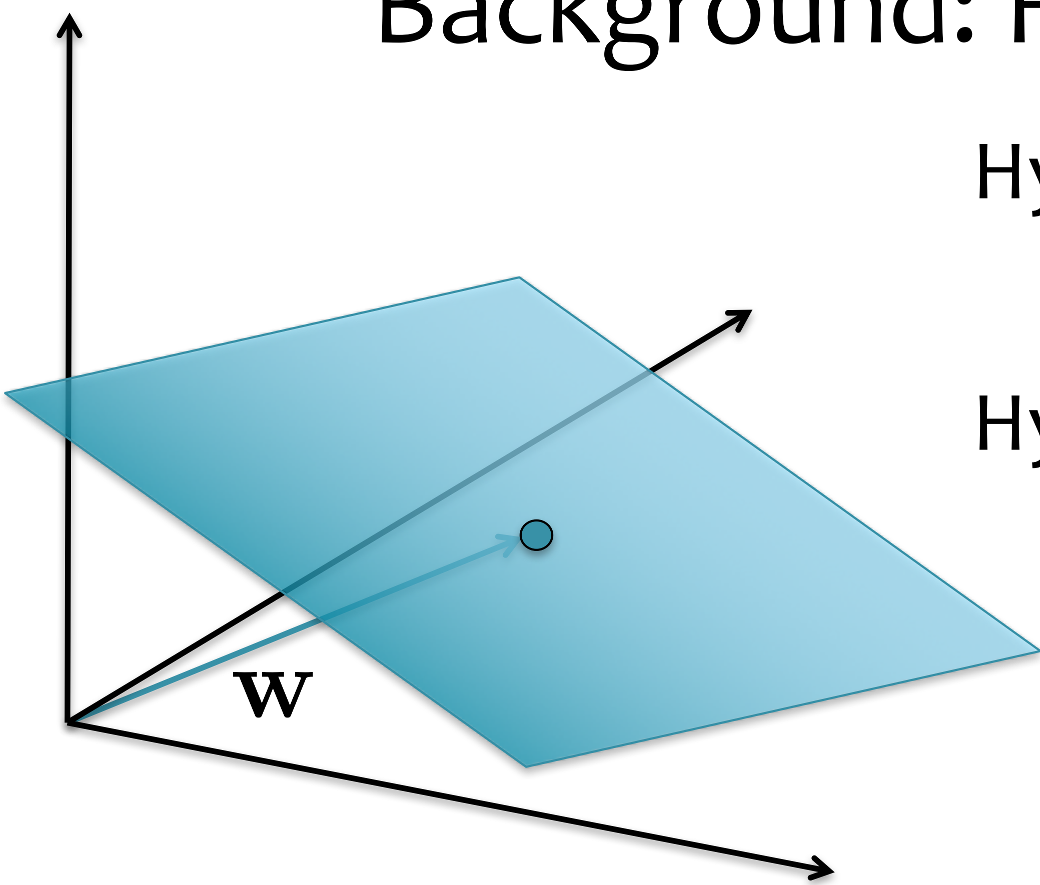
Recall…

Background: Hyperplanes



Why don't we drop the generative model and try to learn this hyperplane directly?

Background: Hyperplanes



Hyperplane (Definition 1):

$$\mathcal{H} = \{\mathbf{x} : \mathbf{w}^T\mathbf{x} = b\}$$

Hyperplane (Definition 2):

$$\mathcal{H} = \{\mathbf{x} : \mathbf{w}^T\mathbf{x} = 0 \text{ and } \mathrm{x}_0 = 1\}$$

Half-spaces:

$$\mathcal{H}^+ = \{\mathbf{x} : \mathbf{w}^T\mathbf{x} > 0 \text{ and } \mathrm{x}_0 = 1\}$$

$$\mathcal{H}^- = \{\mathbf{x} : \mathbf{w}^T\mathbf{x} < 0 \text{ and } \mathrm{x}_0 = 1\}$$

Recall...

d: Hyperplanes

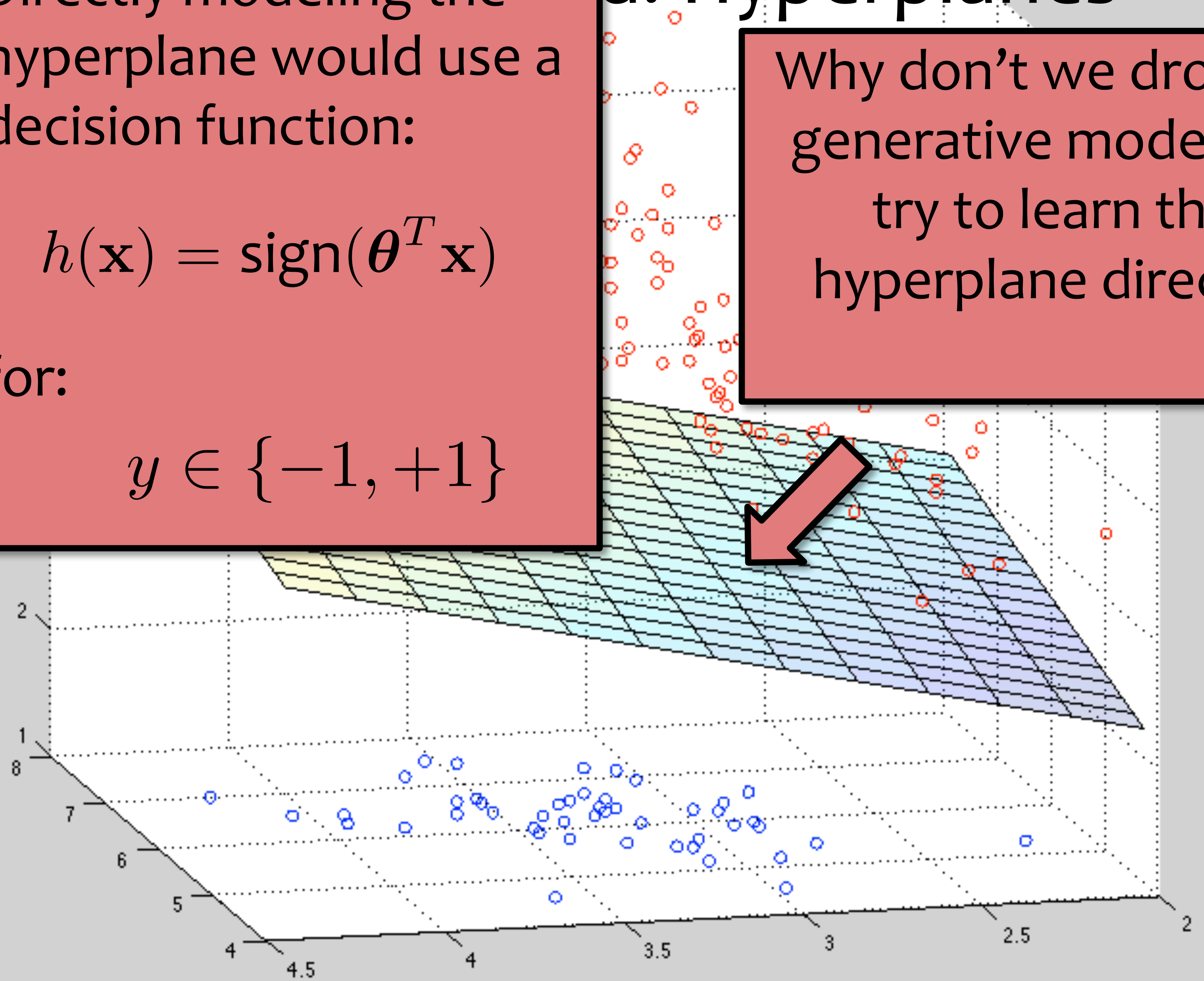
Directly modeling the hyperplane would use a decision function:

$$h(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^T \mathbf{x})$$

for:

$$y \in \{-1, +1\}$$

Why don't we drop the generative model and try to learn this hyperplane directly?



Using gradient ascent for linear classifiers

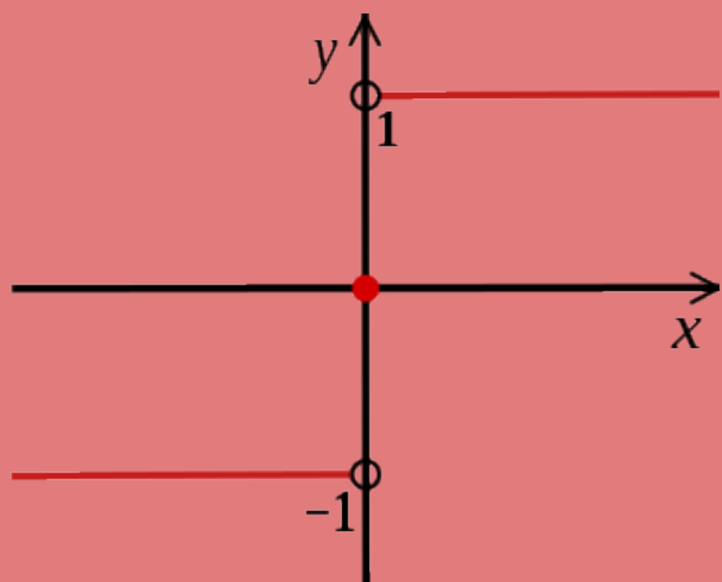
Key idea behind today's lecture:

1. Define a linear classifier (logistic regression)
2. Define an objective function (likelihood)
3. Optimize it with gradient descent to learn parameters
4. Predict the class with highest probability under the model

Using gradient ascent for linear classifiers

This decision function isn't differentiable:

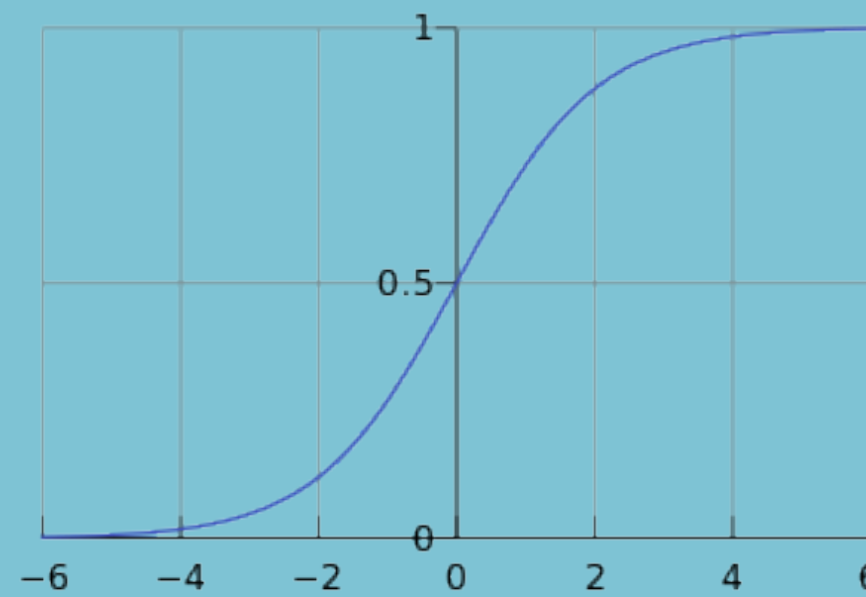
$$h(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^T \mathbf{x})$$



sign(x)

Use a differentiable function instead:

$$p_{\boldsymbol{\theta}}(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})}$$

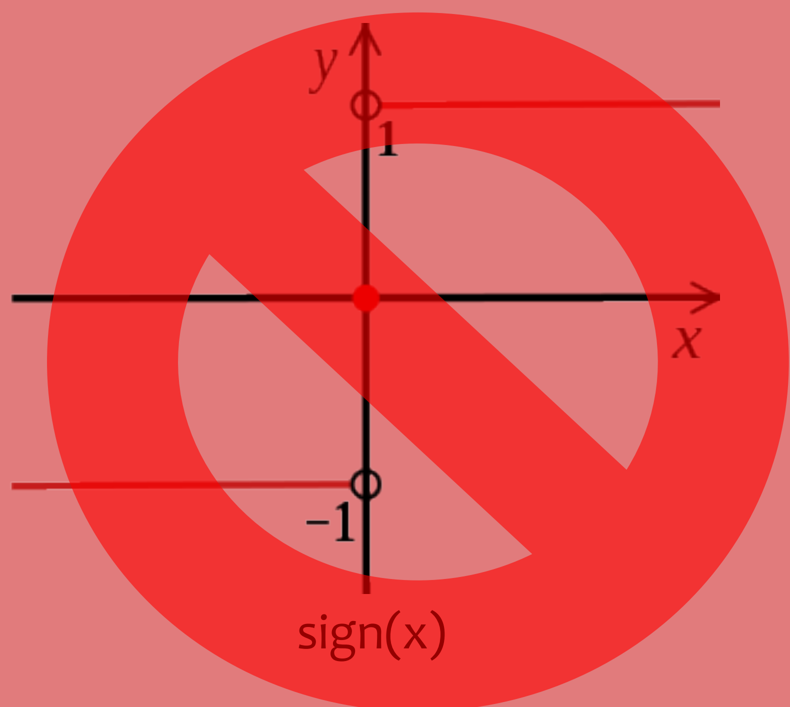


$$\text{logistic}(u) \equiv \frac{1}{1 + e^{-u}}$$

Using gradient ascent for linear classifiers

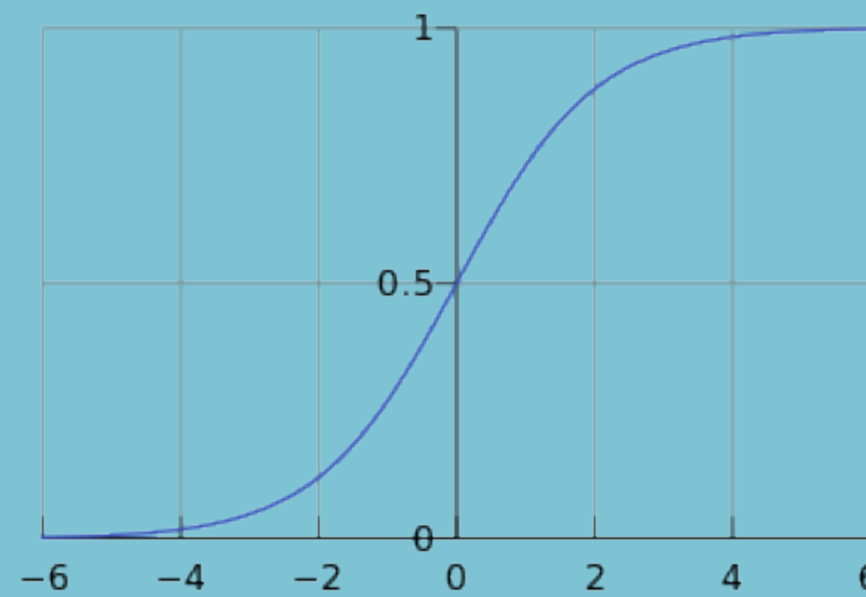
This decision function isn't differentiable:

$$h(\mathbf{x}) = \text{sign}(\boldsymbol{\theta}^T \mathbf{x})$$



Use a differentiable function instead:

$$p_{\boldsymbol{\theta}}(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})}$$



$$\text{logistic}(u) \equiv \frac{1}{1 + e^{-u}}$$

Logistic Regression

Data: Inputs are continuous vectors of length K. Outputs are discrete.

$$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N} \text{ where } \mathbf{x} \in \mathbb{R}^M \text{ and } y \in \{0, 1\}$$

Model: Logistic function applied to dot product of parameters with input vector.

$$p_{\boldsymbol{\theta}}(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})}$$

Learning: finds the parameters that minimize some objective function. $\boldsymbol{\theta}^* = \operatorname*{argmin}_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

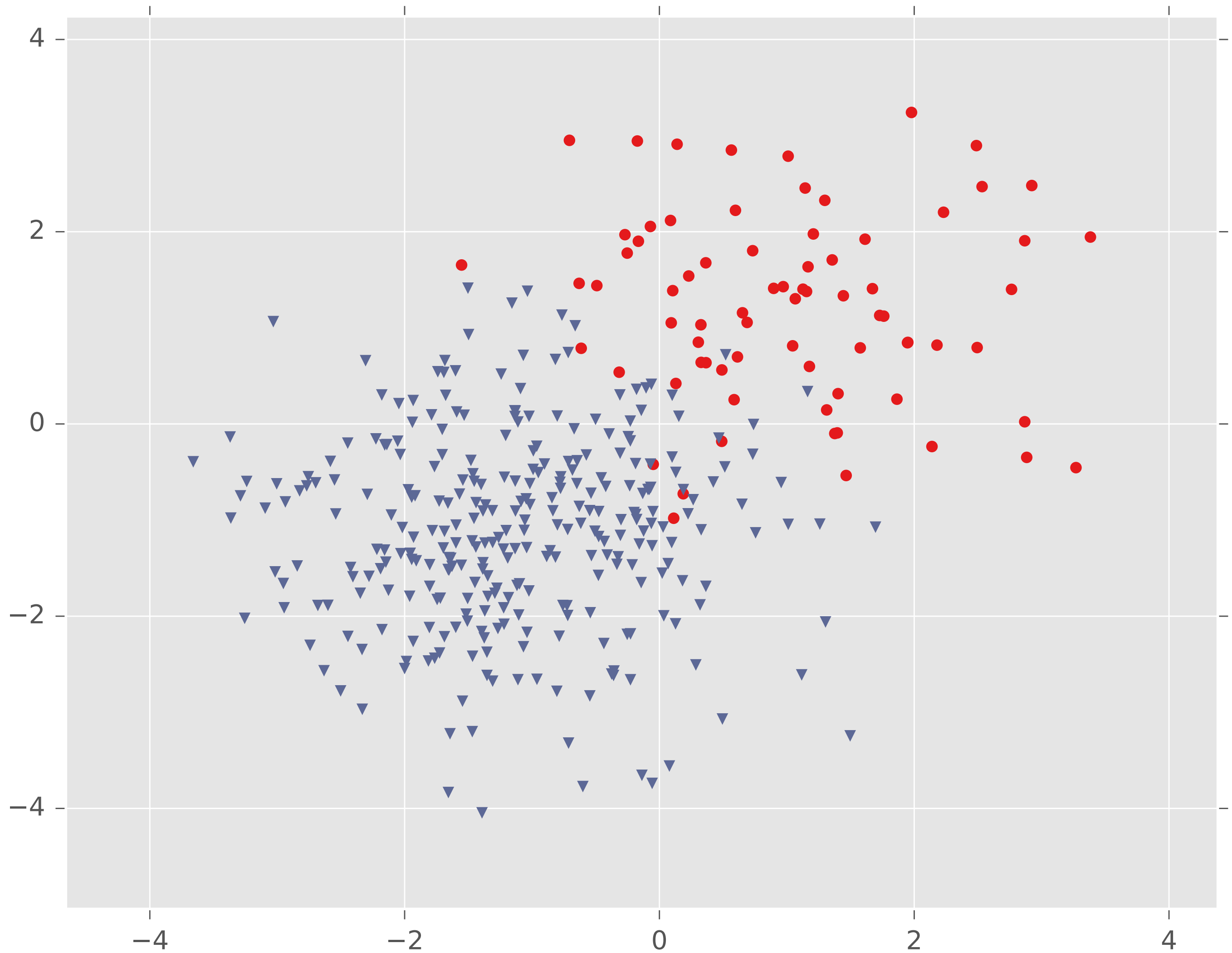
Prediction: Output is the most probable class.

$$\hat{y} = \operatorname*{argmax}_{y \in \{0,1\}} p_{\boldsymbol{\theta}}(y|\mathbf{x})$$

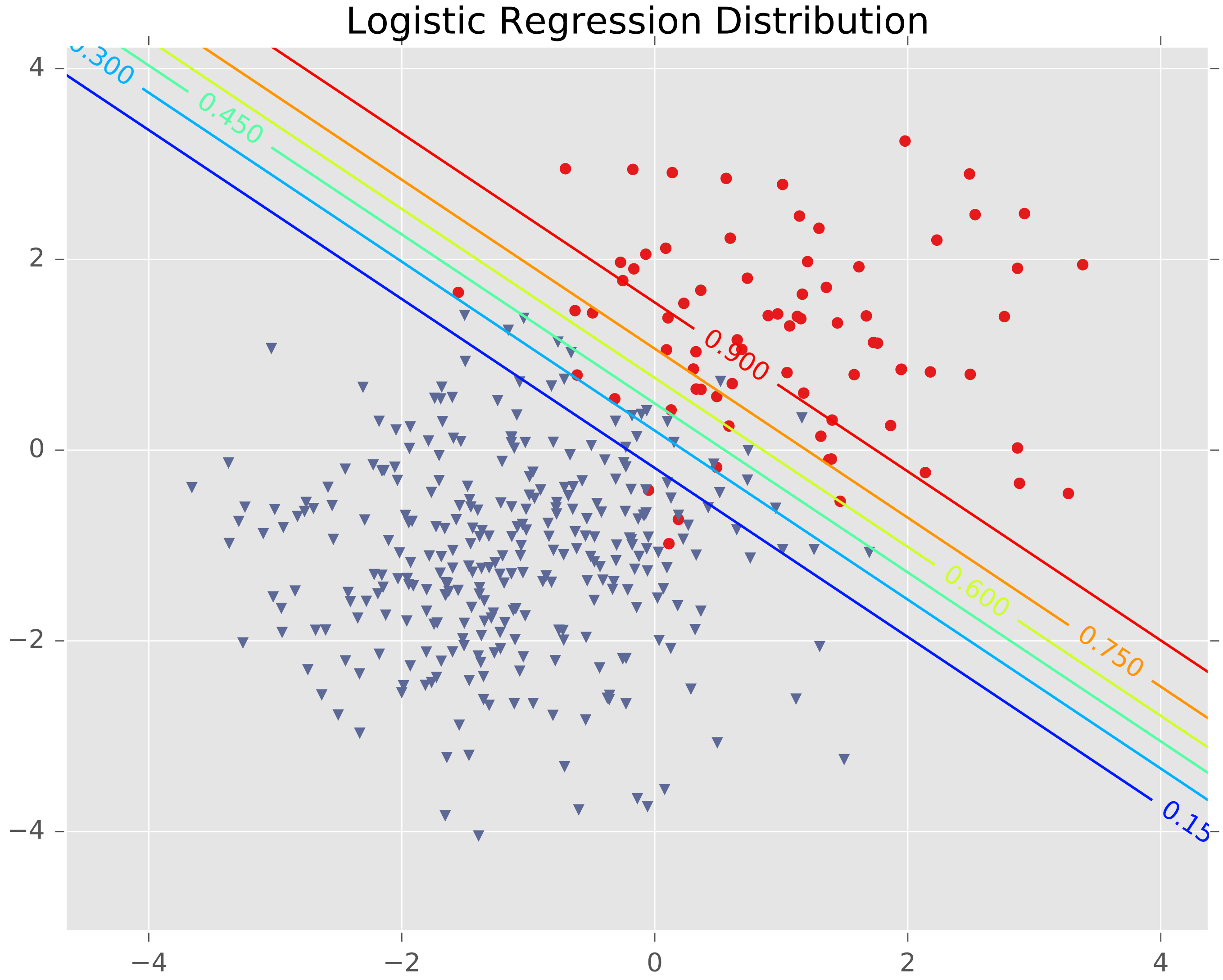
Whiteboard

- Decision boundary
- Bernoulli interpretation

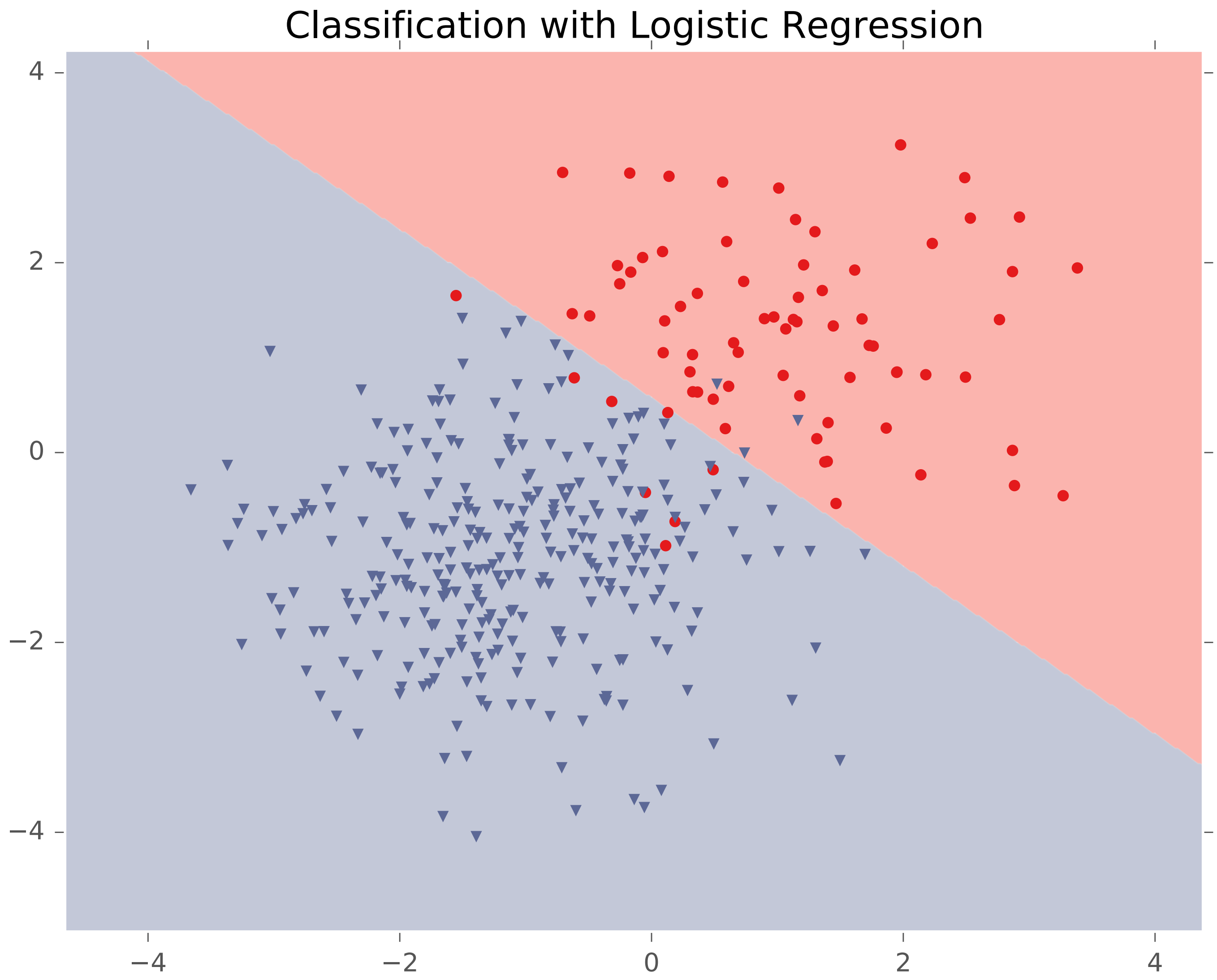
Logistic Regression



Logistic Regression



Logistic Regression



LEARNING LOGISTIC REGRESSION

Maximum **Conditional** Likelihood Estimation

Learning: finds the parameters that minimize some objective function.

$$\boldsymbol{\theta}^* = \operatorname*{argmin}_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

We minimize the *negative* log conditional likelihood:

$$J(\boldsymbol{\theta}) = -\log \prod_{i=1}^{N} p_{\boldsymbol{\theta}}(y^{(i)}|\mathbf{x}^{(i)})$$

Why?

1. We can't maximize likelihood (as in Naïve Bayes) because we don't have a joint model p(x,y)
2. It worked well for Linear Regression (least squares is MCLE)

Maximum **Conditional** Likelihood Estimation

Learning: Four approaches to solving $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\text{argmin}}\, J(\boldsymbol{\theta})$

Approach 1: Gradient Descent
(take larger – more certain – steps opposite the gradient)

Approach 2: Stochastic Gradient Descent (SGD)
(take many small steps opposite the gradient)

Approach 3: Newton's Method
(use second derivatives to better follow curvature)

Approach 4: Closed Form???
(set derivatives equal to zero and solve for parameters)

Maximum **Conditional** Likelihood Estimation

Learning: Four approaches to solving $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\text{argmin}}\, J(\boldsymbol{\theta})$

Approach 1: Gradient Descent
(take larger – more certain – steps opposite the gradient)

Approach 2: Stochastic Gradient Descent (SGD)
(take many small steps opposite the gradient)

Approach 3: Newton's Method
(use second derivatives to better follow curvature)

~~**Approach 4:** Closed Form???~~
~~(set derivatives equal to zero and solve for parameters)~~

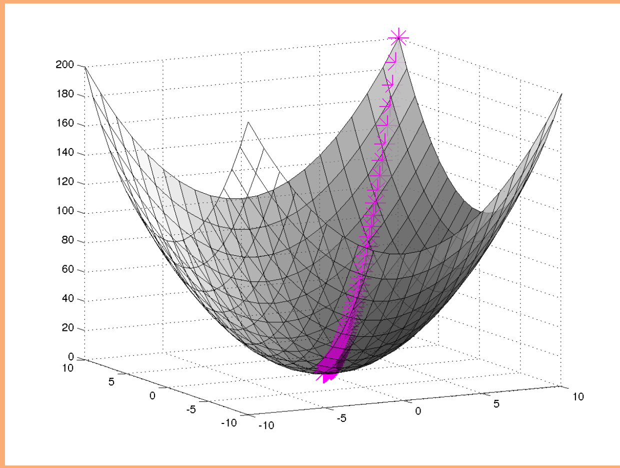
Logistic Regression does not have a closed form solution for MLE parameters.

Recall…

Gradient Descent

Algorithm 1 Gradient Descent

1: **procedure** GD($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
2: $\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
3: $\quad$**while** not converged **do**
4: $\quad\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \lambda \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$
5: $\quad$**return** $\boldsymbol{\theta}$



In order to apply GD to Logistic Regression all we need is the **gradient** of the objective function (i.e. vector of partial derivatives).

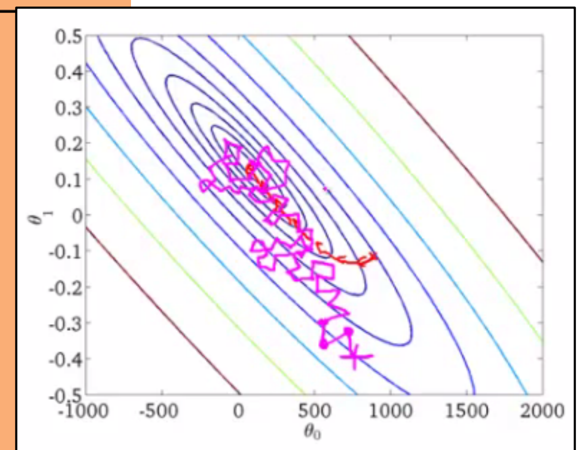
$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \begin{bmatrix} \frac{d}{d\theta_1} J(\boldsymbol{\theta}) \\ \frac{d}{d\theta_2} J(\boldsymbol{\theta}) \\ \vdots \\ \frac{d}{d\theta_M} J(\boldsymbol{\theta}) \end{bmatrix}$$

Stochastic Gradient Descent (SGD)

Recall...

Algorithm 1 Stochastic Gradient Descent (SGD)

procedure SGD($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
  while not converged **do**
    for $i \in$ shuffle($\{1, 2, \ldots, N\}$) **do**
      $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \lambda \nabla_{\boldsymbol{\theta}} J^{(i)}(\boldsymbol{\theta})$
  return θ



We can also apply SGD to solve the MCLE problem for Logistic Regression.

We need a per-example objective:

$$\text{Let } J(\boldsymbol{\theta}) = \textstyle\sum_{i=1}^{N} J^{(i)}(\boldsymbol{\theta})$$
$$\text{where } J^{(i)}(\boldsymbol{\theta}) = -\log p_{\boldsymbol{\theta}}(y^i | \mathbf{x}^i).$$

GRADIENT FOR LOGISTIC REGRESSION

Whiteboard

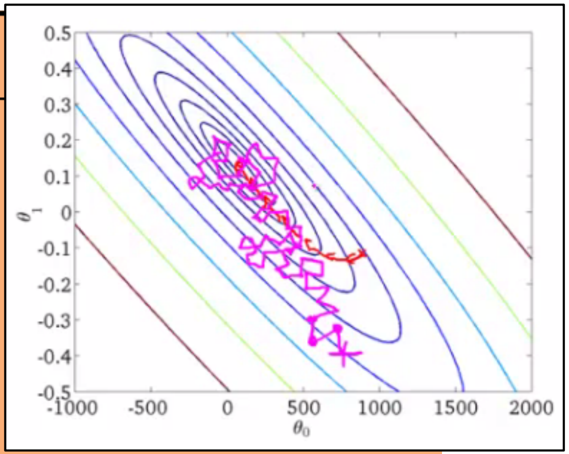
- Partial derivative for Logistic Regression
- Gradient for Logistic Regression

Details: Picking learning rate

- Use grid-search in log-space over small values on a tuning set:
  - e.g., 0.01, 0.001, …
- Sometimes, decrease after each pass:
  - e.g factor of $1/(1 + dt)$, t=epoch
  - sometimes $1/t^2$
- Fancier techniques I won't talk about:
  - Adaptive gradient: scale gradient differently for each dimension (Adagrad, ADAM, ….)

Slide courtesy of William Cohen

SGD for Logistic Regression



Algorithm 1 SGD for Logistic Regression

procedure SGD($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
  while not converged **do**
    for $i \in$ shuffle($\{1, 2, \ldots, N\}$) **do**
      $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \lambda(y^{(i)} - \rho^{(i)})\mathbf{x}^{(i)}$
      where $\rho^{(i)} := 1/(1 + \exp(-\boldsymbol{\theta}^T \mathbf{x}))$
  return $\boldsymbol{\theta}$

We can also apply SGD to solve the MCLE problem for Logistic Regression.

We need a per-example objective:

$$\text{Let } J(\boldsymbol{\theta}) = \textstyle\sum_{i=1}^{N} J^{(i)}(\boldsymbol{\theta})$$
$$\text{where } J^{(i)}(\boldsymbol{\theta}) = -\log p_{\boldsymbol{\theta}}(y^i | \mathbf{x}^i).$$

Takeaways

1. Discriminative classifiers directly model the **conditional**, $p(y|x)$
2. Logistic regression is a **simple linear classifier**, that retains a **probabilistic** semantics
3. Parameters in LR are learned by **iterative optimization** (e.g. SGD)

NON-LINEAR FEATURES

Nonlinear Features

Whiteboard

- Example functions
- Nonlinear Features for Linear Regression
- Nonlinear Features for Logistic Regression
- Nonlinear Features for KNN
- Nonlinear Features for Naïve Bayes

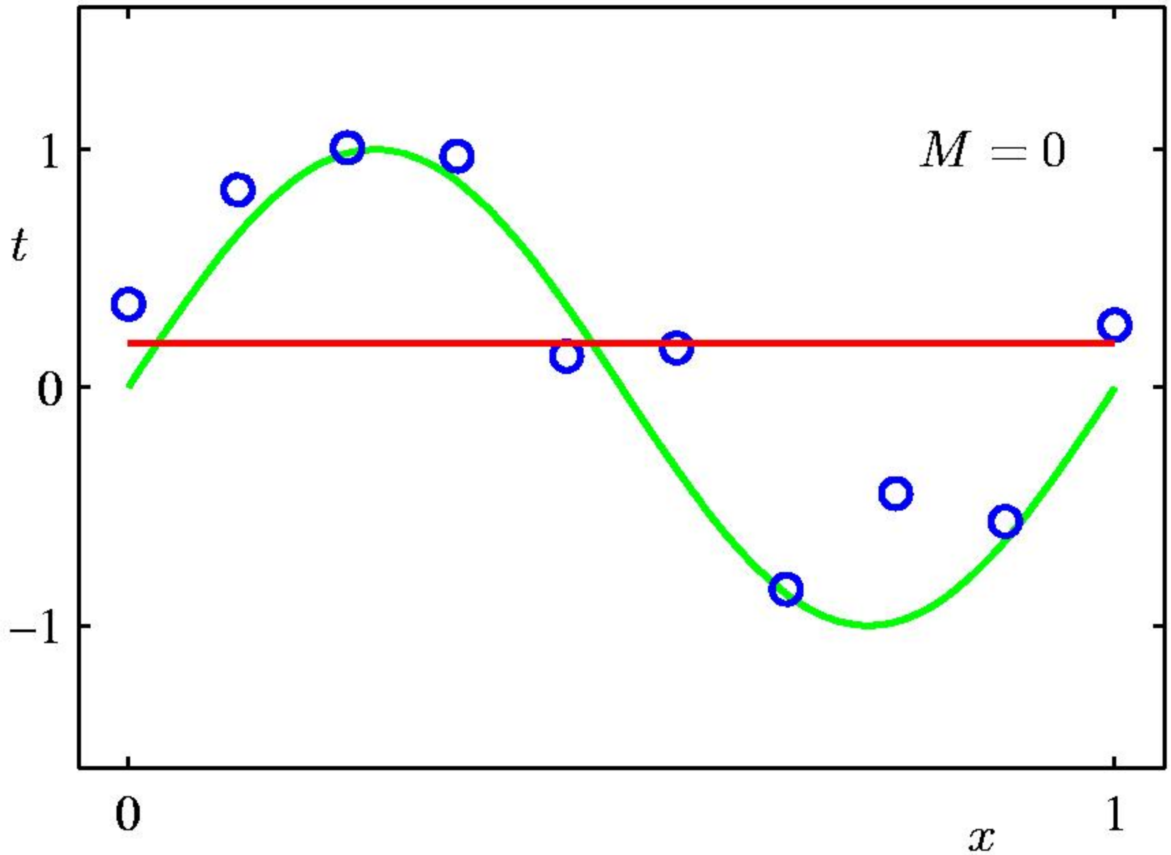
Example: Linear Regression Nonlinear Features

Polynomial basis vectors on a small dataset

– From Bishop Ch 1

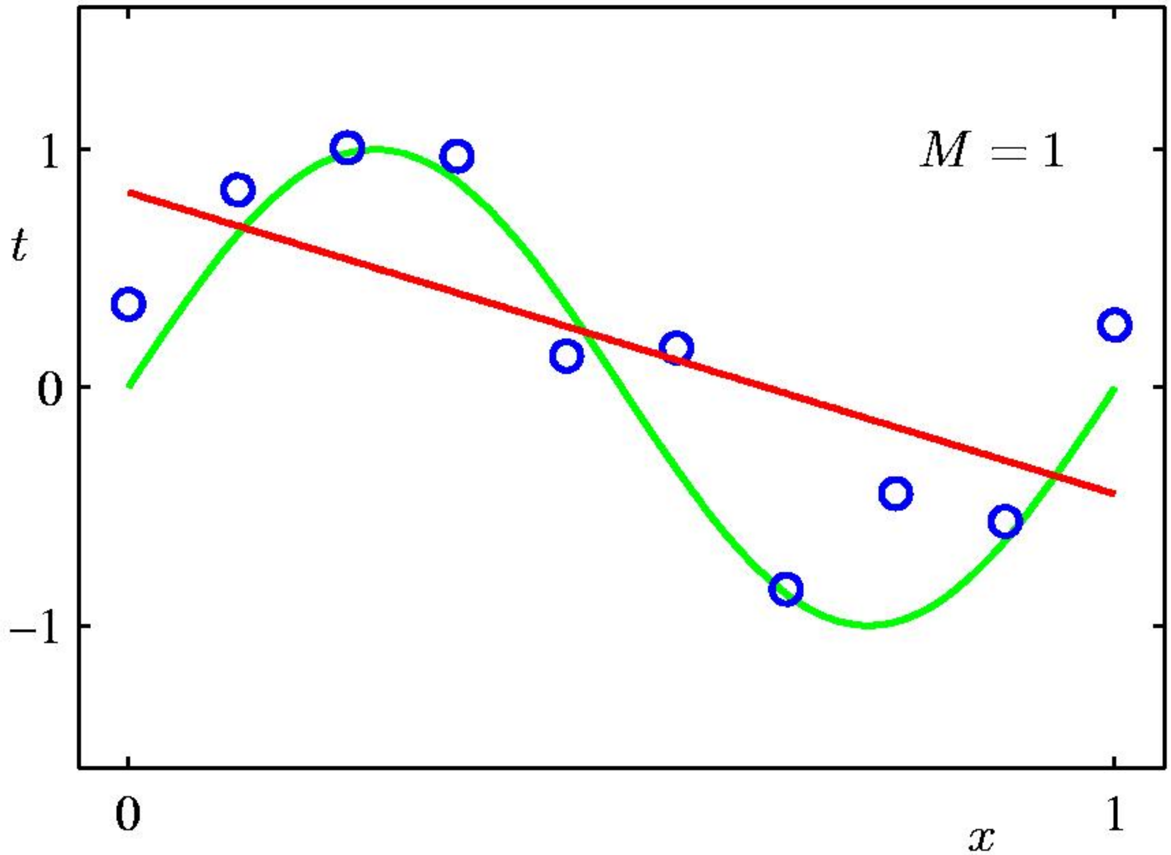
Slide courtesy of William Cohen

0^{th} Order Polynomial



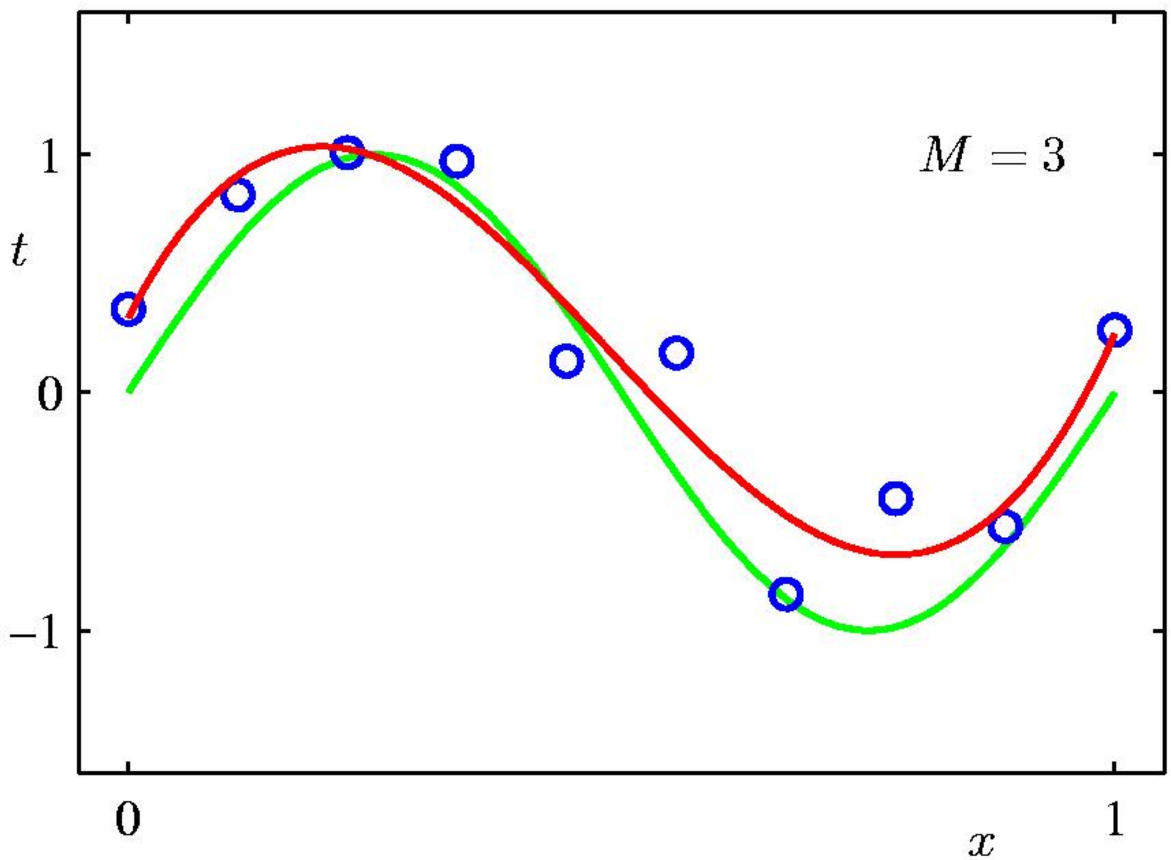
n=10

1st Order Polynomial



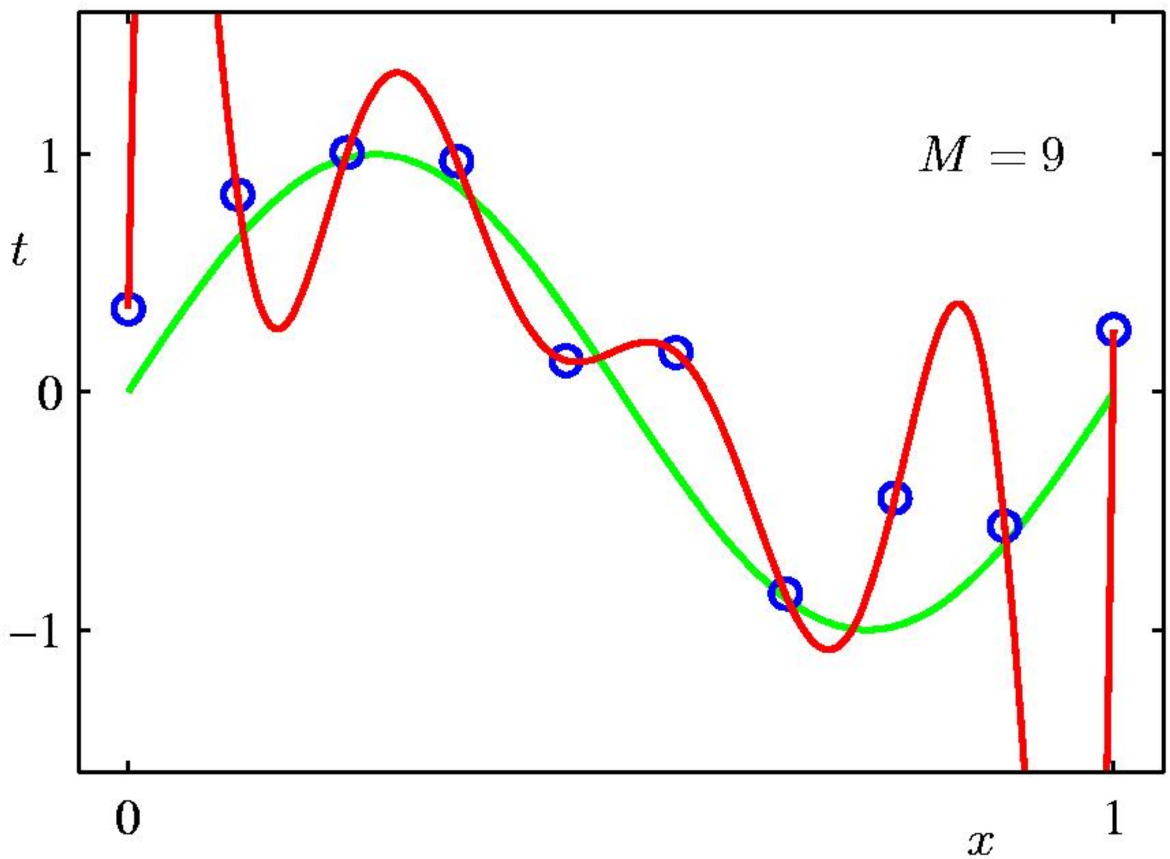
Slide courtesy of William Cohen

3^{rd} Order Polynomial



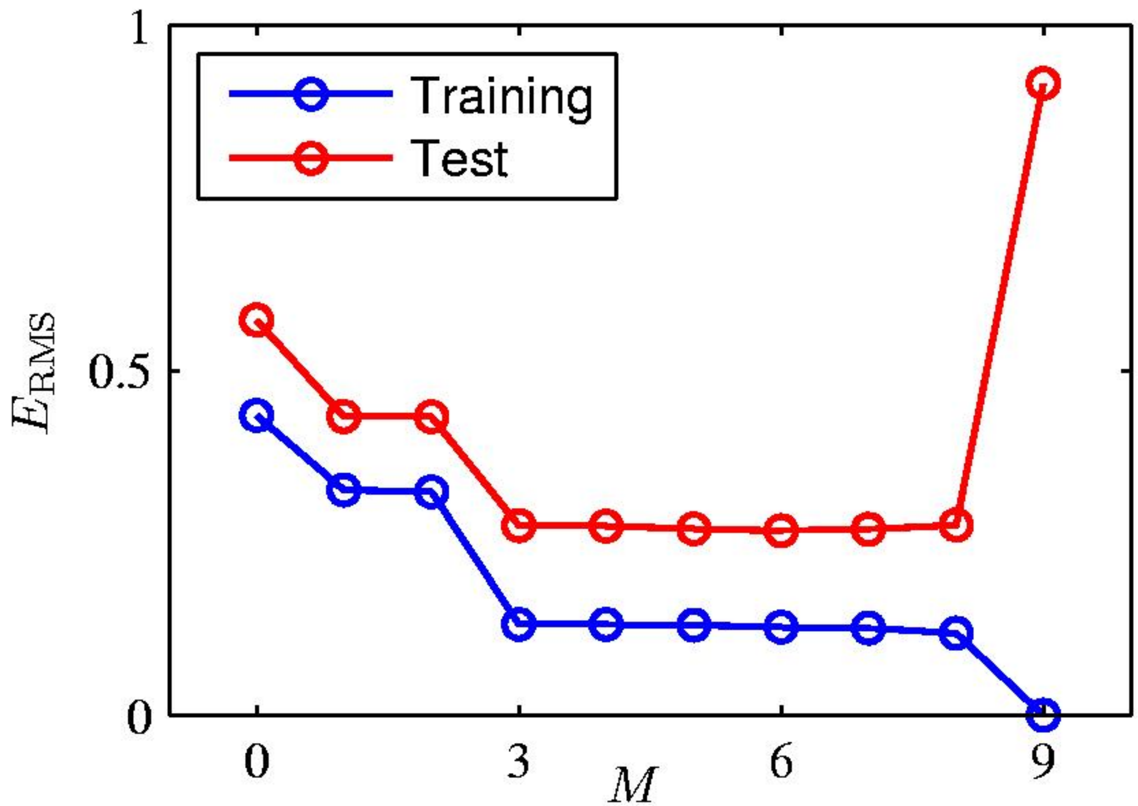
Slide courtesy of William Cohen

9^{th} Order Polynomial



Slide courtesy of William Cohen

Over-fitting



Root-Mean-Square (RMS) Error: $E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$

Slide courtesy of William Cohen

Polynomial Coefficients

	$M=0$	$M=1$	$M=3$	$M=9$
θ_0	0.19	0.82	0.31	0.35
θ_1		-1.27	7.99	232.37
θ_2			-25.43	-5321.83
θ_3			17.37	48568.31
θ_4				-231639.30
θ_5				640042.26
θ_6				-1061800.52
θ_7				1042400.18
θ_8				-557682.99
θ_9				125201.43

Slide courtesy of William Cohen

Overfitting

Definition: The problem of **overfitting** is when the model captures the noise in the training data instead of the underlying structure

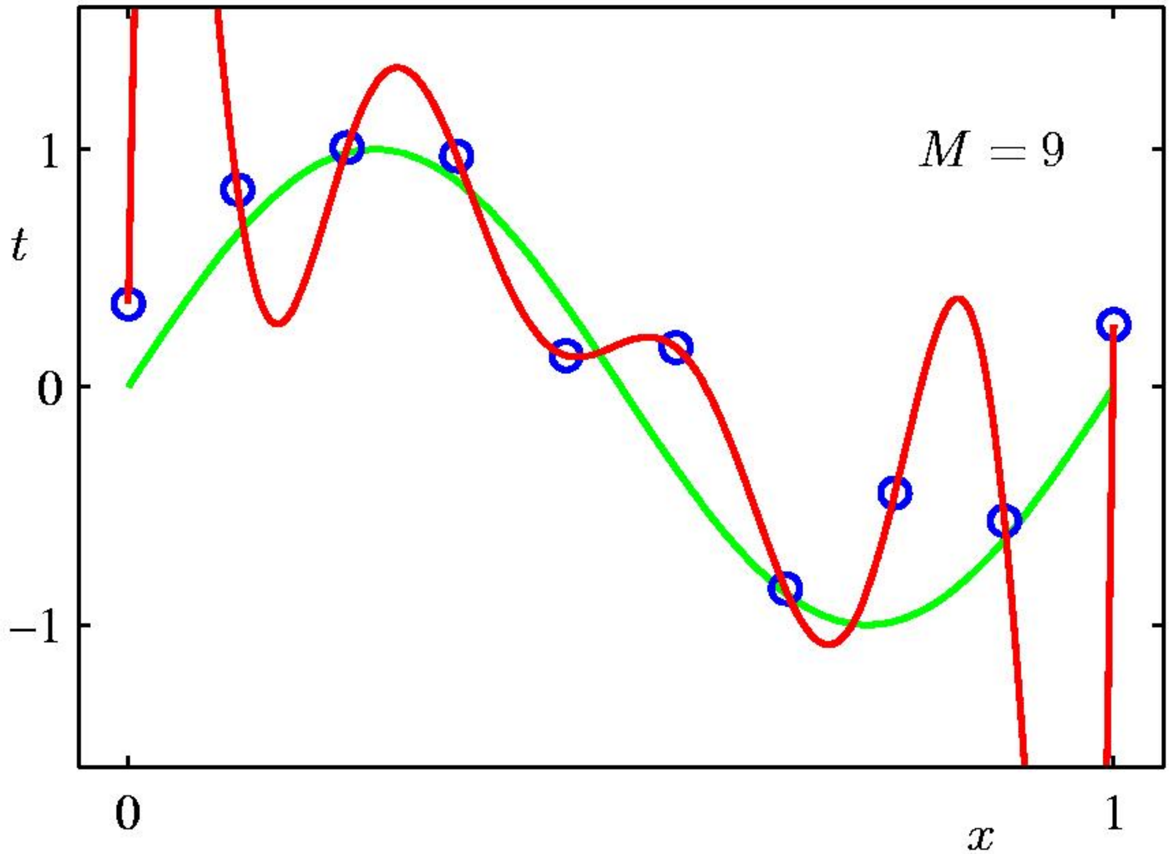
Overfitting can occur in all the models we've seen so far:

- KNN (e.g. when k is small)
- Naïve Bayes (e.g. without a prior)
- Linear Regression (e.g. with basis function)
- Logistic Regression (e.g. with many rare features)

9^{th} Order Polynomial

(Small # of examples)

$$N = 10$$

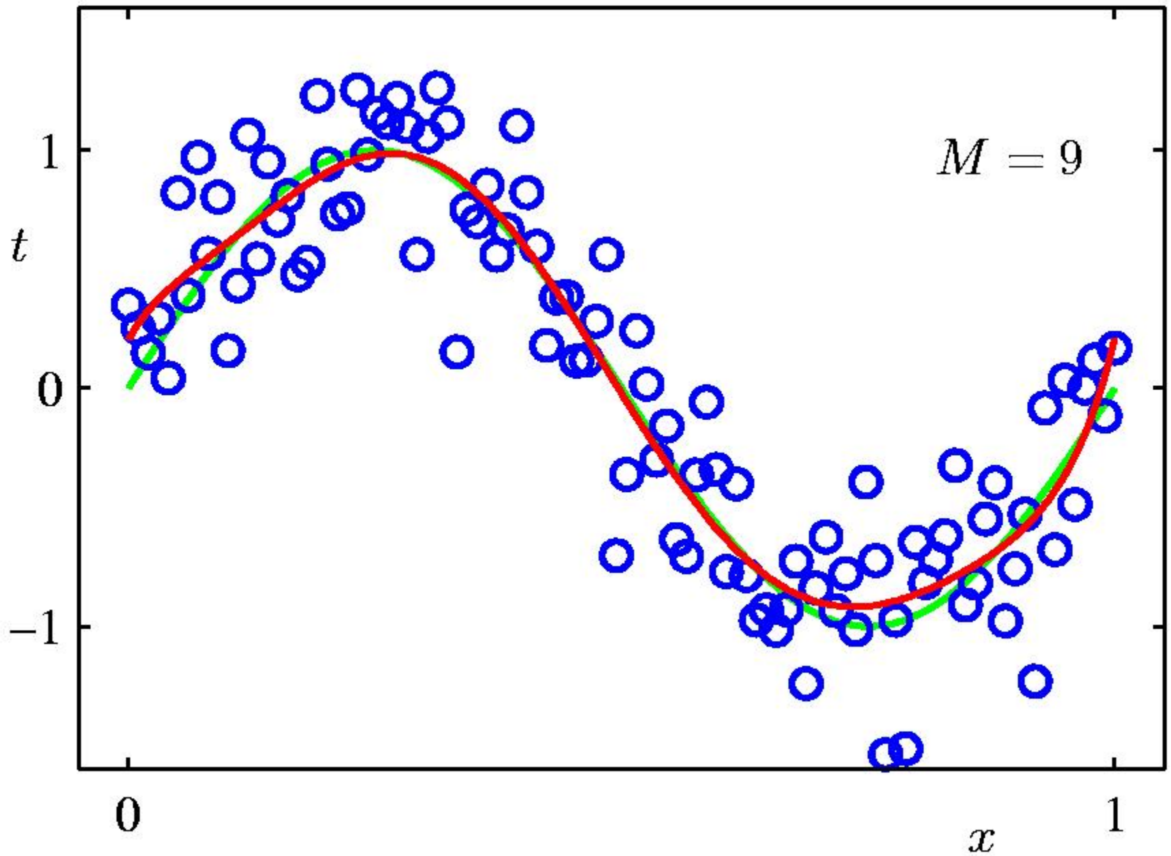


Slide courtesy of William Cohen

9th Order Polynomial

(Large # of examples)

$$N = 100$$



Slide courtesy of William Cohen