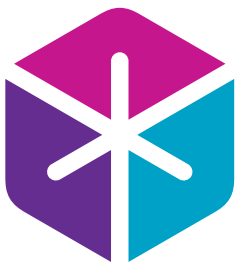
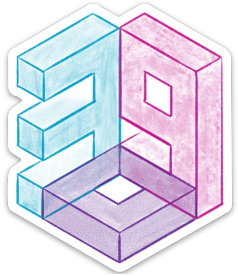



https://introml.mit.edu/

6.390 Intro to Machine Learning

Lecture 5: Features

Shen Shen

March 1, 2024

(some slides adapted from Tamara Broderick and Phillip Isola)

Midterm exam heads-up

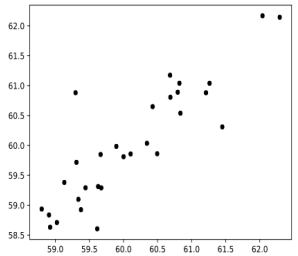
- Wednesday, March 20, 730pm-930pm. Everyone will be assigned an exam room.
- For conflict and/or accommodations, please be sure to email us by Wednesday, **March 6**, at **6.390-personal@mit.edu** .
- Midterm will cover Week 1 till Week 6 (neural networks) materials.
- We will use the regular lecture time/room on March 15 (11am-12pm in 10-250) for midterm review session (the session will be recorded).
- More details (your exam room, practice exams, exam policy, etc.) will be posted on introML homepage this weekend, along with the typical weekly announcements.

Outline

- Recap (linear regression and classification)
- Systematic feature transformations
  - Polynomial features
- Domain-dependent, or goal-dependent, encoding
  - Numerical features
    - Standardizing the data
  - Categorical features
    - One-hot encoding
    - Factored encoding
    - Thermometer encoding

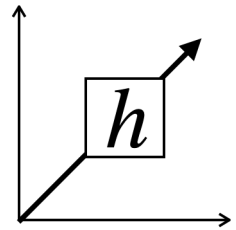
Training

Data



ML algorithm

Hypothesis class
Hyperparameters
If/how to add regularization
Objective (loss) functions

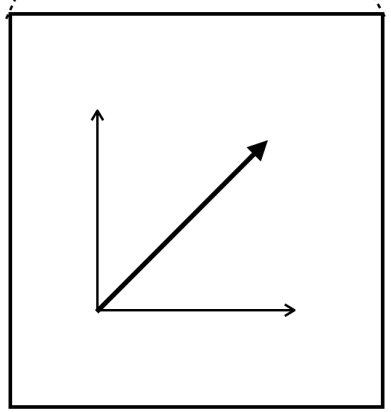


Recap:

- OLS can have analytical formula and "easy" prediction mechanism
- Regularization
- Cross-validation
- Gradient descent

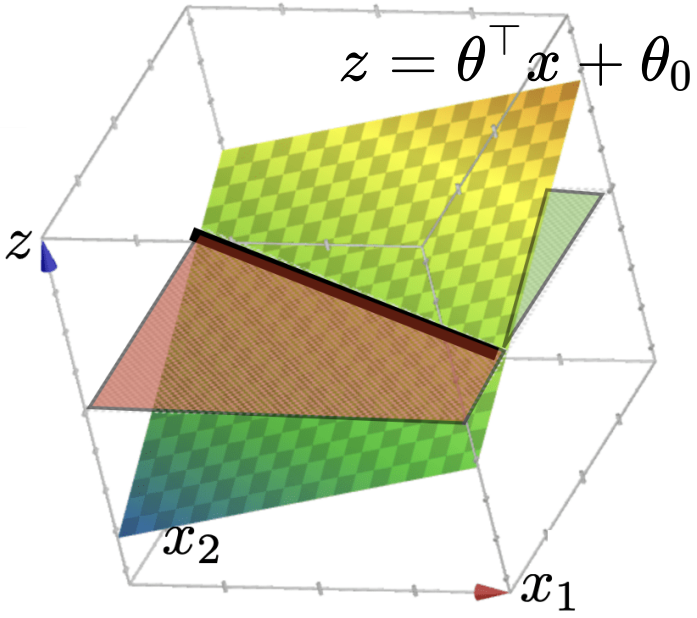
Testing (predicting)

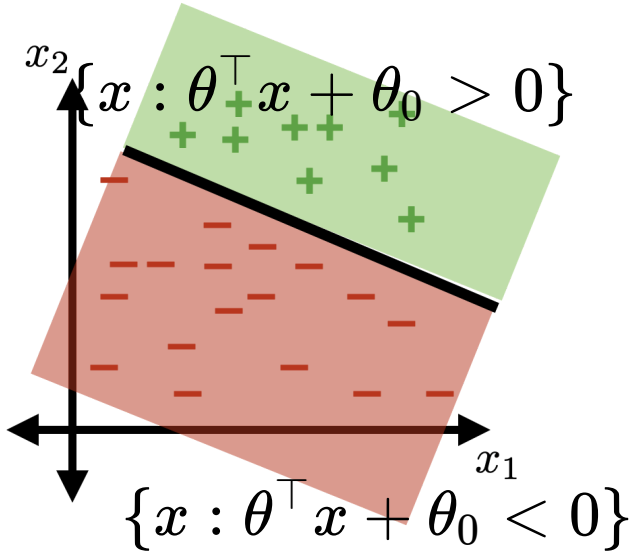
new input x



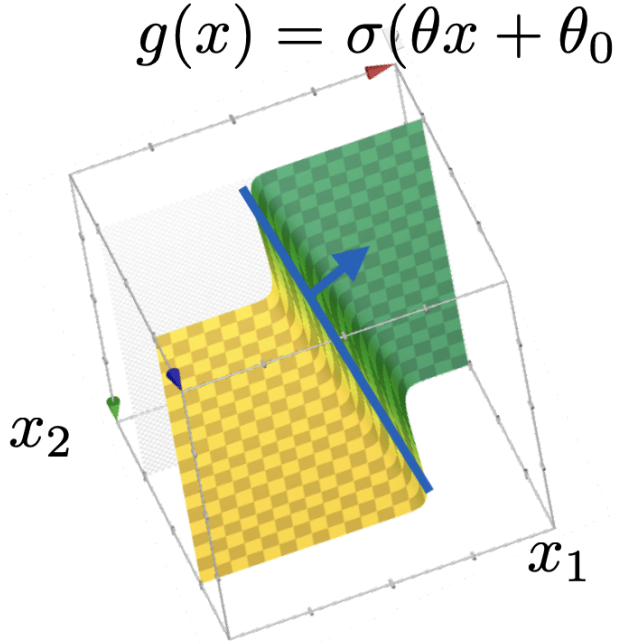
new prediction y

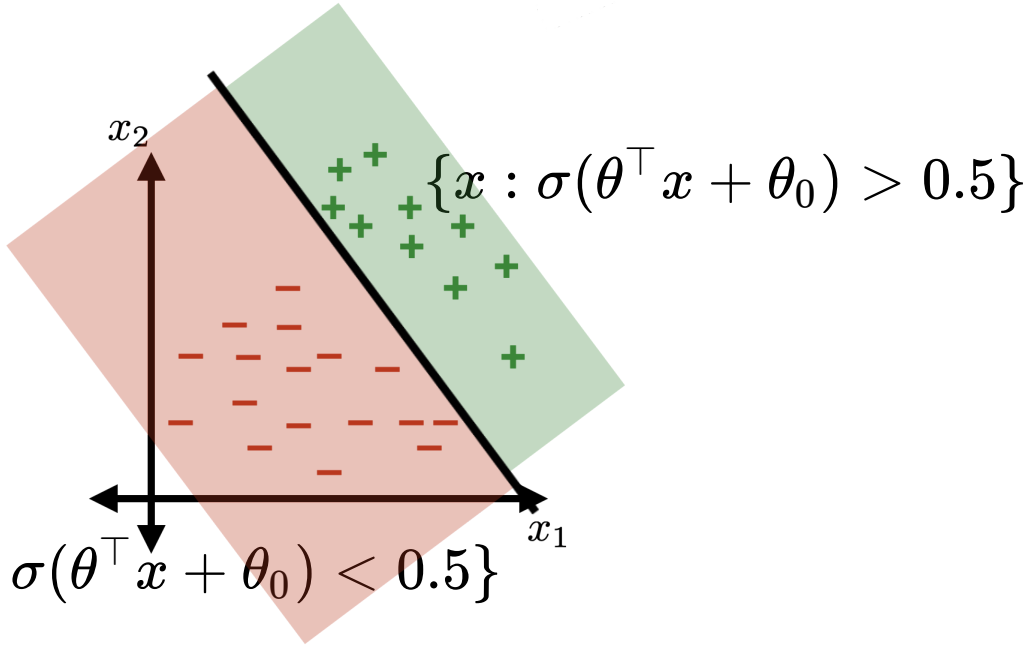
(vanilla, sign-based) linear classifier

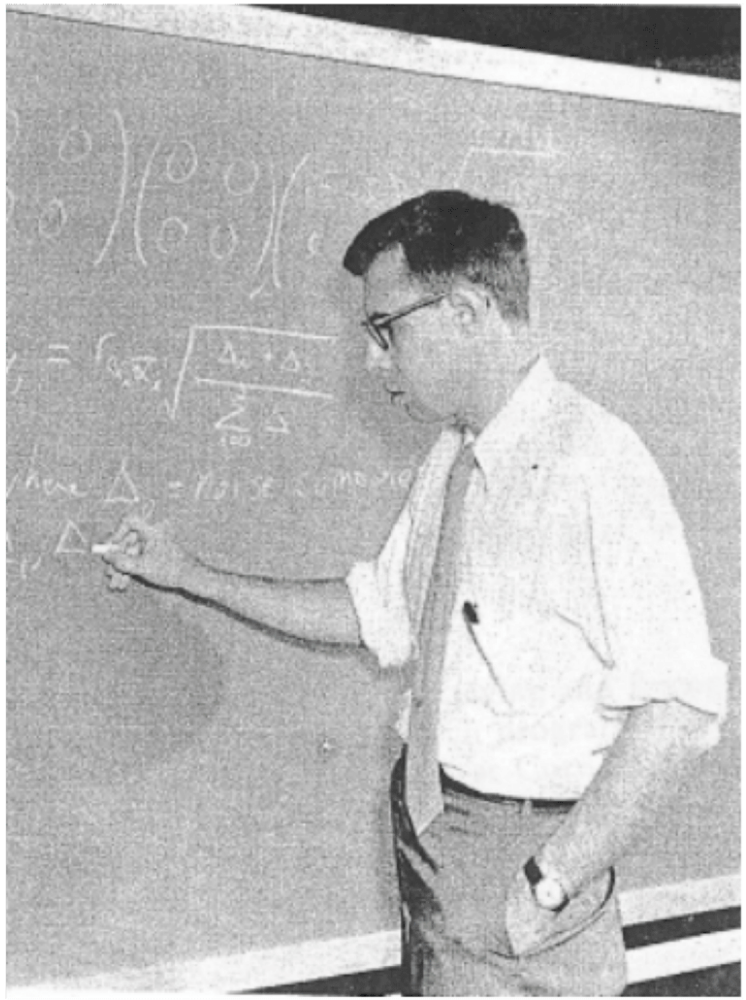




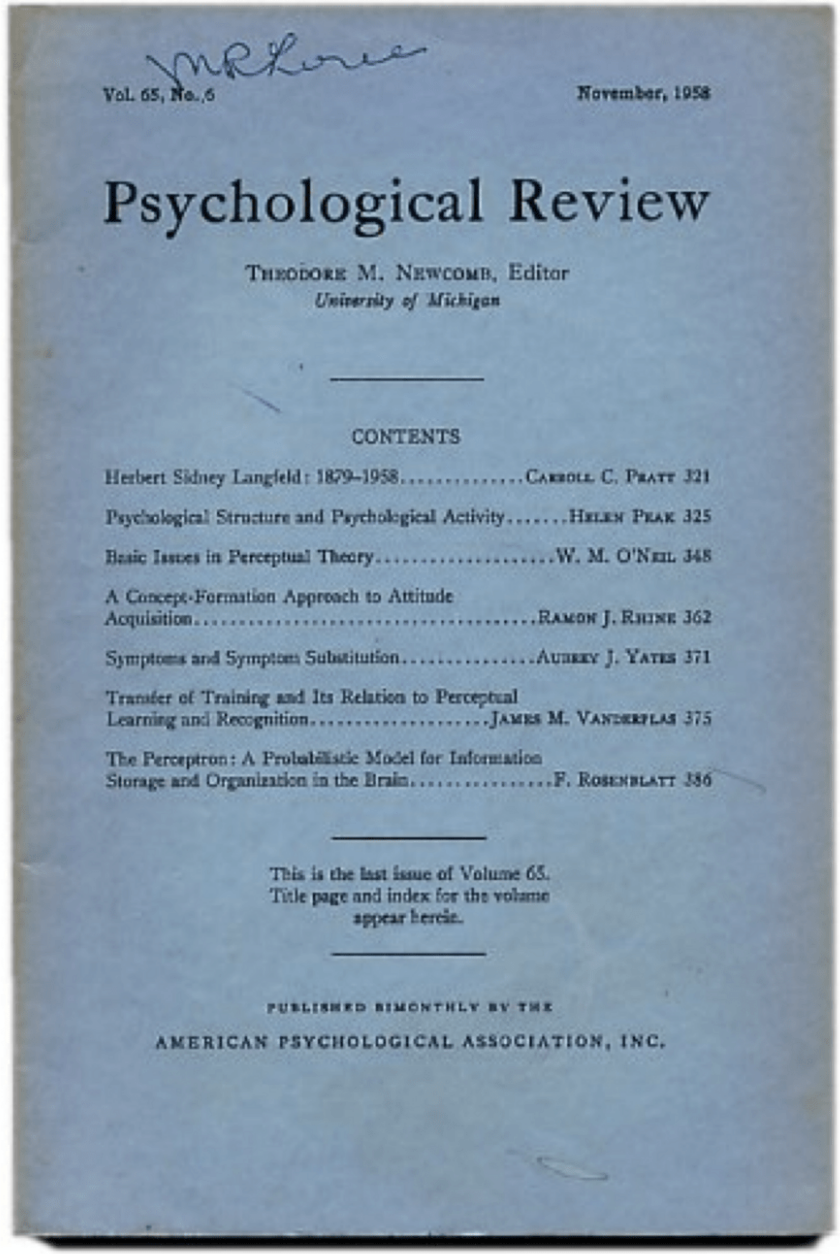
linear logistic regression (classifier)







http://www.ecse.rpi.edu/homepages/nagy/PDF_chrono/2011_Nagy_Pace_FR.pdf. Photo by George Nagy



Vol. 65, No. 6 November, 1958

Psychological Review

THEODORE M. NEWCOMB, Editor
University of Michigan

CONTENTS

Herbert Sidney Langfeld: 1879–1958 ..... CARROLL C. PRATT 321

Psychological Structure and Psychological Activity ..... HELEN PEAK 325

Basic Issues in Perceptual Theory ..... W. M. O'NEIL 348

A Concept-Formation Approach to Attitude Acquisition ..... RAMON J. RHINE 362

Symptoms and Symptom Substitution ..... AUBREY J. YATES 371

Transfer of Training and Its Relation to Perceptual Learning and Recognition ..... JAMES M. VANDERPLAS 375

The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain ..... F. ROSENBLATT 386

This is the last issue of Volume 65.
Title page and index for the volume appear herein.

PUBLISHED BIMONTHLY BY THE
AMERICAN PSYCHOLOGICAL ASSOCIATION, INC.

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.335.3398&rep=rep1&type=pdf

1958 New York Times...

NEW NAVY DEVICE LEARNS BY DOING

Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser

WASHINGTON, July 7 (UPI) —The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

The embryo—the Weather Bureau's $2,000,000 "704" computer—learned to differentiate between right and left after fifty attempts in the Navy's demonstration for newsmen.

The service said it would use this principle to build the first of its Perceptron thinking machines that will be able to read and write. It is expected to be finished in about a year at a cost of $100,000.

Dr. Frank Rosenblatt, designer of the Perceptron, conducted the demonstration. He said the machine would be the first device to think as the human brain. As do human beings, Perceptron will make mistakes at first, but will grow wiser as it gains experience, he said.

Dr. Rosenblatt, a research psychologist at the Cornell Aeronautical Laboratory, Buffalo, said Perceptrons might be fired to the planets as mechanical space explorers.

Without Human Controls

The Navy said the perceptron would be the first non-living mechanism "capable of receiving, recognizing and identifying its surroundings without any human training or control."

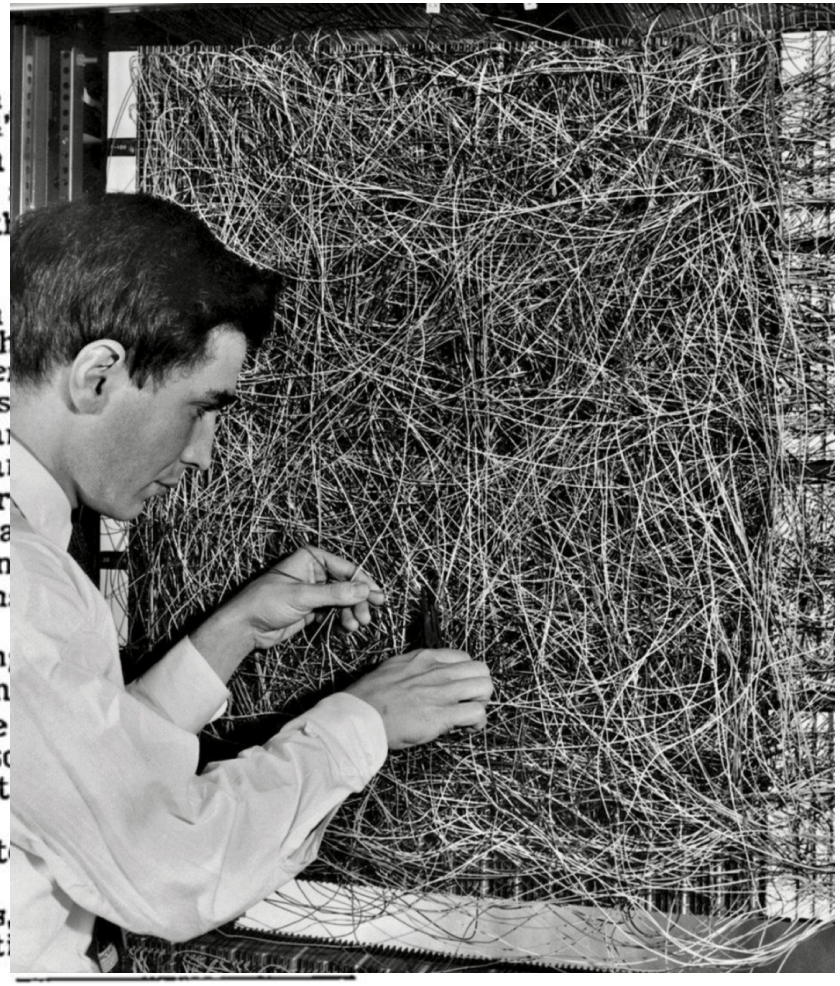
The "brain" is designed to remember images and information it has perceived itself. Ordinary computers remember only what is fed into them on punch cards or magnetic tape.

Later Perceptrons will be able to recognize people and call out their names and instantly translate speech in one language to speech or writing in another language, it was predicted.

Mr. Rosenblatt said in principle it would be possible to build brains that could reproduce themselves on an assembly line and which would be conscious of their existence.

In
"704"
with
side
on t

In
mach
twee
regis
squa
squa
Dr
expla
learn
term
had
chan
Th
have
"ass
elect
like
phot
has
cells
necti



An aside:

- Geometrical understanding of algebraic objects are fundamental to engineering.
- Certainly contributed to many ML algorithms, and continue to influence/inspire new ideas.

The idea of "distance" appeared in

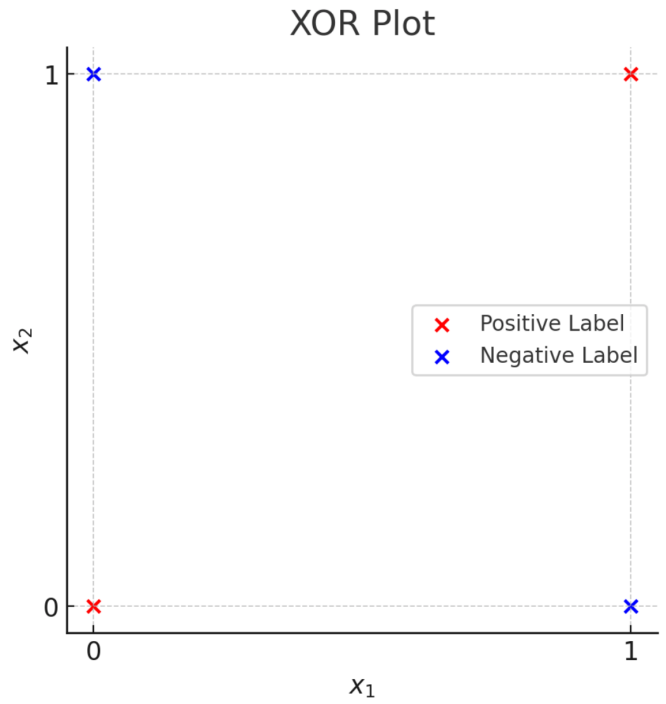
- Linear regression (MSE)
- Logistic regression (data points further from the separator are classified with higher confidence)

it will play a central role in later weeks

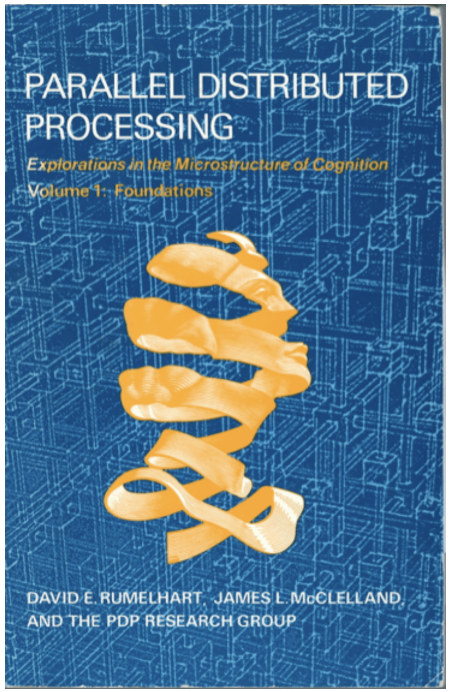
- Nearest neighbor (non-parametric models for supervised learning)
- Clustering (unsupervised learning)

it will play a central role in fundamental algorithms we won't discuss:

- Perceptron
- Support vector machine



- Not linearly separable.
- Proposed by Minsky and Papert, 1970s
- Caused the first AI winder.



- Parallel Distributed Processing (PDP), 1986
- Pointed out key ideas (enabling neural networks):
  - Nonlinear feature transformation
  - "Stacking" transformations
  - Backpropogation

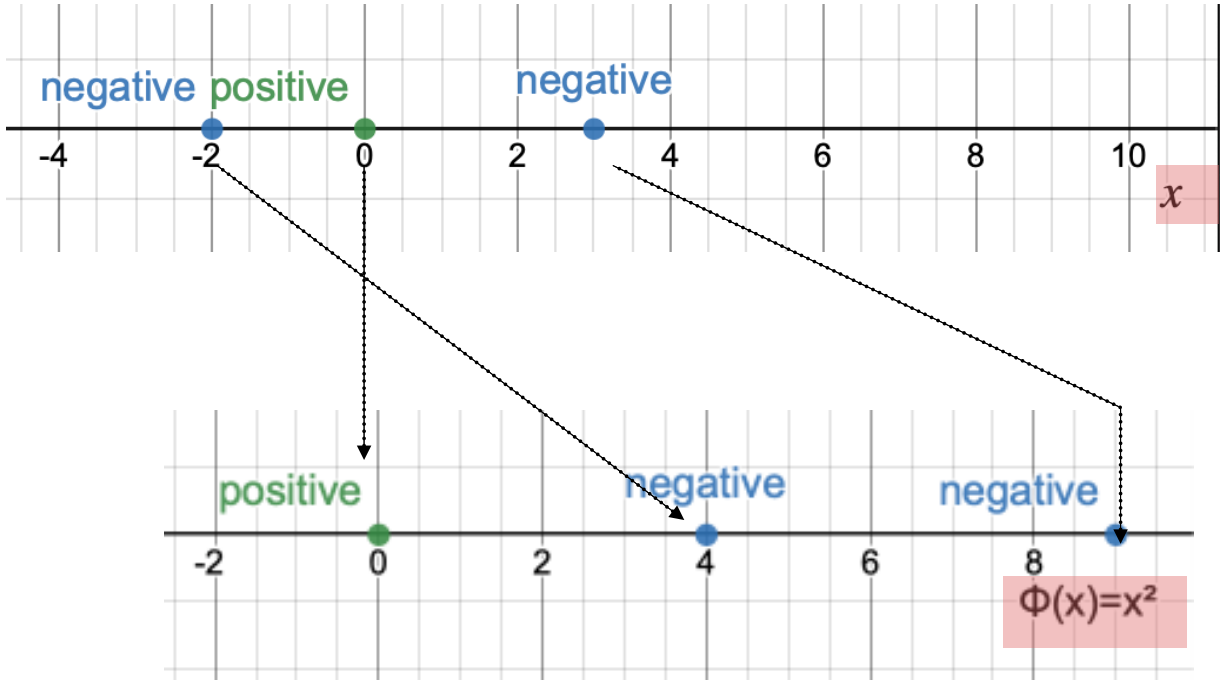
} (next week)

Outline

- Recap (linear regression and classification)
- Systematic feature transformations
  - Polynomial features
  - Other typical fixed feature transformations
- Domain-dependent, or goal-dependent, encoding
  - Numerical features
  - Categorical features
    - One-hot encoding
    - Factored encoding
    - Thermometer encoding

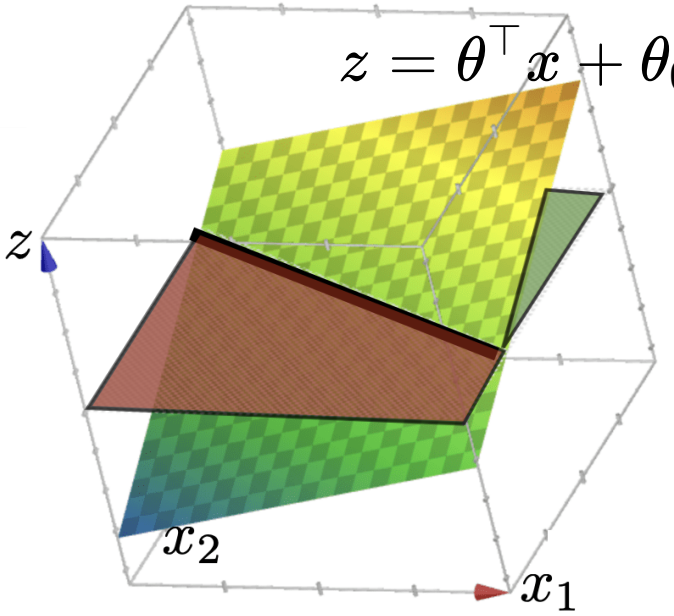
Polynomial features for classification

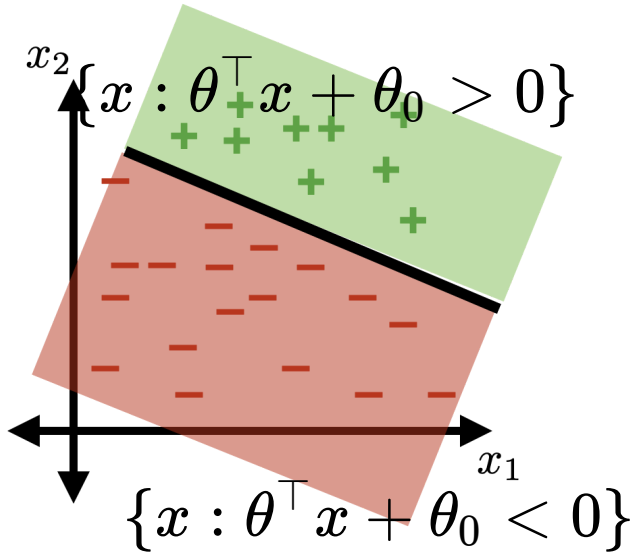
- Not linearly separable in x space



- Linearly separable in $\Phi(x) = x^2$ space (e.g., $\text{sign}(1.5 - \Phi(x))$ is one such perfectly-separating classifier)

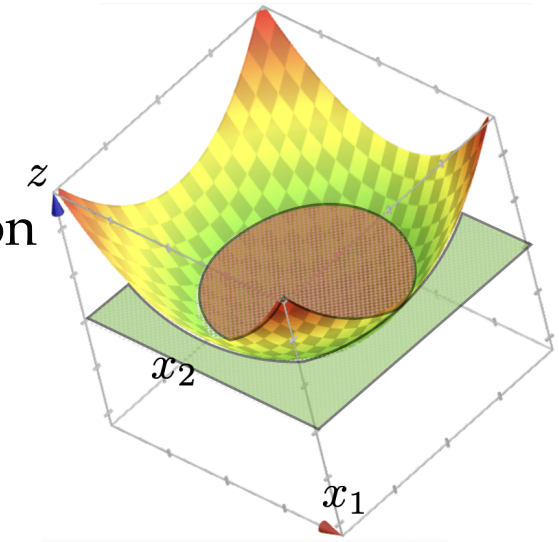
(vanilla, sign-based) linear classifier

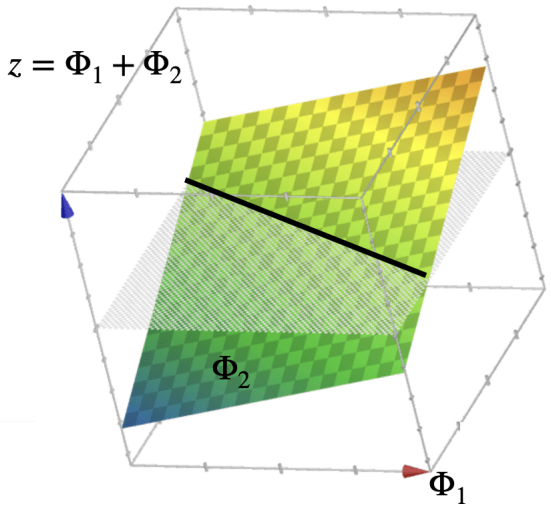


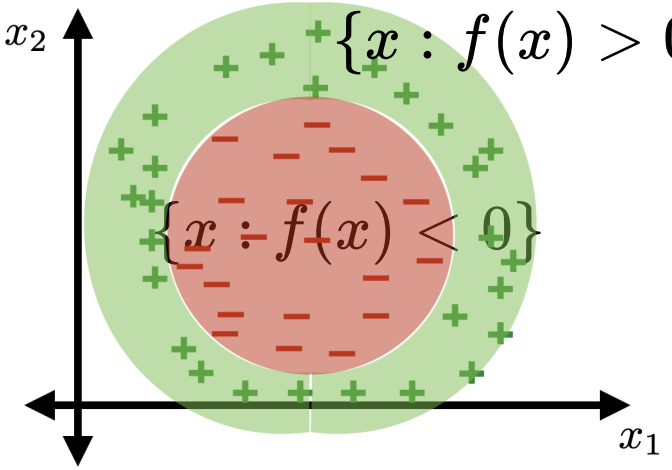


using polynomial feature transformation

$$z = f(\Phi(x))$$
$$= \Phi_1 + \Phi_2$$
$$= x_1^2 + x_2^2$$





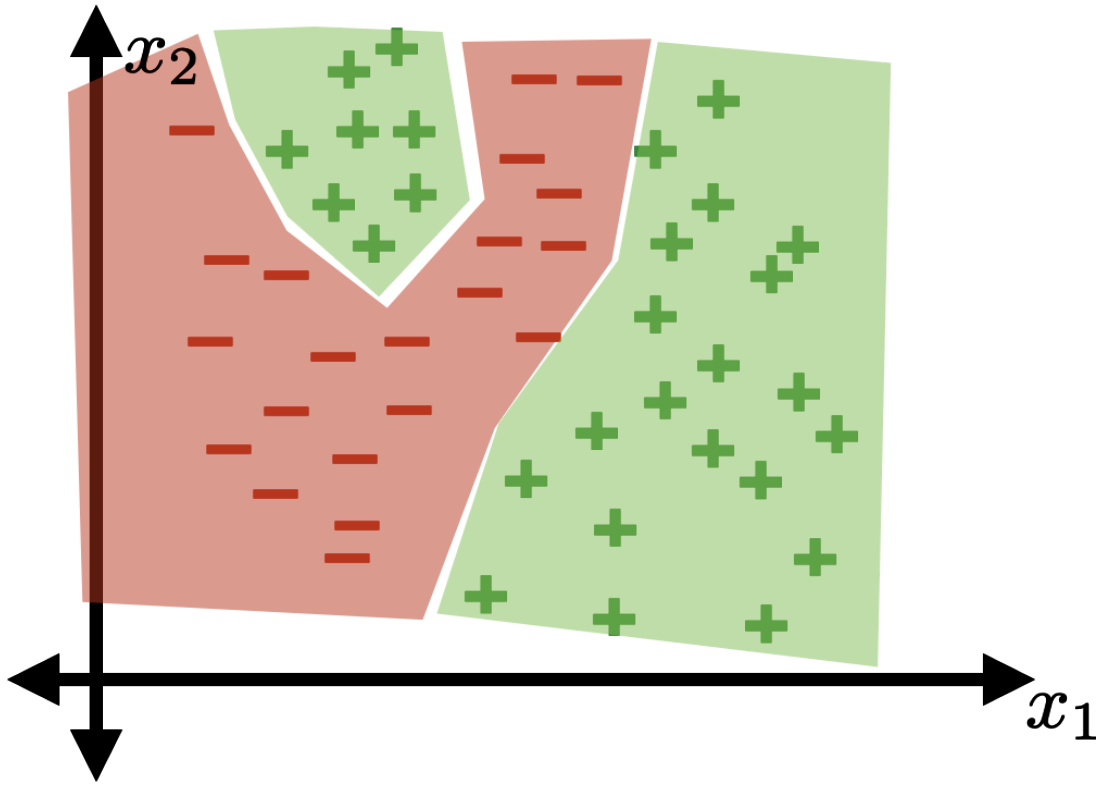


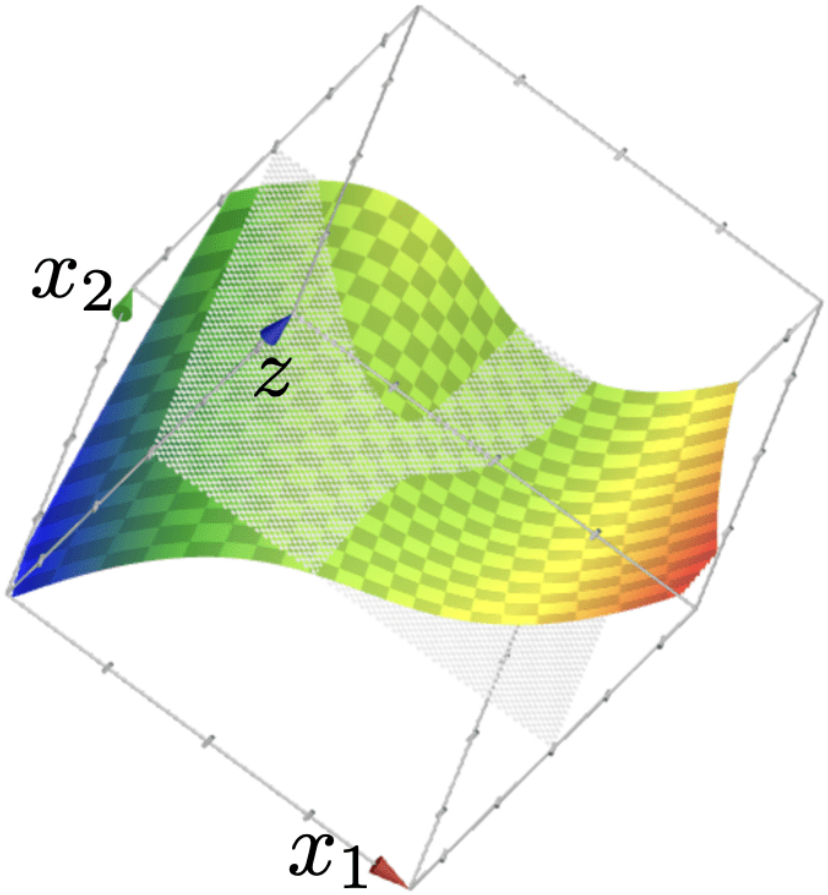
Polynomial Basis Construction

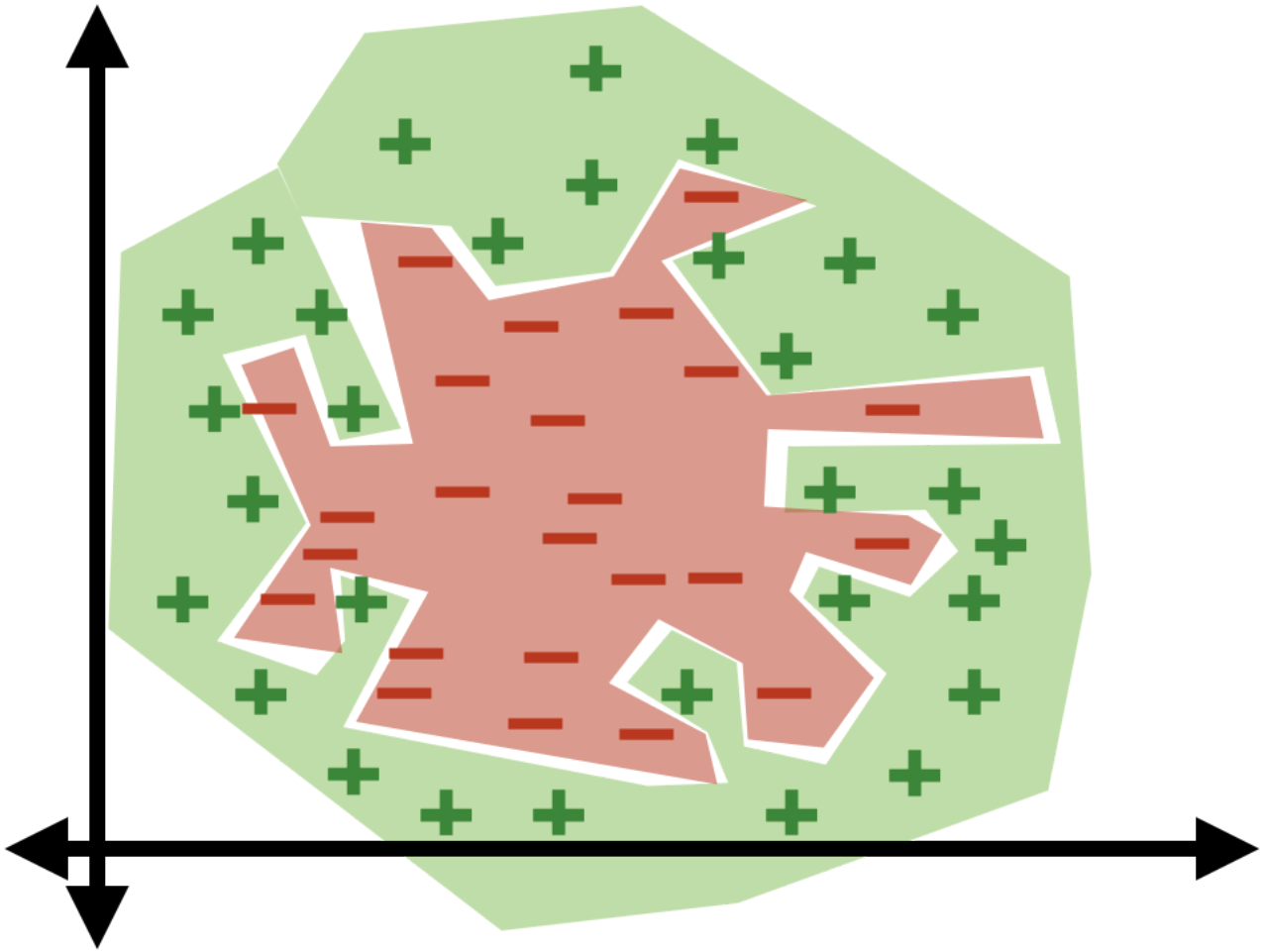
order (k)	Elements in basis when d=1	Elements in basis for d>1
0	$[1]$	$[1]$
1	$[1, x_1]$	$[1, x_1, \ldots, x_d]$
2	$[1, x_1, x_1^2]$	$[1, x_1, \ldots, x_d, x_1^2, x_1 x_2, \ldots, x_{d-1} x_d, x_d^2]$
3	$[1, x_1, x_1^2, x_1^3]$	$[1, x_1, \ldots, x_d, x_1^2, x_1 x_2, \ldots, x_{d-1} x_d, x_d^2, x_1^3, x_1^2 x_2, x_1 x_2 x_3, \ldots, x_d^3]$

- Elements in the basis are the monomials of ("original features" raised up to power k)
- With a given d and k, the basis is **fixed**.

Using polynomial features of order 3



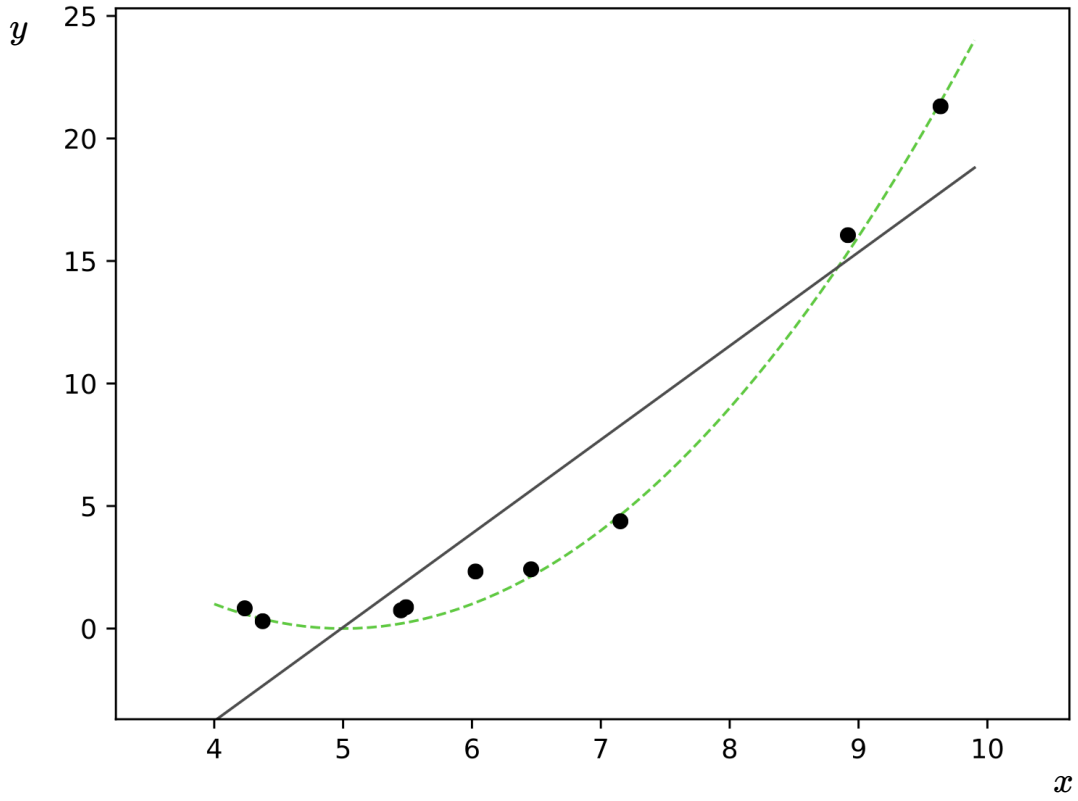




- Using high-order polynomial features, we can get very "nuanced" decision boundary.
- Training error is 0!
- But seems like our classifier is overfitting.
- Tension between richness/expressiveness of hypothesis class and generalization.

Polynomial features for regression

k = 1

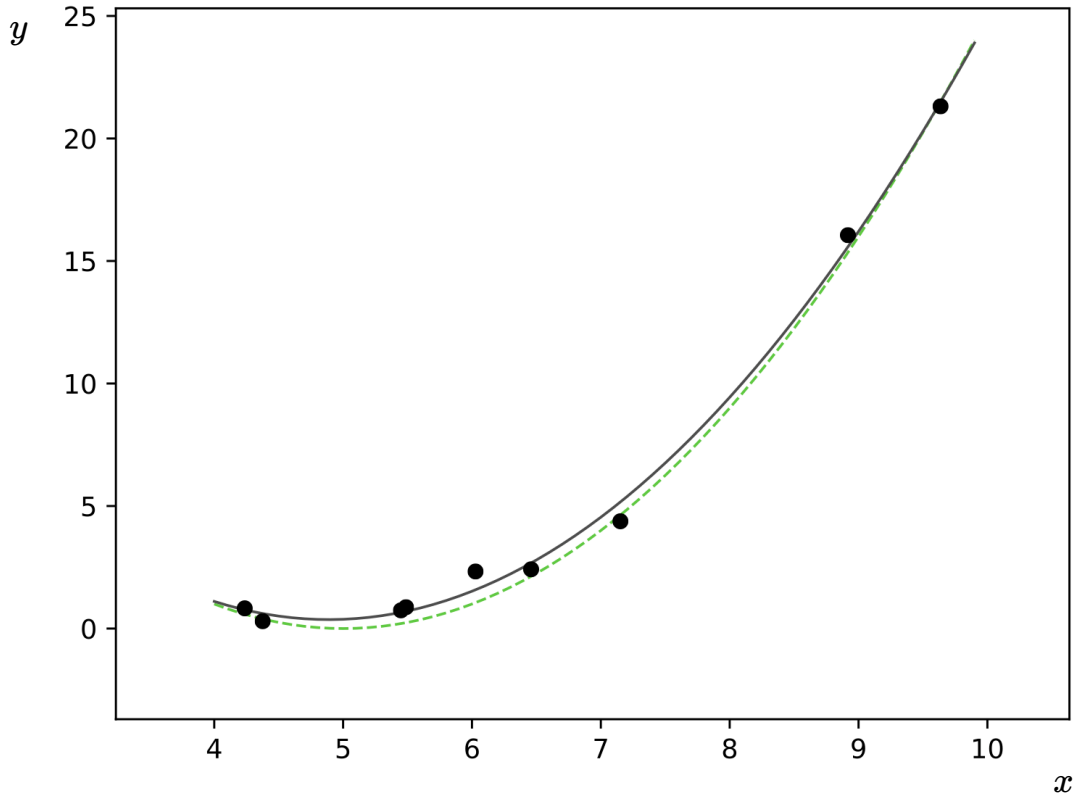


- 9 data points.
- Each data has one-dimensional feature $x \in \mathbb{R}$
- Label $y \in \mathbb{R}$

- Choose $k = 1$
- $h_\theta(x) = \theta_0 + \theta_1 x$
- How many scalar parameters to learn?
- 2 scalar parameters

Polynomial features for regression

k = 2

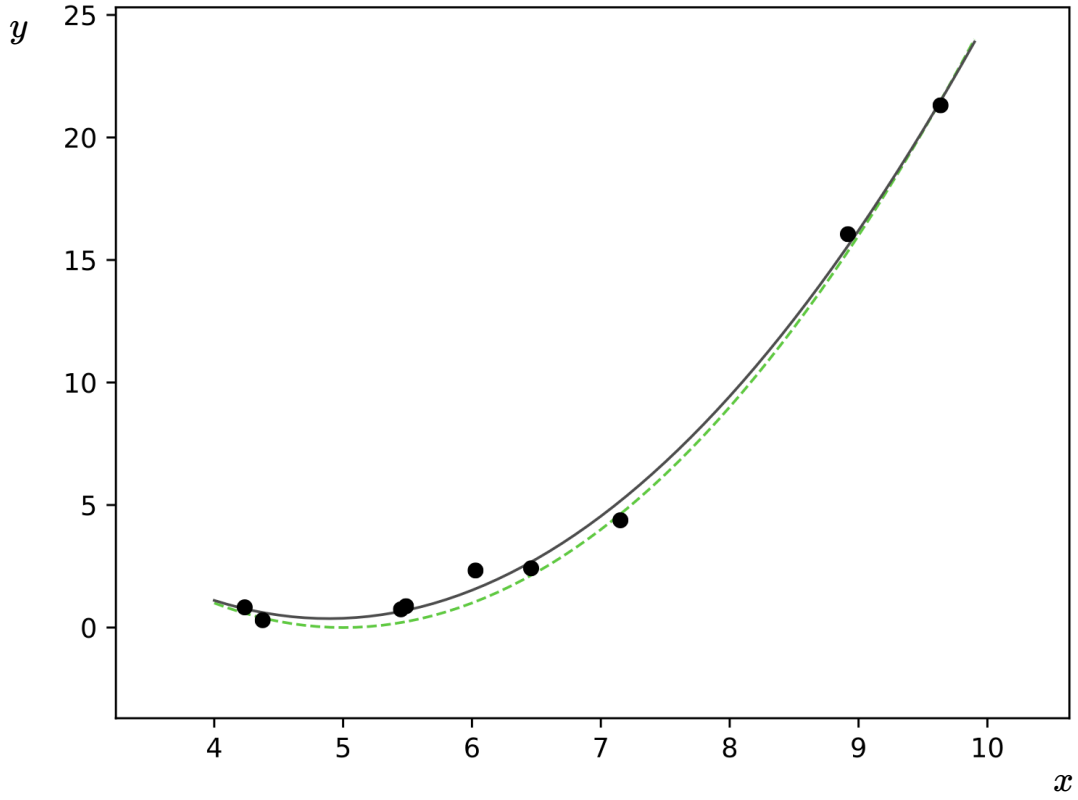


- 9 data points.
- Each data has one-dimensional feature $x \in \mathbb{R}$
- Label $y \in \mathbb{R}$

- Choose $k = 2$
- $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$
- How many scalar parameters to learn?
- 3 scalar parameters

Polynomial features for regression

k = 3

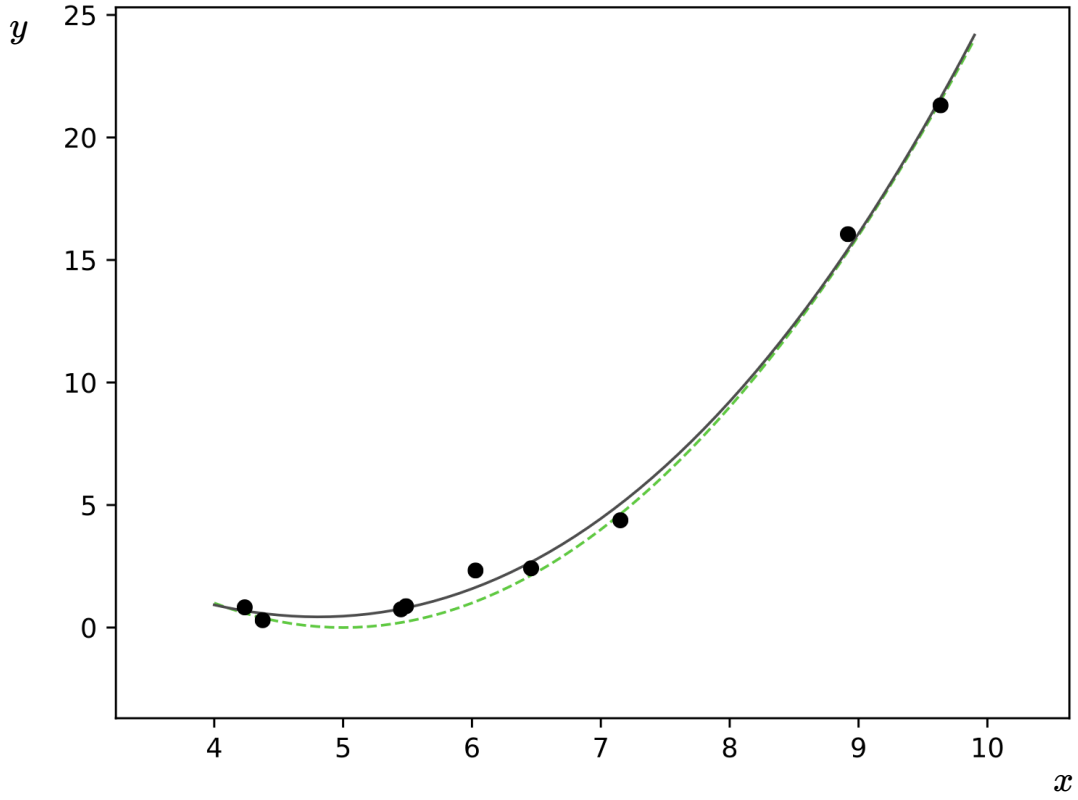


- 9 data points.
- Each data has one-dimensional feature $x \in \mathbb{R}$
- Label $y \in \mathbb{R}$

- Choose $k = 3$
- $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$
- How many scalar parameters to learn?
- 4 scalar parameters

Polynomial features for regression

k = 4

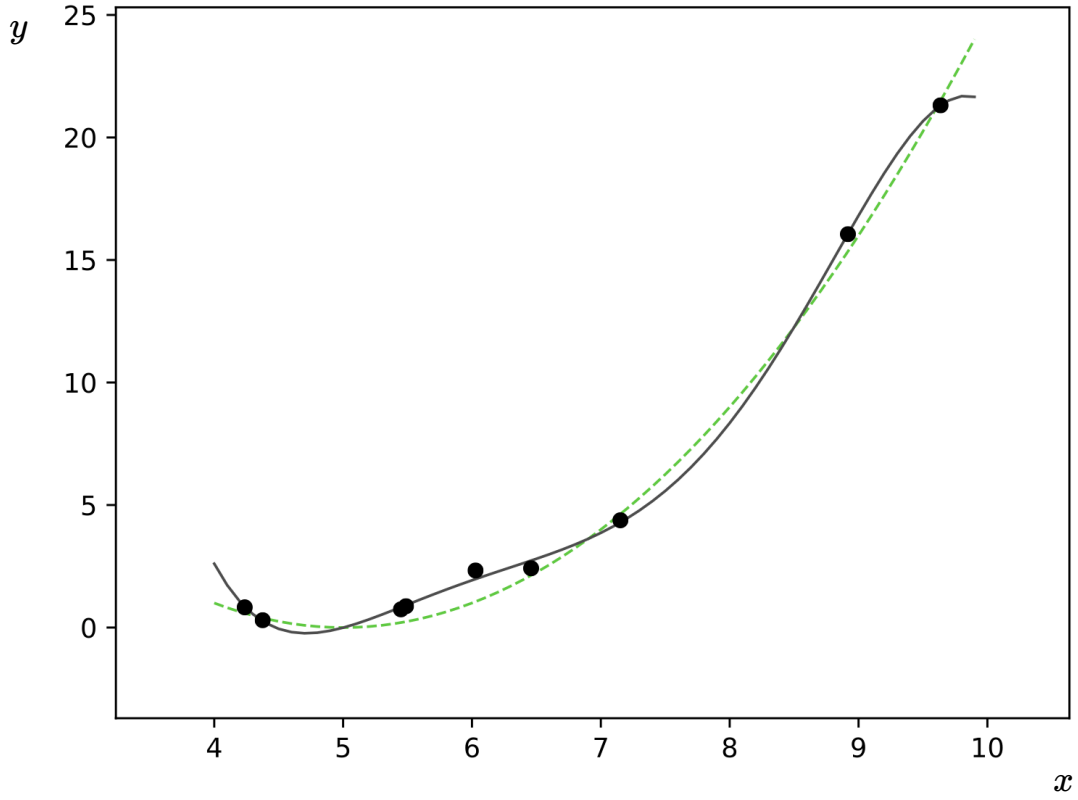


- 9 data points.
- Each data has one-dimensional feature $x \in \mathbb{R}$
- Label $y \in \mathbb{R}$

- Choose $k = 4$
- $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$
- How many scalar parameters to learn?
- 5 scalar parameters

Polynomial features for regression

k = 5

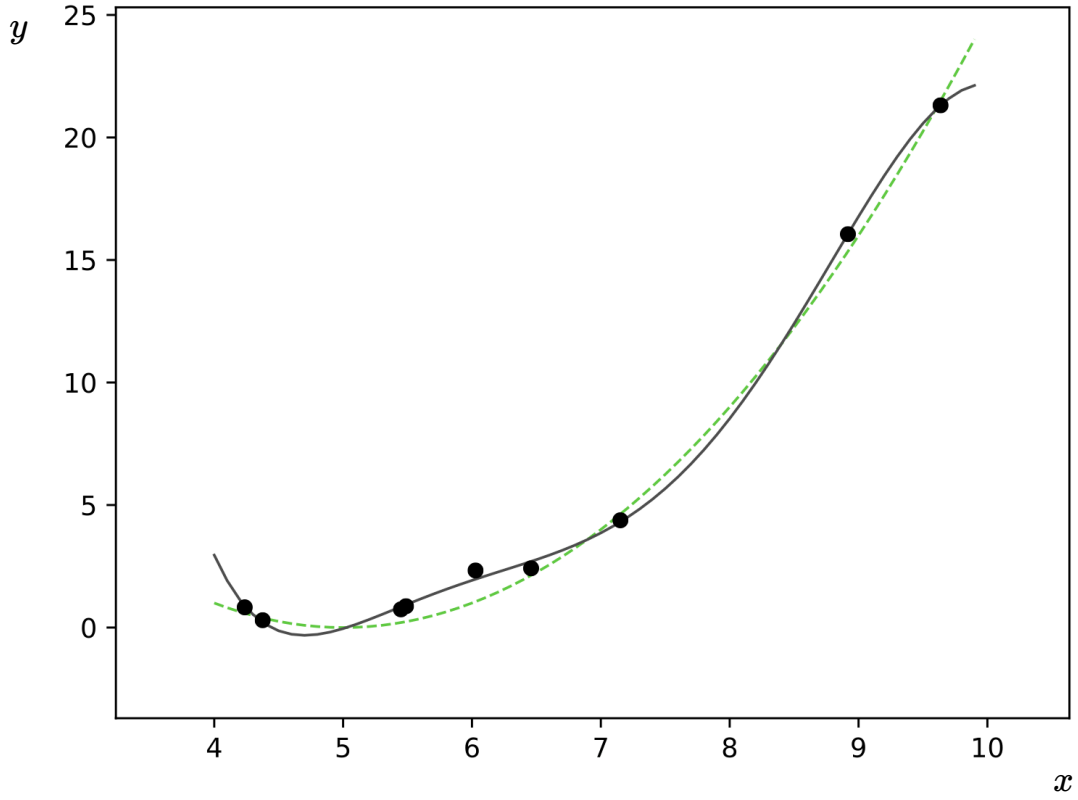


- 9 data points.
- Each data has one-dimensional feature $x \in \mathbb{R}$
- Label $y \in \mathbb{R}$

- Choose $k = 5$
- $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots \theta_k x^k$
- How many scalar parameters to learn?
- 6 scalar parameters

Polynomial features for regression

k = 6

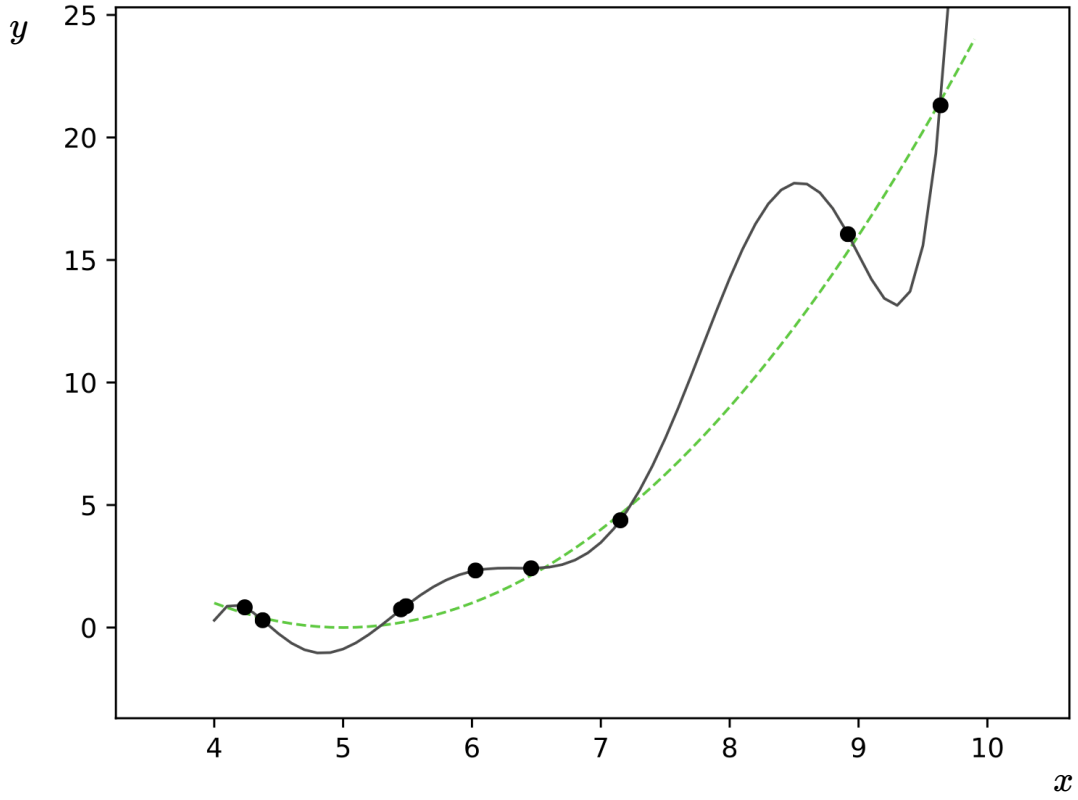


- 9 data points.
- Each data has one-dimensional feature $x \in \mathbb{R}$
- Label $y \in \mathbb{R}$

- Choose $k = 6$
- $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots \theta_k x^k$
- How many scalar parameters to learn?
- 7 scalar parameters

Polynomial features for regression

k = 7

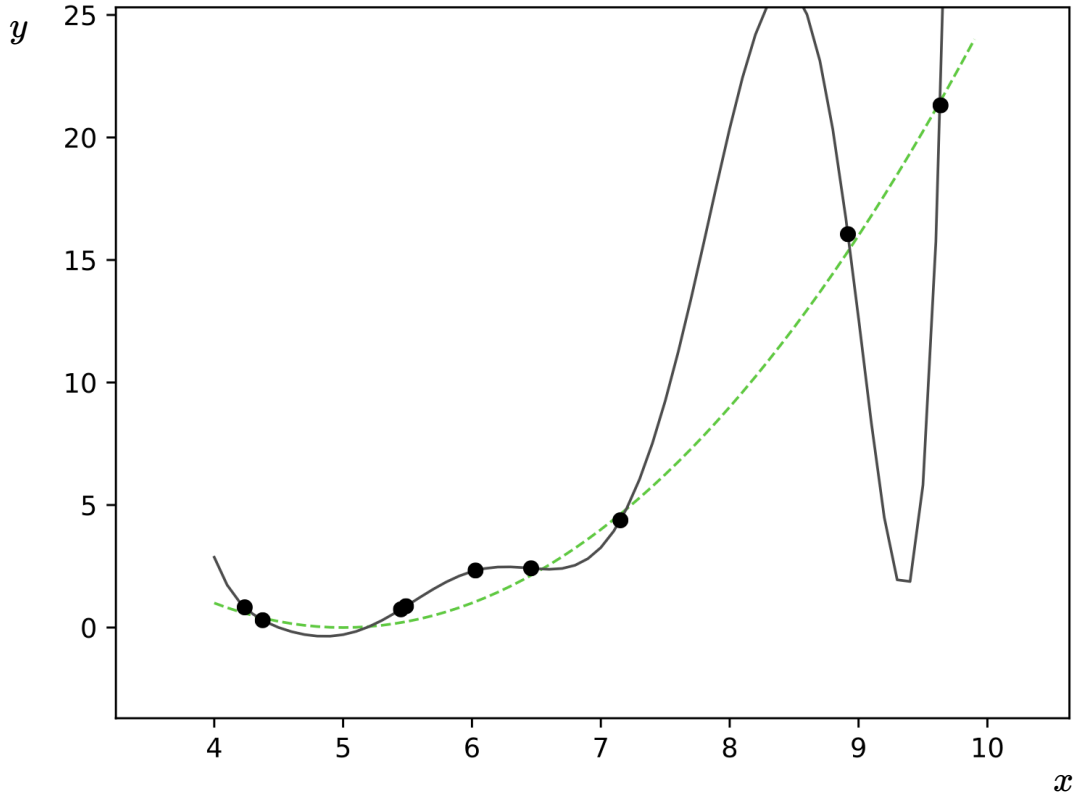


- 9 data points.
- Each data has one-dimensional feature $x \in \mathbb{R}$
- Label $y \in \mathbb{R}$

- Choose $k = 7$
- $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots \theta_k x^k$
- How many scalar parameters to learn?
- 8 scalar parameters

Polynomial features for regression

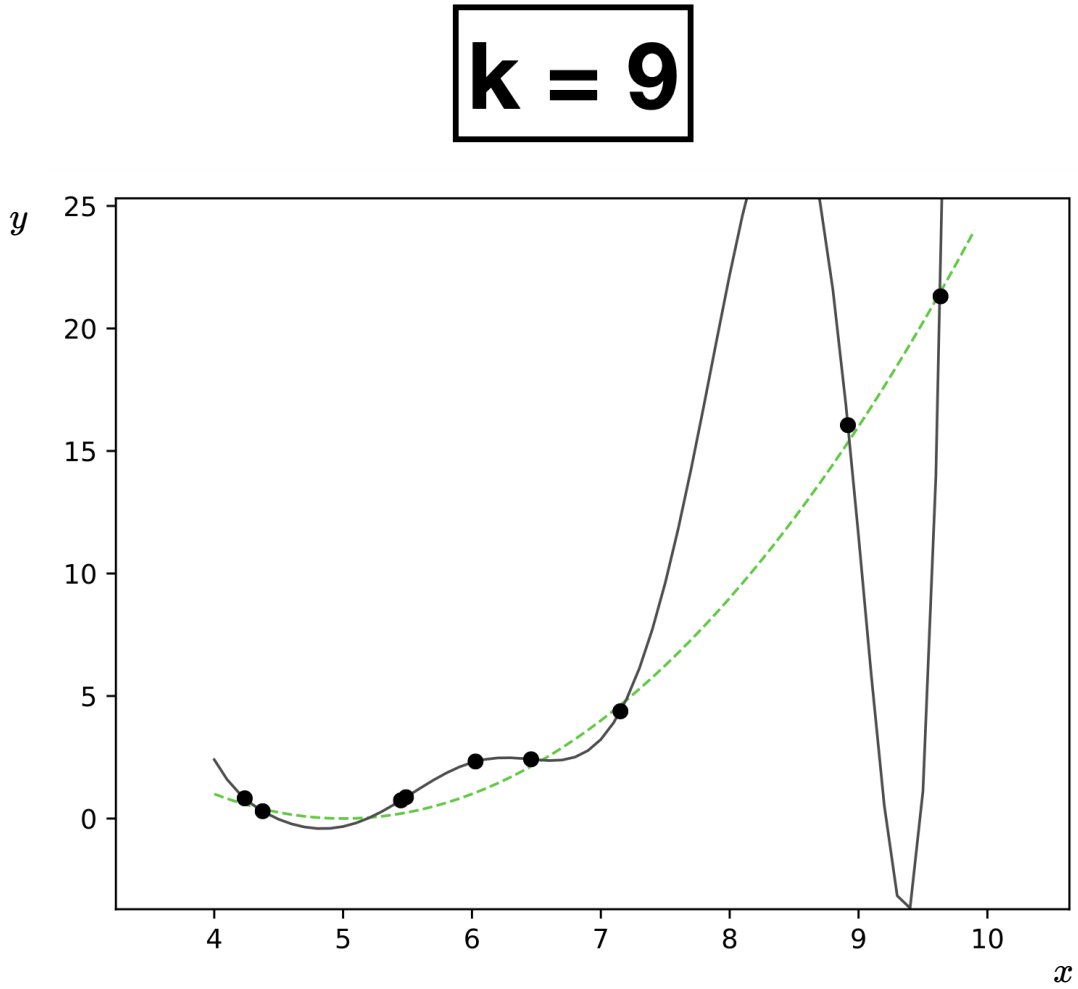
k = 8



- 9 data points.
- Each data has one-dimensional feature $x \in \mathbb{R}$
- Label $y \in \mathbb{R}$

- Choose $k = 8$
- $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots \theta_k x^k$
- How many scalar parameters to learn?
- 9 scalar parameters

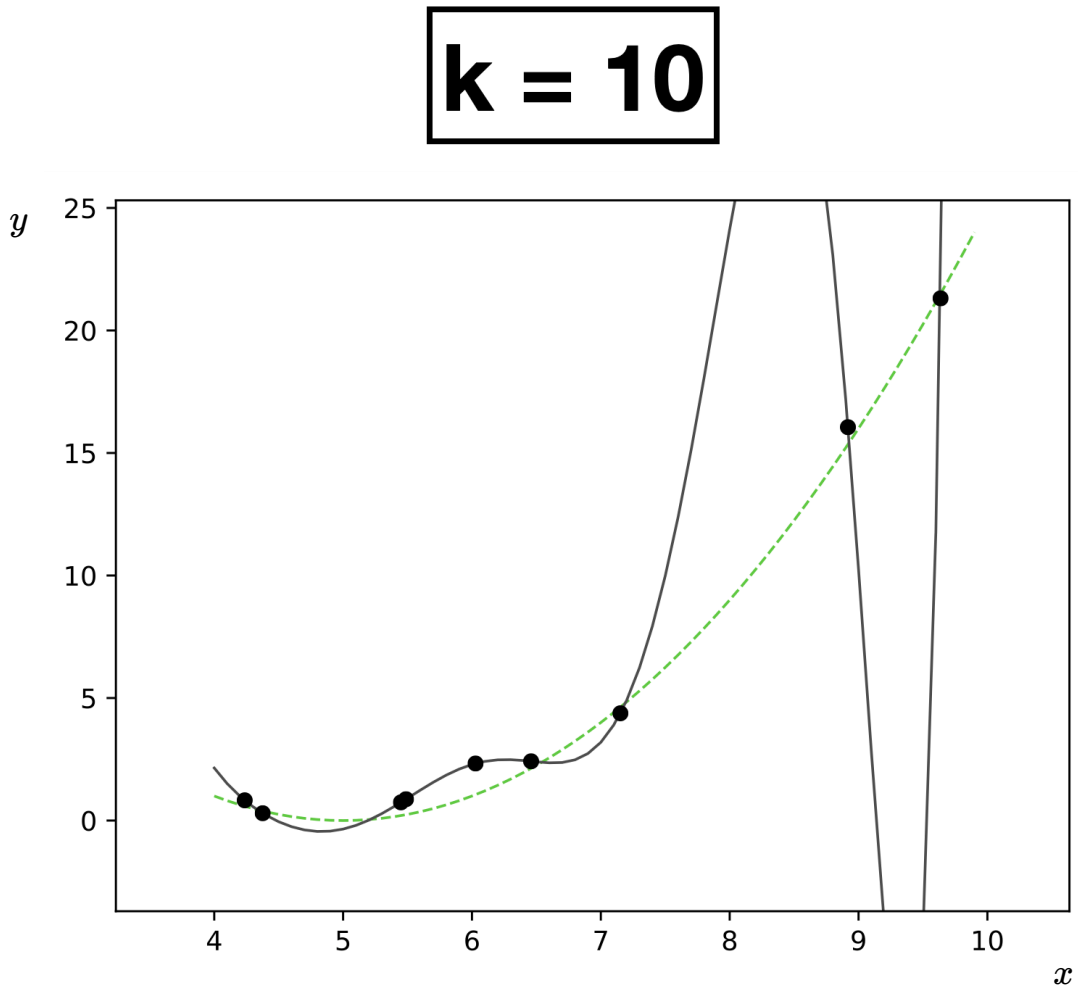
Polynomial features for regression



- 9 data points.
- Each data has one-dimensional feature $x \in \mathbb{R}$
- Label $y \in \mathbb{R}$

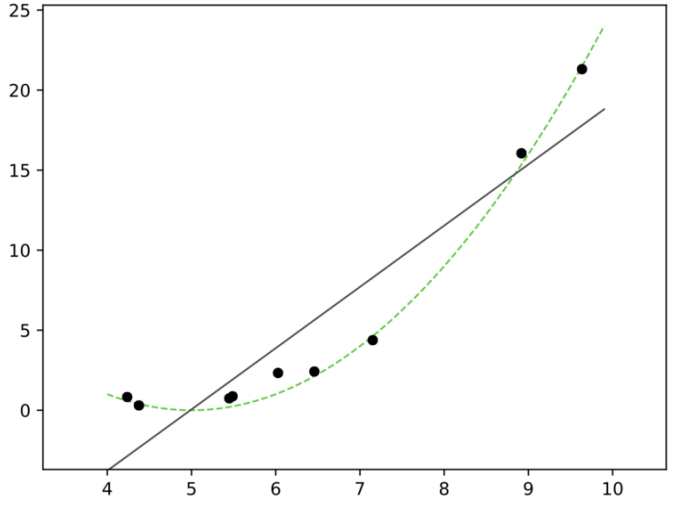
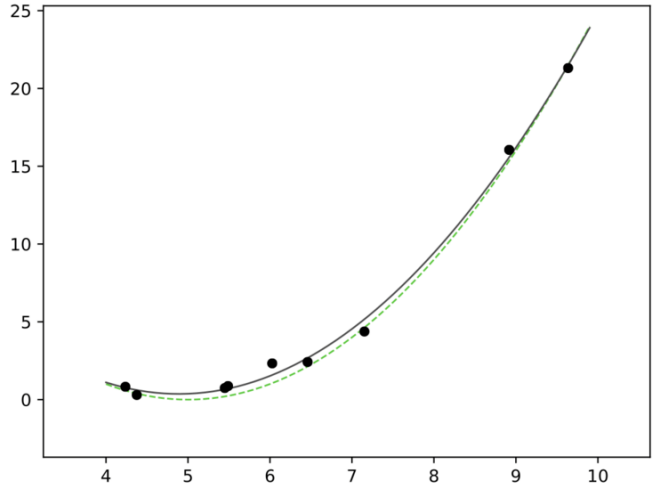
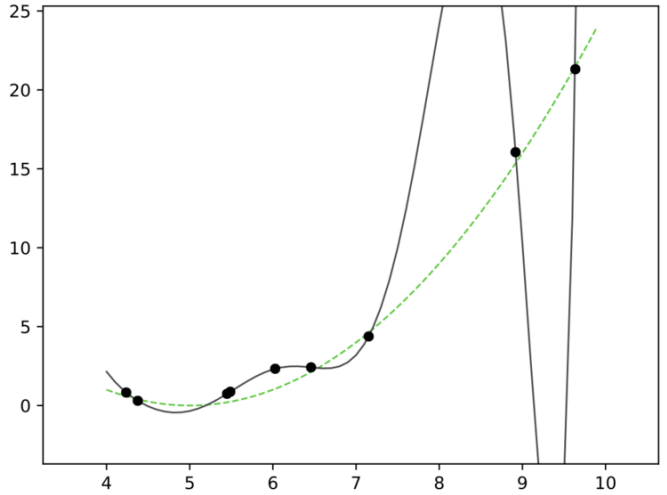
- Choose $k = 9$
- $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots \theta_k x^k$
- How many scalar parameters to learn?
- 10 scalar parameters

Polynomial features for regression



- Choose $k = 10$
- $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots \theta_k x^k$
- How many scalar parameters to learn?
- 11 scalar parameters

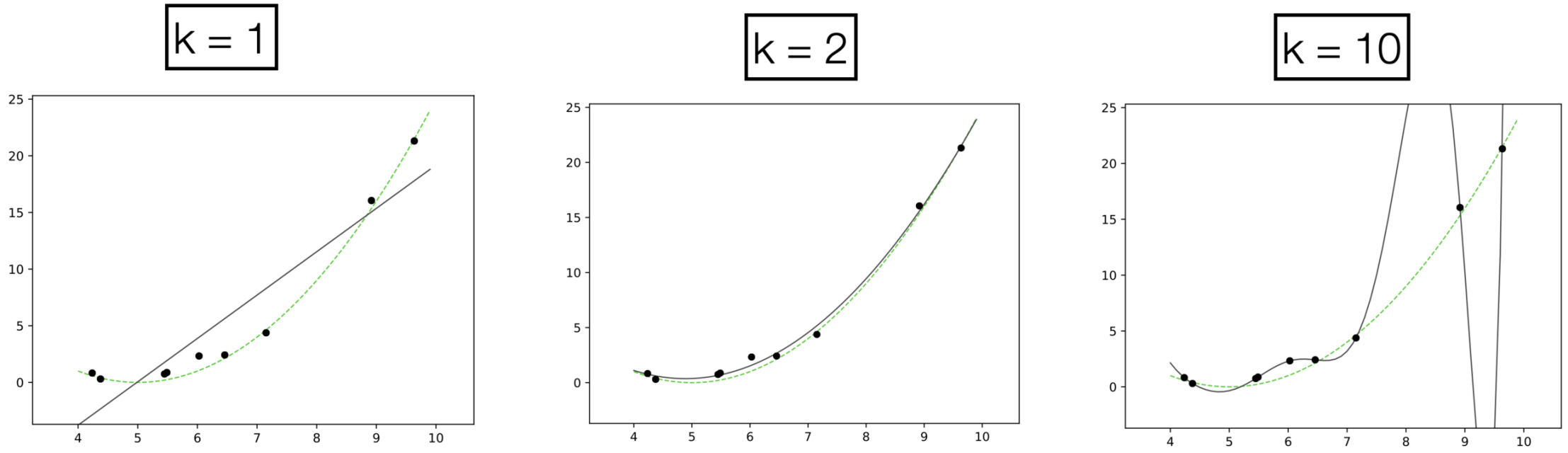
- The fit is perfect but "wild" (compared with the true function).
- Overfitting.
- It occurs when we have **too expressive** of a model (e.g., too many learnable parameters, too few data points to pin these parameters down).

Underfitting	Appropriate model	Overfitting
k = 1	k = 2	k = 10
		
high error on train set	low error on train set	very low error on train set
high error on test set	low error on test set	very high error on test set

Underfitting

Appropriate model

Overfitting



- k is a hyperparameter, can control the capacity/expressiveness of the hypothesis class (model class).
- Complex models with many rich features and free parameters have high capacity.
- How to choose k? Validation/cross-validation.

Quick summary

- Linear models are mathematically and algorithmically convenient but not expressive enough -- by themselves -- for most jobs.
- We can express really rich hypothesis classes by performing a **fixed** non-linear feature transformation first, then applying our linear regression or classification methods.
- Can think of these fixed transformation as "adapters", enabling us to use old tool in more situations.
- Standard feature transformations: polynomials; radial basis functions, absolute-value function.
- Historically, for a period of time, the gist of ML boils down to "feature engineering".
- Nowadays, neural networks can automatically assemble features.

Outline

- Recap (linear regression and classification)
- Systematic feature transformations
  - Polynomial features
- Domain-dependent, or goal-dependent, encoding
  - Numerical features
    - Standardizing the data
  - Categorical features
    - One-hot encoding
    - Factored encoding
    - Thermometer encoding

A more-complete/realistic ML analysis

- 1. Establish a goal & find data

  (Example goal: diagnose if people have heart disease based on their available info.)

- 2. Encode data in useful form for the ML algorithm.
- 3. Choose a loss, and a regularizer. Write an objective function to optimize

  (Example: logistic regression. Loss: negative log likelihood. Regularizer: ridge penalty)

- 4. Optimize the objective function & return a hypothesis

  (Example: analytical/closed-form optimization, sgd)

- 5. Evaluation & interpretation

A more-complete / realistic ML analysis

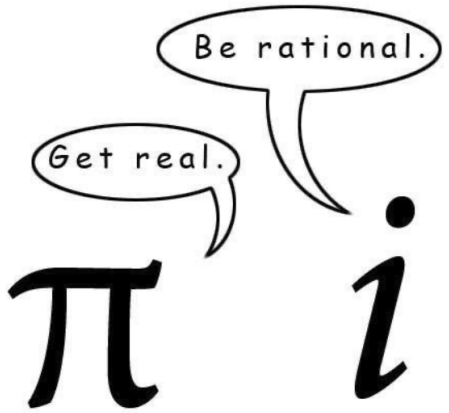
- 1. Establish a goal & find data

(Example goal: diagnose if people have heart disease based on their available info.)

- 2. Encode data in useful form for the ML algorithm.

Identify relevant info and encode as **real** numbers

Encode in such a way that's **sensible** for the task.



- First, need goal & data. E.g. diagnose whether people have heart disease based on their available information

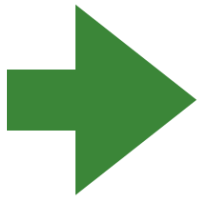
	has heart disease?	resting heart rate (bpm)	pain?	job	medicines	age	family income (USD)	
$y^{(1)}$								$x^{(1)}$
1	no	55	no	nurse	pain	40s	133000	
$y^{(2)}$								$x^{(2)}$
2	no	71	no	admin	beta blockers, pain	20s	34000	
3	yes	89	yes	nurse	beta blockers	50s	40000	
...								...
4	no	67	no	doctor	none	50s	120000	

Encode data in usable form

- Identify the labels and encode as real numbers

	has heart disease?
1	no
2	no
3	yes
4	no

$$\text{‘yes’} \leftrightarrow 1$$
$$\text{‘no’} \leftrightarrow 0$$



	$= y^{(1)}$
1	0
2	0
3	1
4	0

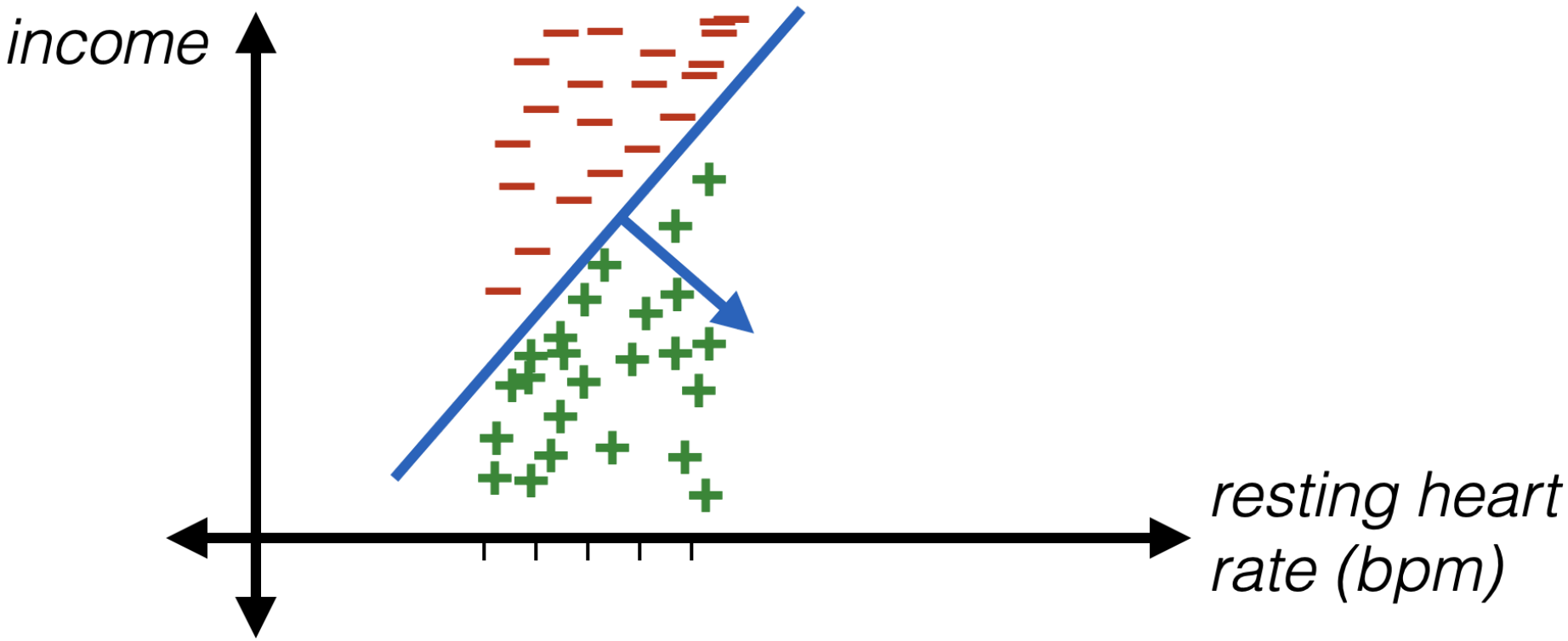
- Save mapping to recover predictions of new points

$$y_{\substack{\text{heart}\\\text{disease}}} = \text{sign}\left(\theta_{\substack{\text{heart}\\\text{rate}}} x_{\substack{\text{heart}\\\text{rate}}} + \theta_{\text{pain}} x_{\text{pain}} + \theta_{\text{job}} x_{\text{job}} + \theta_{\text{pill}} x_{\text{pill}} + \theta_{\text{age}} x_{\text{age}} + \theta_{\text{income}} x_{\text{income}} \right)$$

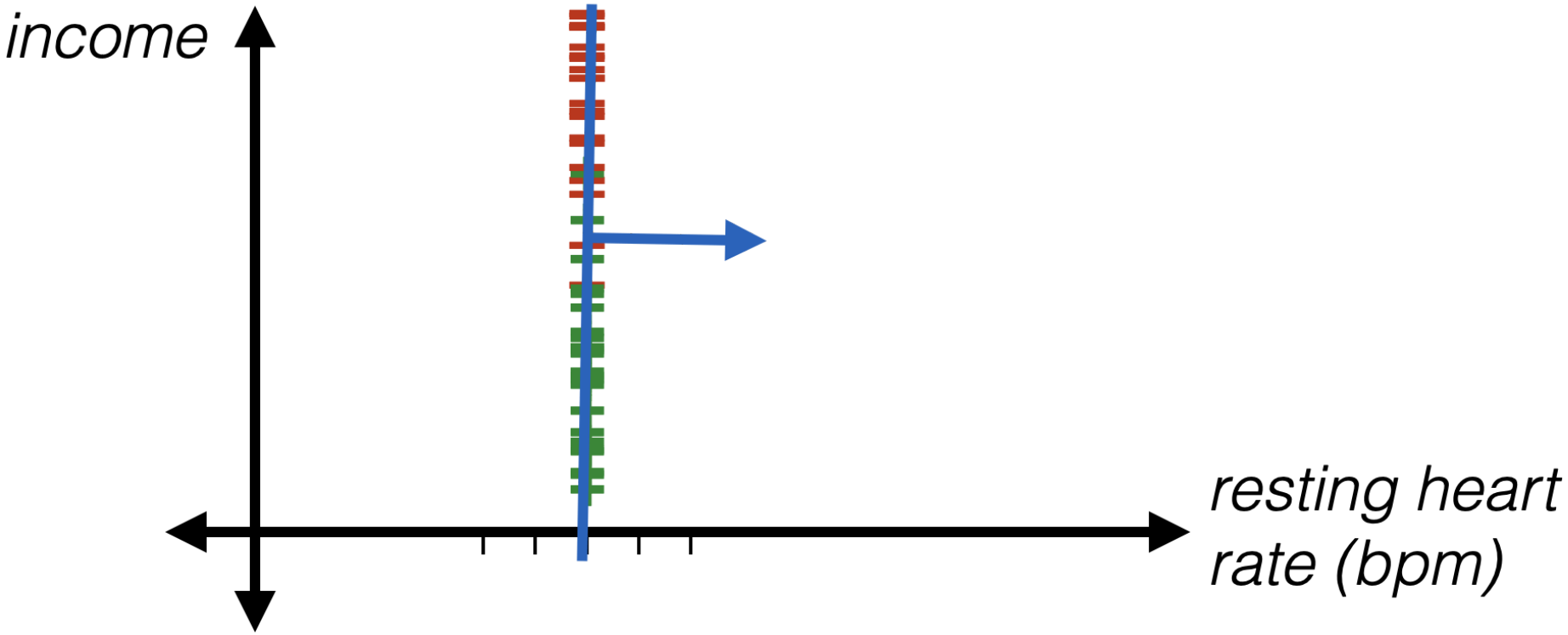
	has heart disease?	resting heart rate (bpm)	pain?	job	medicines	age	family income (USD)
1	0	55	no	nurse	pain	40s	133000
2	0	71	no	admin	beta blockers, pain	20s	34000
3	1	89	yes	nurse	beta blockers	50s	40000
4	0	67	no	doctor	none	50s	120000

- Resting heart rate and income are real numbers already
- Can directly use, but may not want to (see next slide)

Encoding numerical data

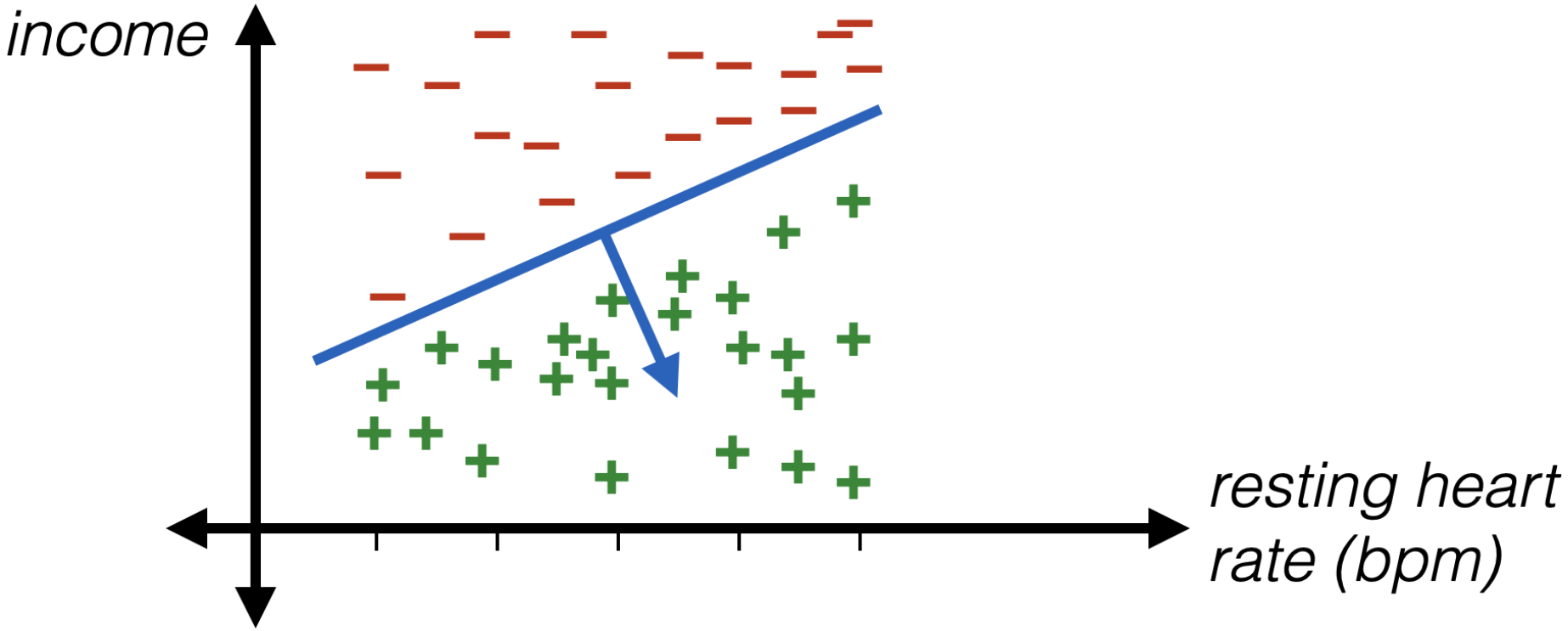


Encoding numerical data



Encoding numerical data

- Idea: standardize numerical data
- For ith feature and data point j: $$\phi_i^{(j)} = \frac{x_i^{(j)} - \text{mean}_i}{\text{stddev}_i}$$



$$y_{\substack{\text{heart}\\\text{disease}}} = \text{sign}\left(\theta_{\substack{\text{heart}\\\text{rate}}} x_{\substack{\text{heart}\\\text{rate}}} + \theta_{\text{pain}} x_{\text{pain}} + \theta_{\text{job}} x_{\text{job}} + \theta_{\text{pill}} x_{\text{pill}} + \theta_{\text{age}} x_{\text{age}} + \theta_{\text{income}} x_{\text{income}} \right)$$

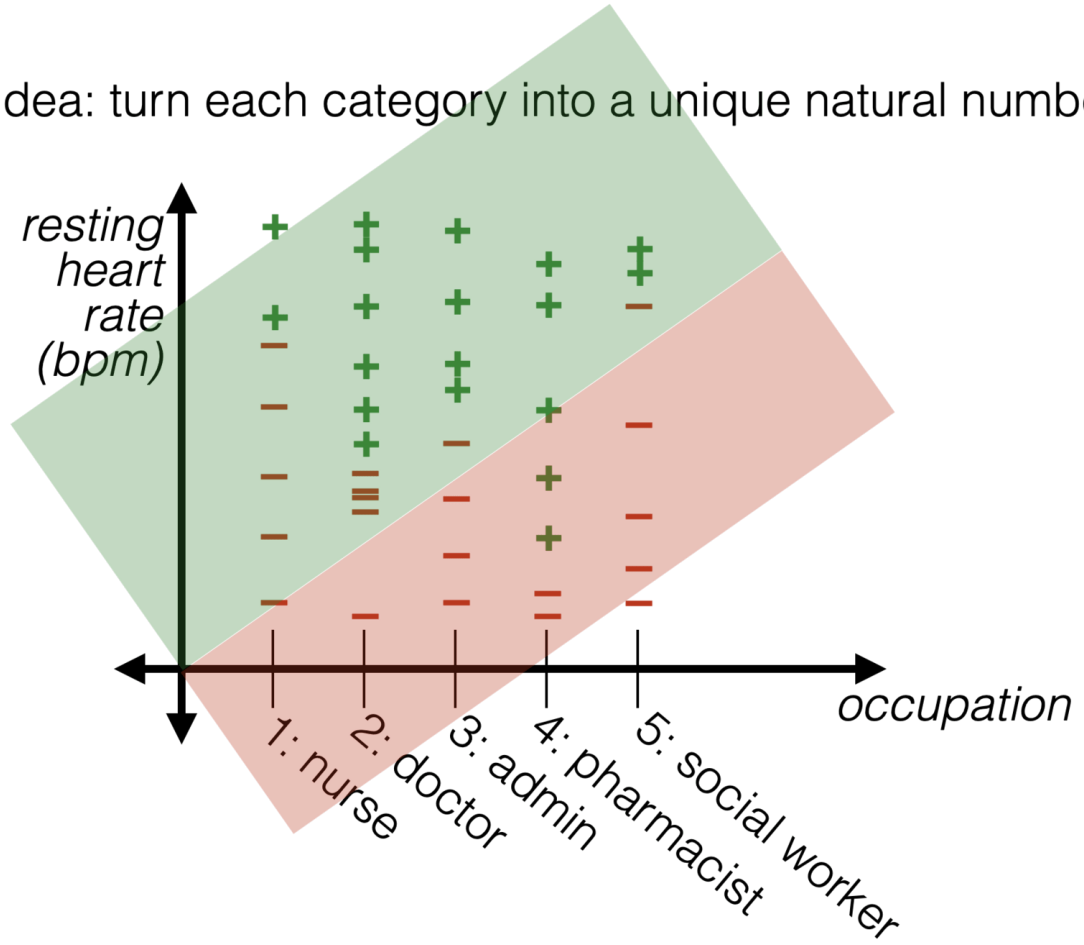
	has heart disease?	resting heart rate (bpm)	pain?	job	medicines	age	family income (USD)
1	0	-1.5	no	nurse	pain	40s	2.075
2	0	0.1	no	admin	beta blockers, pain	20s	-0.4
3	1	1.9	yes	nurse	beta blockers	50s	-0.25
4	0	-0.3	no	doctor	none	50s	1.75

$$y_{\substack{\text{heart}\\\text{disease}}} = \text{sign}\left(\theta_{\substack{\text{heart}\\\text{rate}}} x_{\substack{\text{heart}\\\text{rate}}} + \theta_{\text{pain}} x_{\text{pain}} + \theta_{\text{job}} x_{\text{job}} + \theta_{\text{pill}} x_{\text{pill}} + \theta_{\text{age}} x_{\text{age}} + \theta_{\text{income}} x_{\text{income}} \right)$$

	has heart disease?	resting heart rate (bpm)	pain?	job	medicines	age	family income (USD)
1	0	-1.5	no	nurse	pain	40s	2.075
2	0	0.1	no	admin	beta blockers, pain	20s	-0.4
3	1	1.9	yes	nurse	beta blockers	50s	-0.25
4	0	-0.3	no	doctor	none	50s	1.75

What about jobs?

- Idea: turn each category into a unique natural number



- Problem with this idea:
  - Ordering would matter
  - Incremental in the "job" would matter (by a fixed θ_{job} amount)

$$y_{\substack{\text{heart}\\\text{disease}}} = \text{sign}\left(\theta_{\substack{\text{heart}\\\text{rate}}} x_{\substack{\text{heart}\\\text{rate}}} + \theta_{\text{pain}} x_{\text{pain}} + \theta_{\text{job}} x_{\text{job}} + \theta_{\text{pill}} x_{\text{pill}} + \theta_{\text{age}} x_{\text{age}} + \theta_{\text{income}} x_{\text{income}} \right)$$

Better idea: One-hot encoding

	ϕ_i	ϕ_{i+1}	ϕ_{i+2}	ϕ_{i+3}	ϕ_{i+4}
nurse	1	0	0	0	0
admin	0	1	0	0	0
pharmacist	0	0	1	0	0
doctor	0	0	0	1	0
social worker	0	0	0	0	1

$$y_{\substack{\text{heart}\\\text{disease}}} = \text{sign}\Big(\theta_{\substack{\text{heart}\\\text{rate}}} x_{\substack{\text{heart}\\\text{rate}}} + \theta_{\text{pain}} x_{\text{pain}} + \underline{\theta_{\text{job}} x_{\text{job}}} + \theta_{\text{pill}} x_{\text{pill}} + \theta_{\text{age}} x_{\text{age}} + \theta_{\text{income}} x_{\text{income}} \Big)$$

$$\theta_{\text{job1}}\phi_{\text{job1}} + \theta_{\text{job2}}\phi_{\text{job2}} + \theta_{\text{job3}}\phi_{\text{job3}} + \theta_{\text{job4}}\phi_{\text{job4}} + \theta_{\text{job5}}\phi_{\text{job5}}$$

$$\theta_{\text{job1}}\phi_{\text{job1}} + \theta_{\text{job2}}\phi_{\text{job2}} + \theta_{\text{job3}}\phi_{\text{job3}} + \theta_{\text{job4}}\phi_{\text{job4}} + \theta_{\text{job5}}\phi_{\text{job5}}$$

$$y_{\substack{\text{heart}\\\text{disease}}} = \text{sign}\Big(\theta_{\substack{\text{heart}\\\text{rate}}} x_{\substack{\text{heart}\\\text{rate}}} + \theta_{\text{pain}} x_{\text{pain}} + \cancel{\theta_{\text{job}} x_{\text{job}}} + \theta_{\text{pill}} x_{\text{pill}} + \theta_{\text{age}} x_{\text{age}} + \theta_{\text{income}} x_{\text{income}} \Big)$$

	has heart disease?	resting heart rate (bpm)	pain?	j1,j2,j3,j4,j5	medicines	age	family income (USD)
1	0	-1.5	no	1,0,0,0,0	pain	40s	2.075
2	0	0.1	no	0,1,0,0,0	beta blockers, pain	20s	-0.4
3	1	1.9	yes	1,0,0,0,0	beta blockers	50s	-0.25
4	0	-0.3	no	0,0,0,1,0	none	50s	1.75

What about medicine?

	has heart disease?	resting heart rate (bpm)	pain?	j1,j2,j3,j4,j5	medicines	age	family income (USD)
1	0	-1.5	no	1,0,0,0,0	pain	40s	2.075
2	0	0.1	no	0,1,0,0,0	beta blockers, pain	20s	-0.4
3	1	1.9	yes	1,0,0,0,0	beta blockers	50s	-0.25
4	0	-0.3	no	0,0,0,1,0	none	50s	1.75

- Should we use one-hot encoding?

	ϕ_i	ϕ_{i+1}	ϕ_{i+2}	ϕ_{i+3}
pain	1	0	0	0
pain & beta blockers	0	1	0	0
beta blockers	0	0	1	0
no medications	0	0	0	1

$$\theta_{\text{combo1}}\phi_{\text{combo1}} + \theta_{\text{combo2}}\phi_{\text{combo2}} + \theta_{\text{combo3}}\phi_{\text{combo3}} + \theta_{\text{combo4}}\phi_{\text{combo4}}$$

Better idea: factored encoding

	ϕ_i	ϕ_{i+1}
pain	1	0
pain & beta blockers	1	1
beta blockers	0	1
no medications	0	0

$$\theta_{\text{pain-pill}}\phi_{\text{pain-pill}} + \theta_{\text{beta-pill}}\phi_{\text{beta-pill}}$$

Recall, if used one-hot, need exact combo in data to learn corresponding parameter

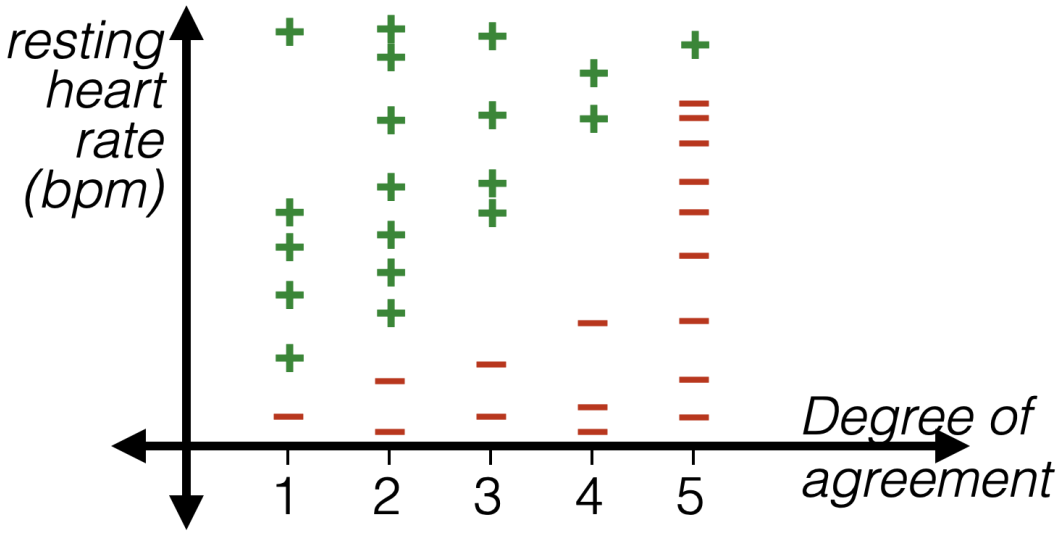
	ϕ_i	ϕ_{i+1}	ϕ_{i+2}	ϕ_{i+3}
pain	1	0	0	0
pain & beta blockers	0	1	0	0
beta blockers	0	0	1	0
no medications	0	0	0	1

$$\theta_{\text{combo1}}\phi_{\text{combo1}} + \theta_{\text{combo2}}\phi_{\text{combo2}} + \theta_{\text{combo3}}\phi_{\text{combo3}} + \theta_{\text{combo4}}\phi_{\text{combo4}}$$

Thermometer encoding

- Numerical data: order on data values, and differences in value are meaningful
- Categorical data: no order on data values, one-hot
- Ordinal data: order on data values, but differences not meaningful

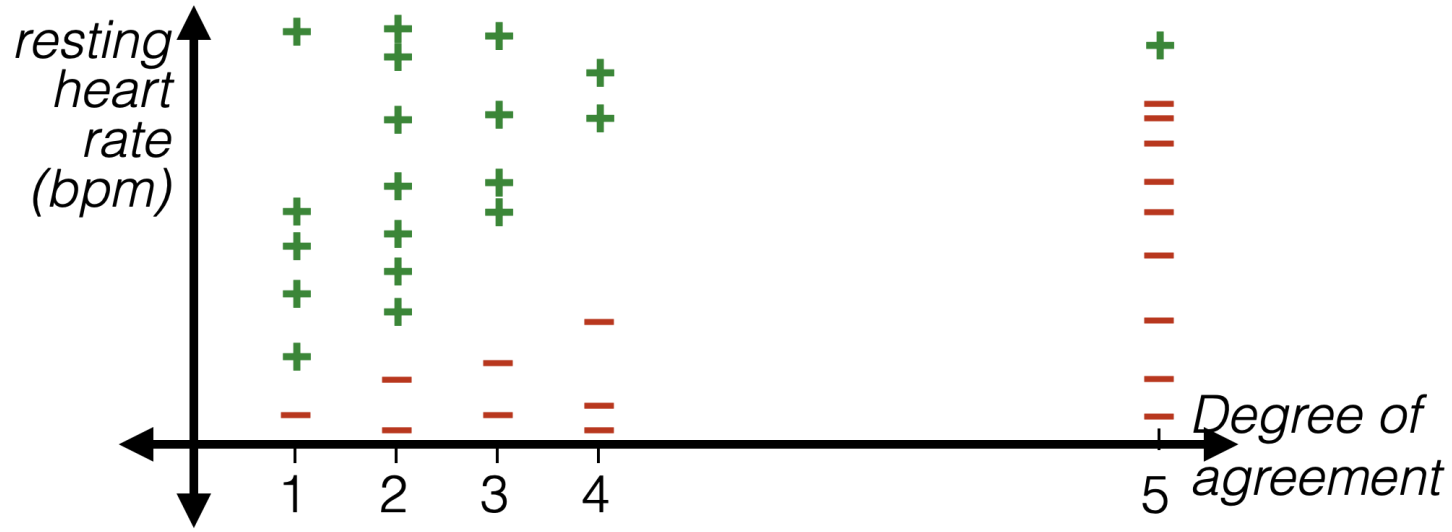
Strongly disagree	**Disagree**	**Neutral**	**Agree**	**Strongly agree**
1	2	3	4	5



Thermometer encoding

- Numerical data: order on data values, and differences in value are meaningful
- Categorical data: no order on data values, one-hot
- Ordinal data: order on data values, but differences not meaningful

Strongly disagree	**Disagree**	**Neutral**	**Agree**	**Strongly agree**
1	2	3	4	5



Thermometer encoding

- Numerical data: order on data values, and differences in value are meaningful
- Categorical data: no order on data values, one-hot
- Ordinal data: order on data values, but differences not meaningful

$\theta_{\text{how-agreed}}\phi_{\text{how-agreed}}$

Strongly disagree	**Disagree**	**Neutral**	**Agree**	**Strongly agree**
1	2	3	4	5

Strongly disagree	**Disagree**	**Neutral**	**Agree**	**Strongly agree**
1,0,0,0,0	1,1,0,0,0	1,1,1,0,0	1,1,1,1,0	1,1,1,1,1

$$\theta_{\text{strong-disagree-base}}\phi_{\text{strong-disagree-base}} +$$
$$\theta_{\text{slightly-more-agreement}}\phi_{\text{slightly-more-agreement}} +$$
$$\theta_{\text{from-disagree-to-neutral}}\phi_{\text{from-disagree-to-neutral}} +$$
$$\theta_{\text{from-neutral-to-agree}}\phi_{\text{from-neutral-to-agree}} +$$
$$\theta_{\text{from-agree-to-strongly-agree}}\phi_{\text{from-agree-to-strongly-agree}}$$

Summary

- Linear models are mathematically and algorithmically convenient but not expressive enough -- by themselves -- for most jobs.
- We can express really rich hypothesis classes by performing a **fixed** non-linear feature transformation first, then applying our linear (regression or classification) methods.
- When we "set up" a problem to apply ML methods to it, it's important to encode the inputs in a way that makes it easier for the ML method to exploit the structure.
- Foreshadowing of neural networks, in which we will learn complicated continuous feature transformations.

We'd love it for you to share some lecture feedback.

Thanks!