



COMPUTER SCIENCE

INDIANA UNIVERSITY

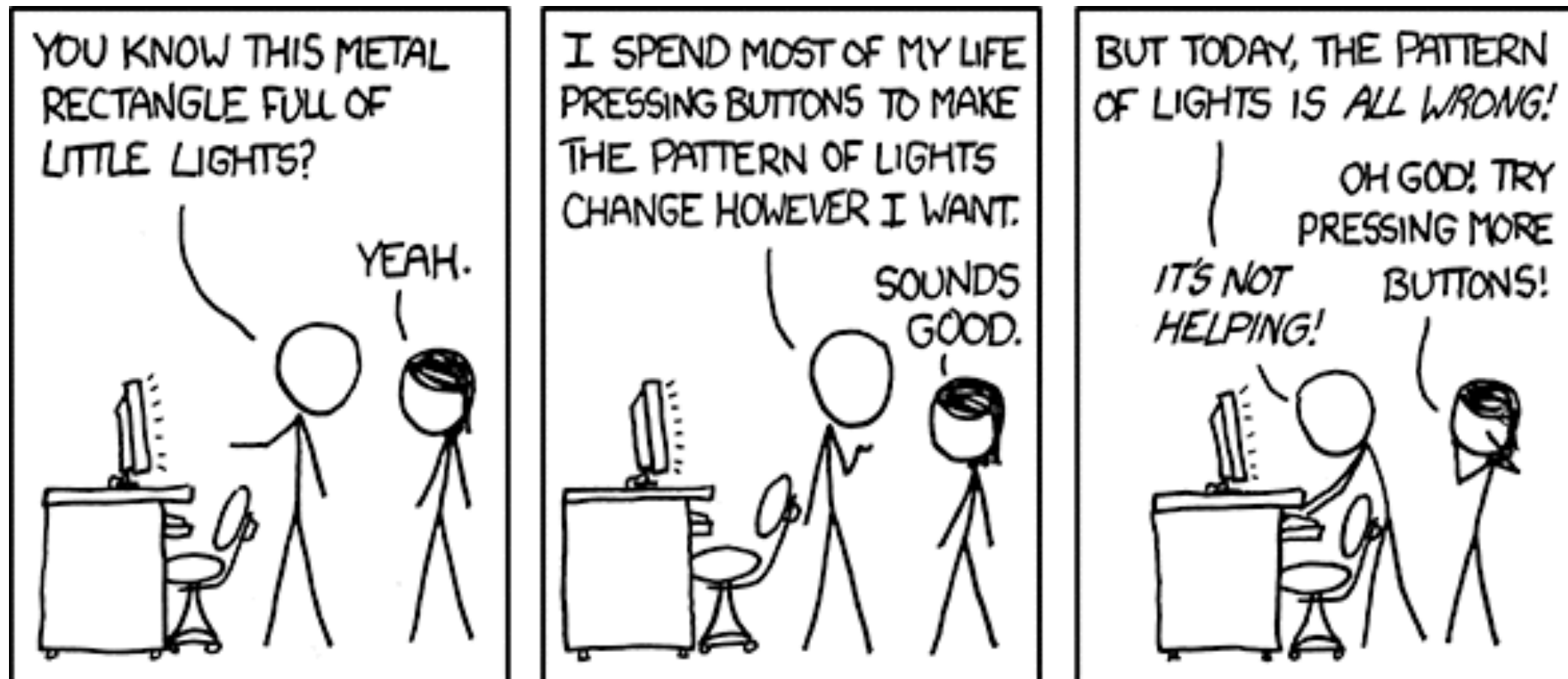
School of Informatics and Computing
Bloomington

Principles of Machine Learning

CSCI-B 455
Spring 2017
Martha White

What is this course about?

- **One line goal:** demystify the seemingly huge collection of machine learning algorithms by exploring their mathematical foundation



What is this course about?

- The focus is on fundamental data analysis/statistics concepts, and not necessarily on application of these algorithms
 - though you will get to do that too
- Application of algorithms is simpler when you understand their development and underlying assumptions
- Overall goal: understand how to use (heaps of) data to make predictions about novel events, including
 - what assumptions to make and how to formalize the problem
 - how to derive algorithms for your problem
 - ascertain your confidence in the predictions (i.e., evaluate your approach)



Basic information

Class meets:

Time: MW 2:30pm – 3:45pm

Place: SPEA, PV 247

Instructor:

Martha White

Office: Lindley Hall 401E

Email: martha@indiana.edu

Web: <http://homes.soic.indiana.edu/martha/>

Office Hours:

Time: T 3:00pm-5:00pm (is this a good time?)
or by appointment

Place: Lindley Hall 401E

Class Web Site:

<https://iu.instructure.com/courses/1560796>



Associate Instructors (Teaching Assistants)

Inhak Hwang
Andrew Patterson

Office Hours:

Times:



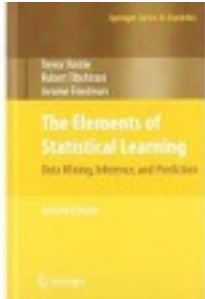
T 2:00pm-4:00pm (is this a good time?)

Place: Somewhere in Lindley Hall (TBD)

Come to office hours!



Textbook information

- **Main notes** provided in Canvas
 - written by Predrag Radivojac and myself (about 130 pages)
- **[Optional] reference material/recommended readings:**
 - An Introduction to Statistical Learning, 2013 (accessible and concise)
 - Bayesian Reasoning and Machine Learning by D. Barber, Cambridge Press 2012.
 - Pattern Recognition and Machine Learning by C. M. Bishop, Springer 2006.
 - The Elements of Statistical Learning by T. Hastie, R. Tibshirani, and J. Friedman, 2009

Marks

- 40%: Assignments (4), a mixture of mathematical and programming exercises (code provided in python)
 - Must complete assignments independently
 - Fourth assignment provides an opportunity to work on a more open-ended problem, using your dataset of choice and whatever algorithms you like
- 15%: Quizzes (2)
- 35%: Final exam, May 1, 10:00 p.m. - noon
- 10%: Thought questions (3), show that you're reading and thinking about the material



More on grading

- Top performers will get A+
 - This course is not curved; I decide the letter thresholds (Note that I don't enjoy having to assign you a grade, but it helps you learn)
- Previous course distribution: 1 C, 3 C+, 4 B-, 6 B, 9 B+, 4 A-, 9 A, 7 A+
- Final mark includes only top 3 best assignments for each student
 - i.e., your worst assignment mark will not be included in your final mark
- No arguing for marks with TAs
 - should only ask: "Can you help me understand this question?"
 - Even if you get 10% improvement on an assignment, at best this translates into 1% for the final grade. It is a waste of your (and everyone's) time.



More about assignments

- **All assignments are individual:** no discussing questions
 - The assignments are feasible on your own; you will learn much more and get more out of the course when you accomplish this
- Acknowledge sources (websites, books) in your documents
 - can be typed up in LaTeX; I have given you latex files on Canvas, for each assignment
 - or write legibly and upload scanned document
- We are very serious about academic honesty

My expectations

- Basic mathematical skills
 - some calculus
 - some probabilities
 - some linear algebra
 - I will give crash courses in these along the way, so if you do not have this background, a willingness to learn these topics is a prerequisite
- You are hardworking and motivated to learn (machine learning)
- You are motivated to think beyond the material and ask open-ended questions



What to expect from this course

- I know you have expectation of me too
 - I will try to be transparent in marking and course choices
 - I am here to help you learn; I will treat you with respect and listen thoughtfully to your questions (I love to answer questions and give advice!)
 - Feel free to give me feedback (e.g., Miss Martha, you are talking too fast)
 - I will provide an mechanism for anonymous feedback
- This course will be quite mathematical, with derivations of details
 - this is absolutely necessary, and will make you much more skilled in ML
- By the end of the course, you should have a good grasp of fundamental concepts in ML and algorithm derivation for ML



Thought questions

- University is about thinking and asking questions
 - e.g., research is actually about asking good questions
 - a question does not have to have an answer, but it can be thought-provoking or make you think about a topic differently
- “Thought questions” correspond to (short) readings in the notes, and should demonstrate you’ve read and thought about the topics
- There are no stupid questions (so ask any questions in class/office hours/email), but “thought questions” are to demonstrate insight and so I have some requirements



General format for thought questions

- (1) First show/explain how you understand a concept
- (2) Given this context, propose a follow-up question
- (3) Propose an answer to the question, or how you might find it
- Additional note: framing a coherent, concise thought is a skill. When writing your thought question, ask yourself: is this clear?



Examples of “good” thought questions

- After reading about independence, I wonder how one could check in practice if two variables are independent, given a database of samples? Is this even possible? One possible strategy could be to approximate their conditional distributions, and examine the effects of changing a variable. But it seems like there could be other more direct or efficient strategies.
- PCA encodes a simple linear relationship between the data and underlying subspace. Why is PCA so widely used? It seems simple and I would not expect it to be able to encode complex properties. Potentially its simplicity is an answer.



Examples of “bad” thought questions

- I don't understand linear regression. Could you explain it again? (i.e. a request for me to explain something, without any insight)
- Derive the maximum likelihood approach for a Gaussian. (i.e., an exercise question from a textbook)
- What is the difference between a probability mass function and a probability density function? (i.e., a question that could easily be answered from reading the definitions in the notes)
- How are Boltzmann machines and feedforward neural networks different? (i.e., again a definition)
 - But the following modification would be good: “I can see that Boltzmann machines and feedforward neural networks are different, in that the first is undirected and the second directed. How does this difference impact modeling properties and accuracy of estimation in practice?”



What is machine learning?

- We could label it in many different ways including
 - Data-driven approach to artificial intelligence
 - Applied statistics
- Some keywords associated with machine learning
 - Prediction, probability, samples, data, function, optimization, stochastic, ...



ICML 2015 word cloud

Where did machine learning come from?

- Let's first step back to the goals of artificial intelligence
- Traditional AI approaches were expert-based
 - logic approaches for theorem proving
 - expert systems
- Machine learning arose as a data-driven approach to solve artificial intelligence problems
 - why the shift? increased computation, availability of data and efficacy of data driven approaches (largely driven by availability of data)



What are the ultimate goals?

- The focus for many ML researchers has shifted from AI towards generally solving important (practical) problems
 - computer vision, speech recognition, clustering, modeling temporal data, ...
- This includes a focus on understanding intrinsic properties of a learning problem
 - is it difficult to learn (e.g., NP-hard)?
 - how can it be formulated in a precise way? (e.g., explicit probabilistic assumptions, preference for “simpler” hypotheses)
 - how many samples are needed to learn the model (epsilon) accurately?
 - how well does the learned model generalize to new samples?



A starting example

- Our goal: obtain a function to predict house prices, given age
 - $f(\text{age}) = \text{price of house}$
- Imagine you have a dataset of previous house sales this year, with attribute information Age and target Price
 - $(\text{age}_1, \text{price}_1), (\text{age}_2, \text{price}_2), \dots, (\text{age}_9, \text{price}_9)$
- Presumably these previous houses give us some information about the function between age and price
- Idea: if we can learn a function to accurately recreate these (age, price) pairs, then it could provide good predictions



Formalizing the optimization

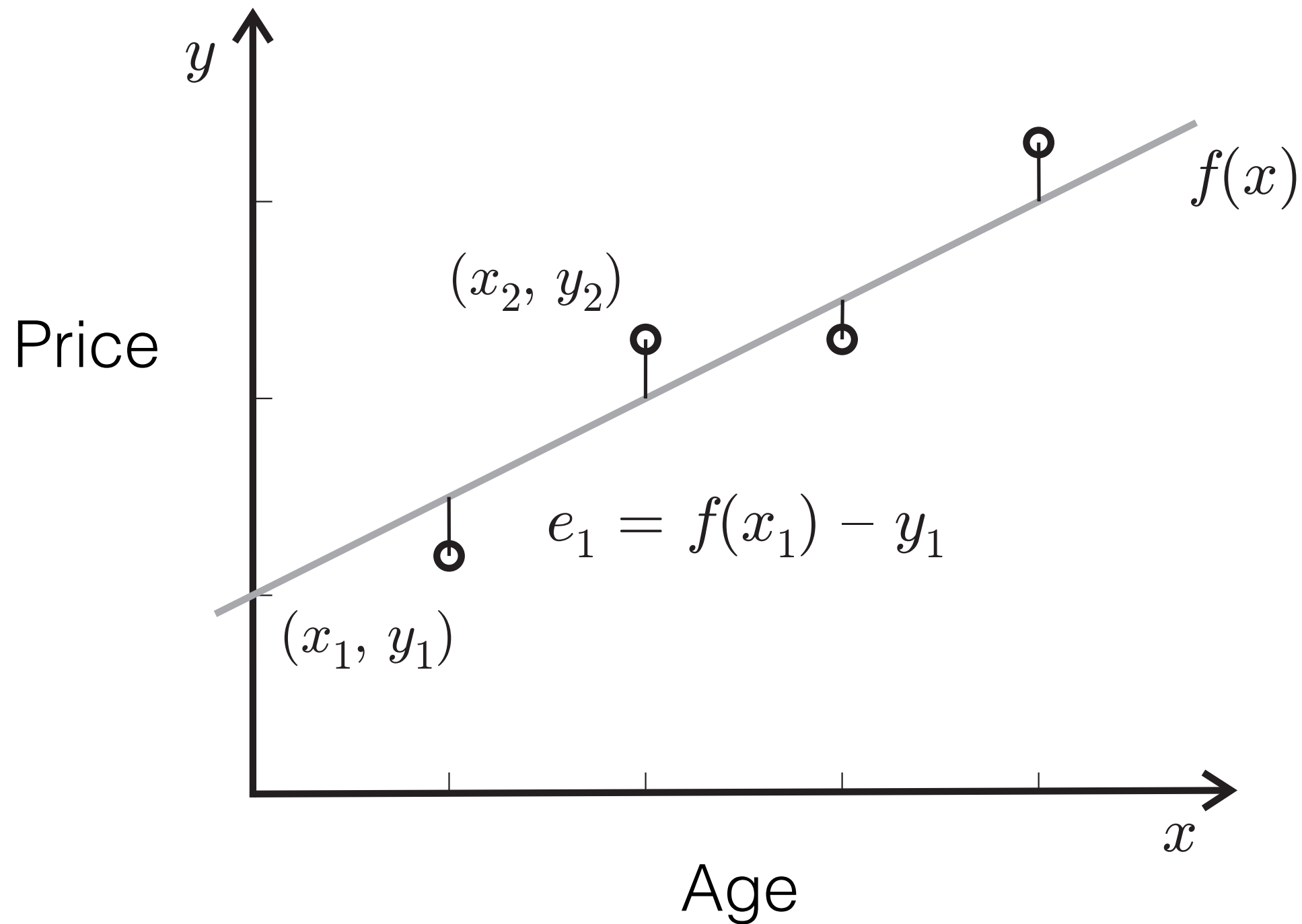
Let x = age and y = price

Make difference between $f(x_i)$ and y_i small

$$\text{Minimize } \sum_{i=1}^9 (f(x_i) - y_i)^2$$

Need to pick a form for f

Linear function f



$$f(x) = w_0 + w_1 x$$

Solving for the optimal function

$$\min_{f \text{ in function space}} \sum_{i=1}^9 (f(x_i) - y_i)^2$$
$$= \min_{w_0, w_1} \sum_{i=1}^9 (w_0 + w_1 x_i - y_i)^2$$

- We will see later how to solve this
- Questions:
 - Would you use this to predict the price of a house? Why or why not?
 - Will this predict well? How do we know?
 - What is missing to make these assessments?

General approach

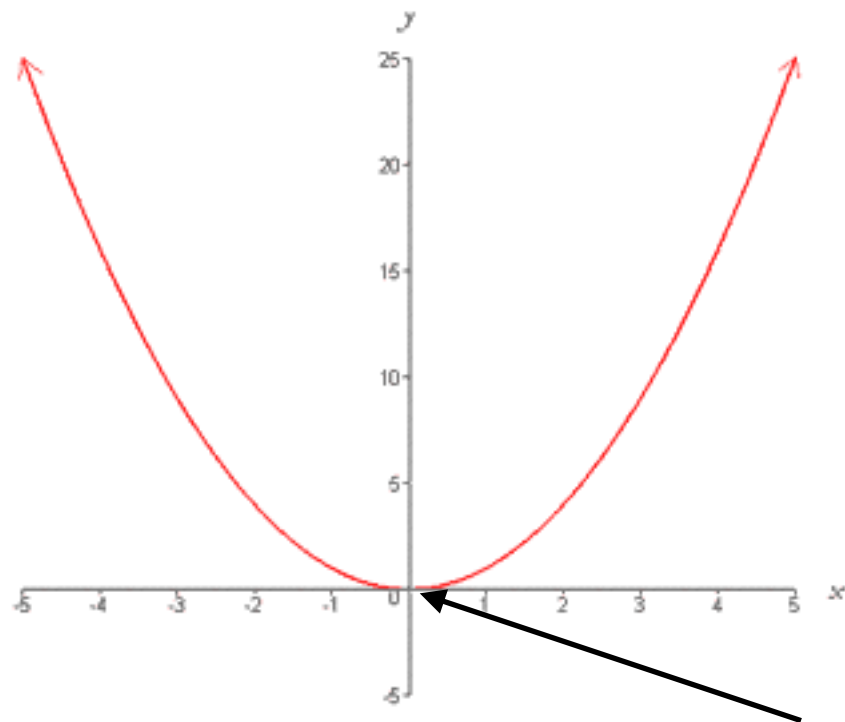
- Specify the problem and analyze the solution using a probabilistic approach
 - e.g., unlikely that we can learn a deterministic function from $f(\text{age})$ to price, many houses will have the same age but different prices
 - need to specify the problem so there is a distribution over targets (price), given attributes about the item (age)
 - stochasticity largely comes from partial observability
- Generalize beyond simple linear functions
 - polynomial regression
 - radial basis function network
 - neural networks



Exercise

- Solve for optimal w for function $f(x) = xw$

$$\min_w \sum_{i=1}^n (x_i w - y_i)^2$$



Optimal point, derivative is zero

Demo: classification

- Imagine have houses described by age and size, and now want to determine if cost is high or low (classify into two groups)
- Now want a function that either
 - returns -1 or 1
 - or return $f(x) < 0$ for class 1 and $f(x) > 0$ for class 2
- Let's look at a line learned by logistic regression (a simple ML algorithm) for separating two groups



Uses of machine learning in the real world

- Character recognition for mail addresses (as early as mid 90s)
 - 30% accuracy in 1997 to 88% accuracy in 2004, 98% now
- Spam filtering $p(\text{email} = \text{spam} | \text{information about email})$
- Netflix challenge (matrix completion)
- Speech recognition (deep learning)

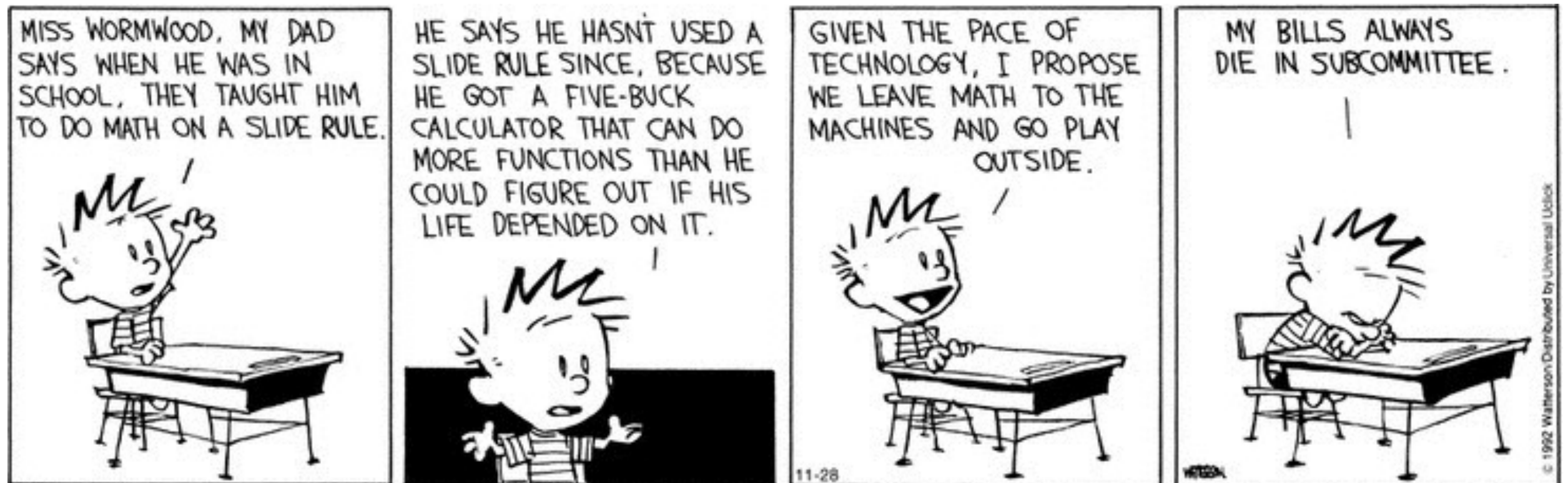


Topic overview

- Parameter estimation and prediction problems (Chapters 2 and 3)
 - the core background for modeling in machine learning
- Linear regression (Chapter 4)
- Linear classifiers (Chapter 6)
- Representations for ML (Chapter 7)
 - help make linear predictors more powerful (e.g., neural networks)
- Statistical learning theory and empirical evaluation (Chapter 8)
- ... and any other topics listed on the syllabus if we have time (e.g., boosting)



Next: crash course in probability



- Probabilities underly much of machine learning
 - enable precise modeling of uncertainty
- Understanding assumptions and how to build extensions is key for effectively using machine learning algorithms

Anticipated questions

- Where can I find background exercises/questions?
 - Try machine learning books (e.g., Barber's book)
 - Try applied statistics textbooks (e.g., All of Statistics)
- The course is too slow / too fast
 - If too slow, come talk to me and I'll show you how to get more from it
 - If too fast, don't be too frustrated; slowly the information and way of thinking start to make sense.



Anticipated questions

- Why are we learning such simple models? Isn't the modern approach neural nets and we should start there?
 - The foundational material is key for properly understanding more complex models. As you will see, building up from the foundation is critical to understanding neural nets. Many people do not understand that backpropagation is gradient descent, the choice of activation functions, etc.
 - Without an in-depth understanding, you will not be able to use machine learning effectively for your novel setting, and may even make terrible modeling decisions



Anticipated questions

- Why aren't we programming more? I don't need to learn about linear regression, I can just use packages.
- Quote from previous student: "One thing I would like to strengthen is the importance of knowing the mathematical details of how to deduct each model. Some people might think it is too tedious to know, but I have done projects on modifying training algorithms, like adding priors or changing the lose functions, and it requires me to know those details so that I know what I am supposed to do. Projects related to basic models, like linear regression, logistic regression, naive bayes, decision tree and support vector machine, require me to truly understand the models. Although there are various packages to choose and one don't need to implement a model from scratch, knowing the details helps in parameter tuning and feature engineering and it also makes me creative in finding new ideas."



Anticipated questions

- My math skills are poor. What should I do?
 - Math is just a tool/language. Practice and become more comfortable with this language. A common pitfall is to try to intuit all the math; I recommend against this. For example, try to learn the notation behind probability first, before getting a strong intuitive grasp, and once you are more comfortable with the notation, then start searching for intuition
- I'm rusty at programming. Am I going to fail?
 - The amount you program is limited. I provide python code to read in data and do basic learning on that data. You will simply have to modify this code, likely amounting to at most 500 lines of code.



Anticipated questions

- I was hoping we would learn about topic x, but it looks like it is not listed. Can we learn about topic x?
 - With the foundations from this course, you will much more easily be able to go learn about more advanced topic x
 - Otherwise, particularly later in the course, come chat with me and we can discuss topic x



How do learning problems differ?

- They can be categorized across several dimensions
 - **Control versus prediction:** though a control algorithm will likely use predictions to improve decision-making (e.g., reinforcement learning)
 - **Supervised and unsupervised:** supervised learning is for prediction, unsupervised learning is usually for visualization or representation learning; some algorithms combine these two components
 - **How they are used:** empower decision-making of end user OR autonomously control system
 - ... there are more, but these are some main ones

How do the algorithms differ?

- **Algorithms** also differ in many ways, even for similar problems
 - Process data incrementally (as stream) or in batch
 - Low computation (or memory) versus heavy computation (or memory)
 - Data efficient (needs only a few samples to learn a good model)
 - Consistency: with more samples, model approaches “true” model
 - ... other common algorithmic distinctions, such as approximate vs. exact, or randomized vs deterministic



Let's look at some fun examples!

- Commute times (independent identically distributed (iid) data)
- Weather prediction (temporally connected data)
 - machine learning is often used for time series, but in the specific case of weather, mostly expert models appear to be used (for now...)
- Octopus arm simulator (machine learning for control)
 - we will not look at control algorithms; however, their development uses the fundamental concepts in this course



Commute times



LILAC PETAL
8830-4

9	—	—	10	8
5	10	11	14	11
13	11	12	9	11
10	11	13	—	12
11	13	10	11	9
14	9	12	10	11
11	10	12	23	12
8	14	12	10	8
—	11	11	—	9
—	11	12	10	12

PANTONE
UNIVERSITY

11	10	11	—	9	7	11	—	12	10
10	11	10	10	—	10	10	11	11	13
9	10	11	12	11	12	11	12	11	13
8	11	12	13	12	13	12	13	12	13
7	12	13	14	13	14	13	14	13	14
6	13	14	15	14	15	14	15	14	15
5	14	15	16	15	16	15	16	15	16
4	15	16	17	16	17	16	17	16	17
3	16	17	18	17	18	17	18	17	18
2	17	18	19	18	19	18	19	18	19
1	18	19	20	19	20	19	20	19	20

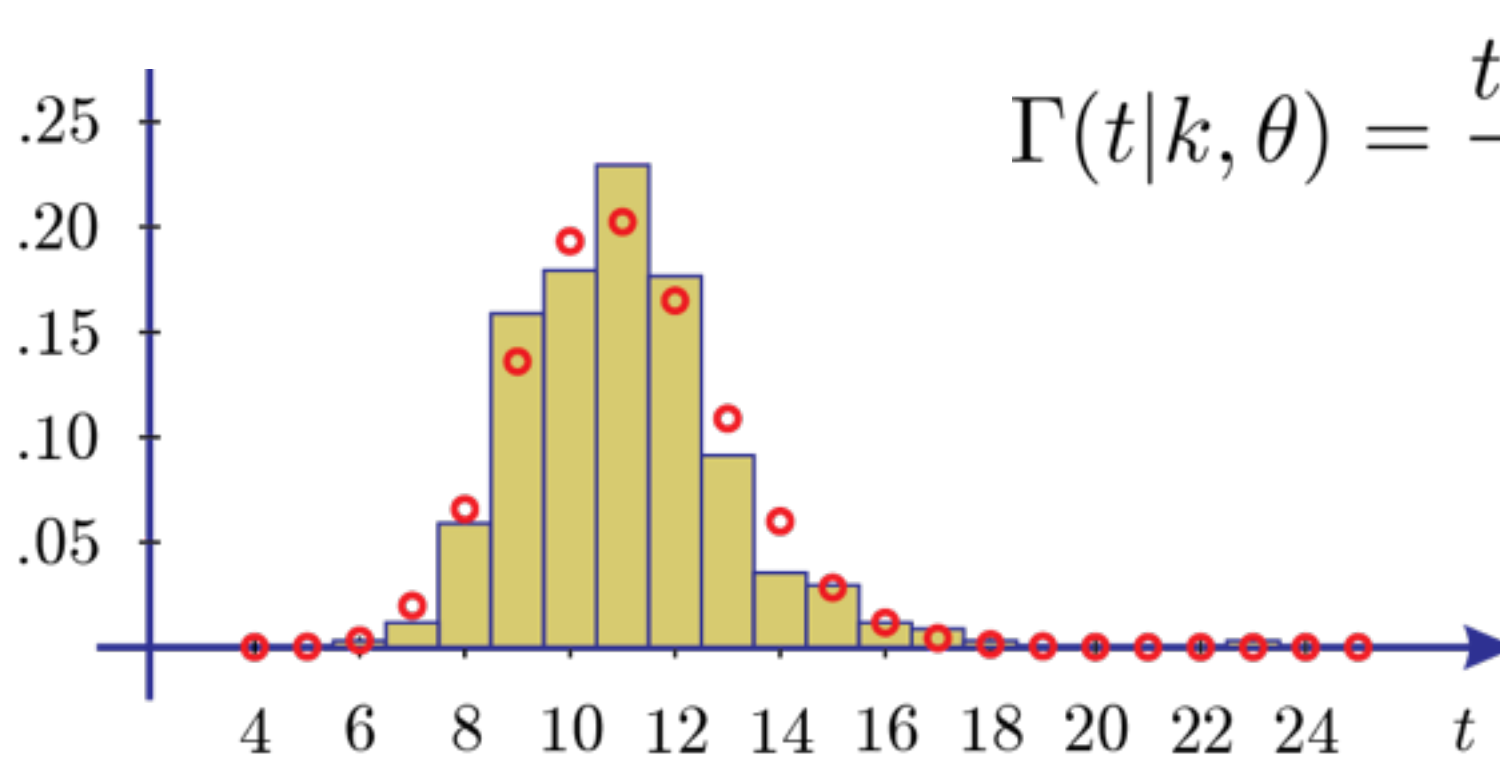
12	15	11	17	12	15	11	17
11	10	10	12	11	10	10	12
11	14	14	11	13	18	16	14
12	9	11	—	—	8	9	11
11	14	13	10	11	9	4	13
7	15	12	10	11	12	13	11
9	9	13	10	11	—	13	—
11	13	12	11	11	12	12	9
9	10	10	12	10	12	11	10
13	13	11	8	—	12	11	7
12	9	10	9	13	—	15	10
10	—	12	10	11	17	11	9
13	13	13	13	13	13	13	13

deas
COLOR

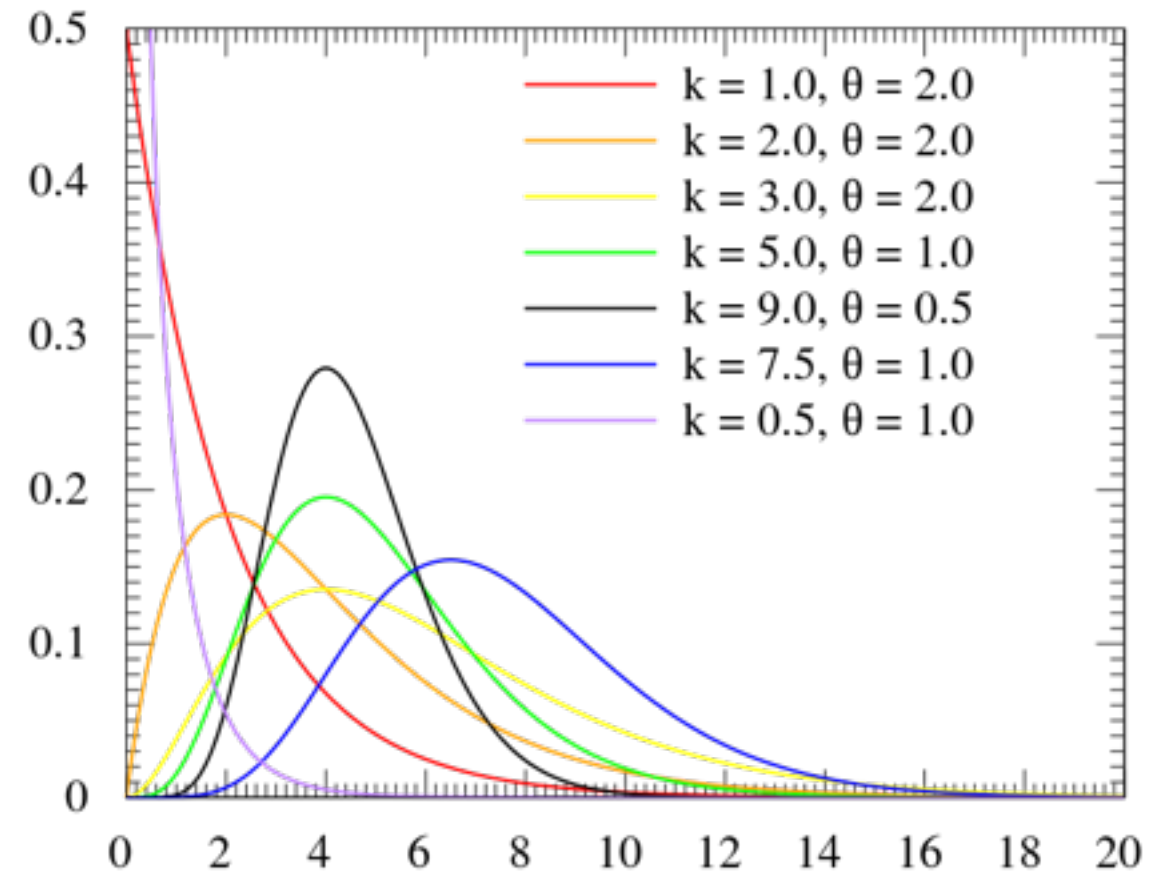
10	10	7
12	—	11
10	11	10
12	12	12
12	12	12
11	11	11
11	11	11
12	12	12
10	10	10
13	12	12
11	13	13

How might you try to predict your commute time for today?

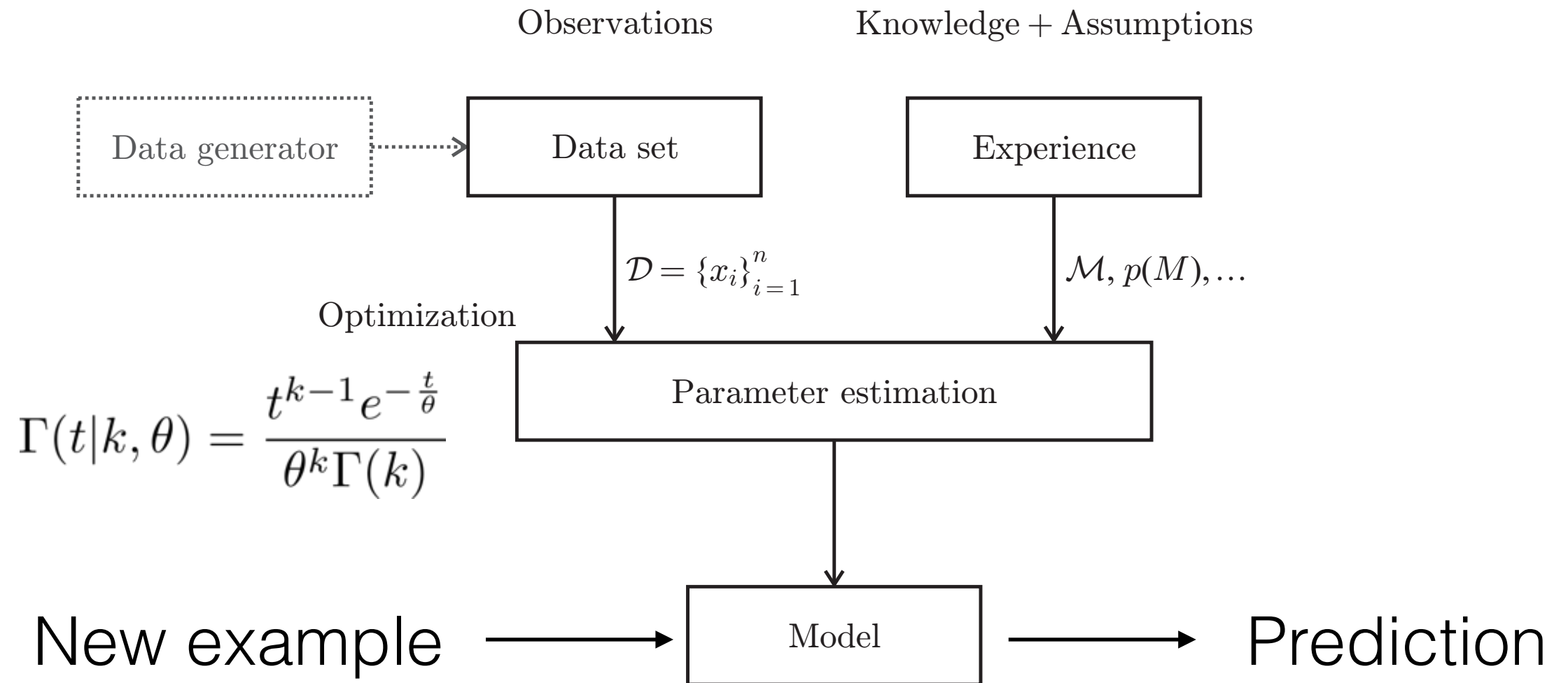
Commute times (2)



$$\Gamma(t|k, \theta) = \frac{t^{k-1} e^{-\frac{t}{\theta}}}{\theta^k \Gamma(k)}$$



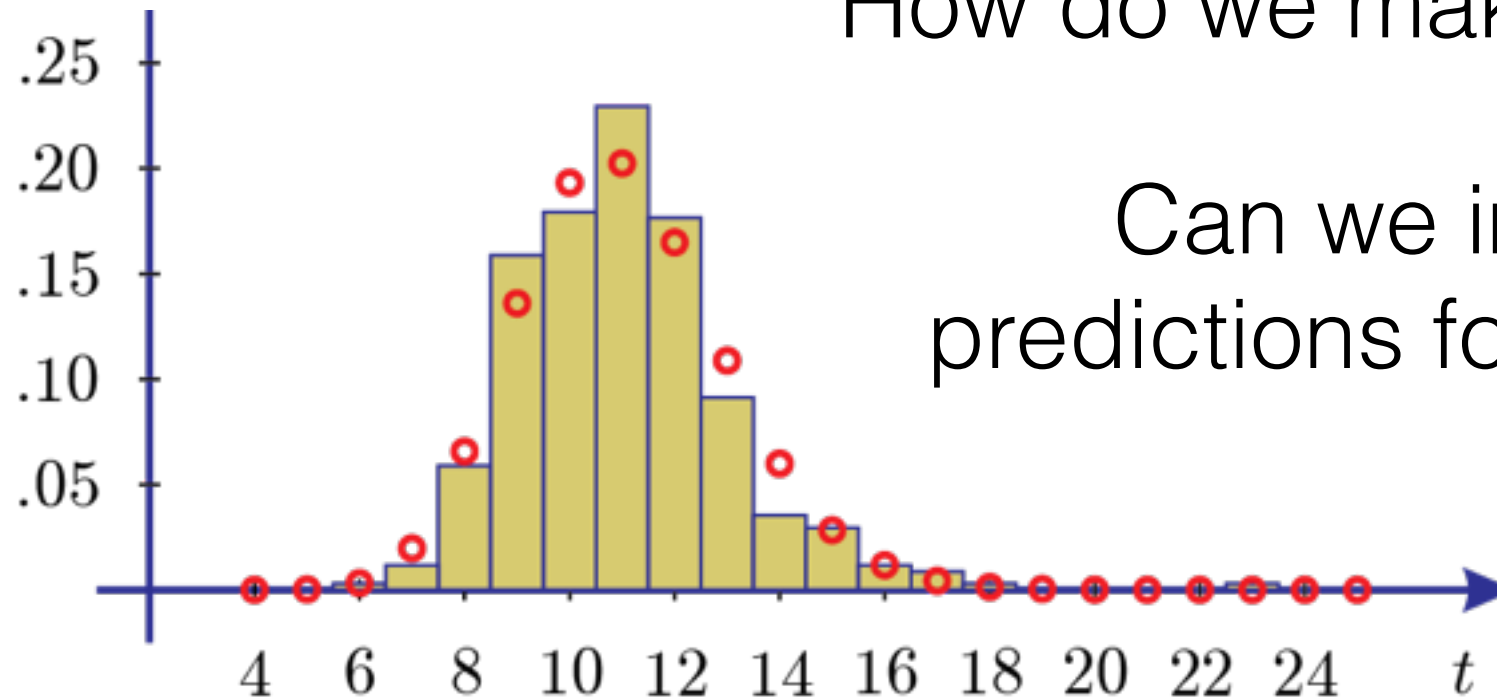
Summarized flow



Commute times (3)

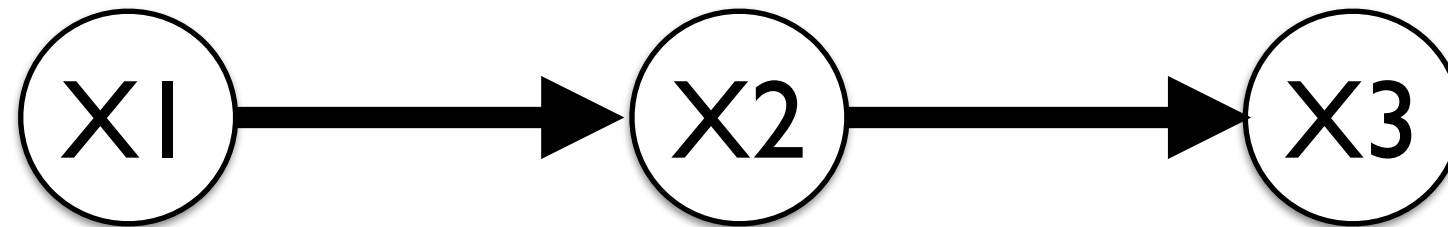
How do we make a prediction?

Can we improve our predictions for this question?



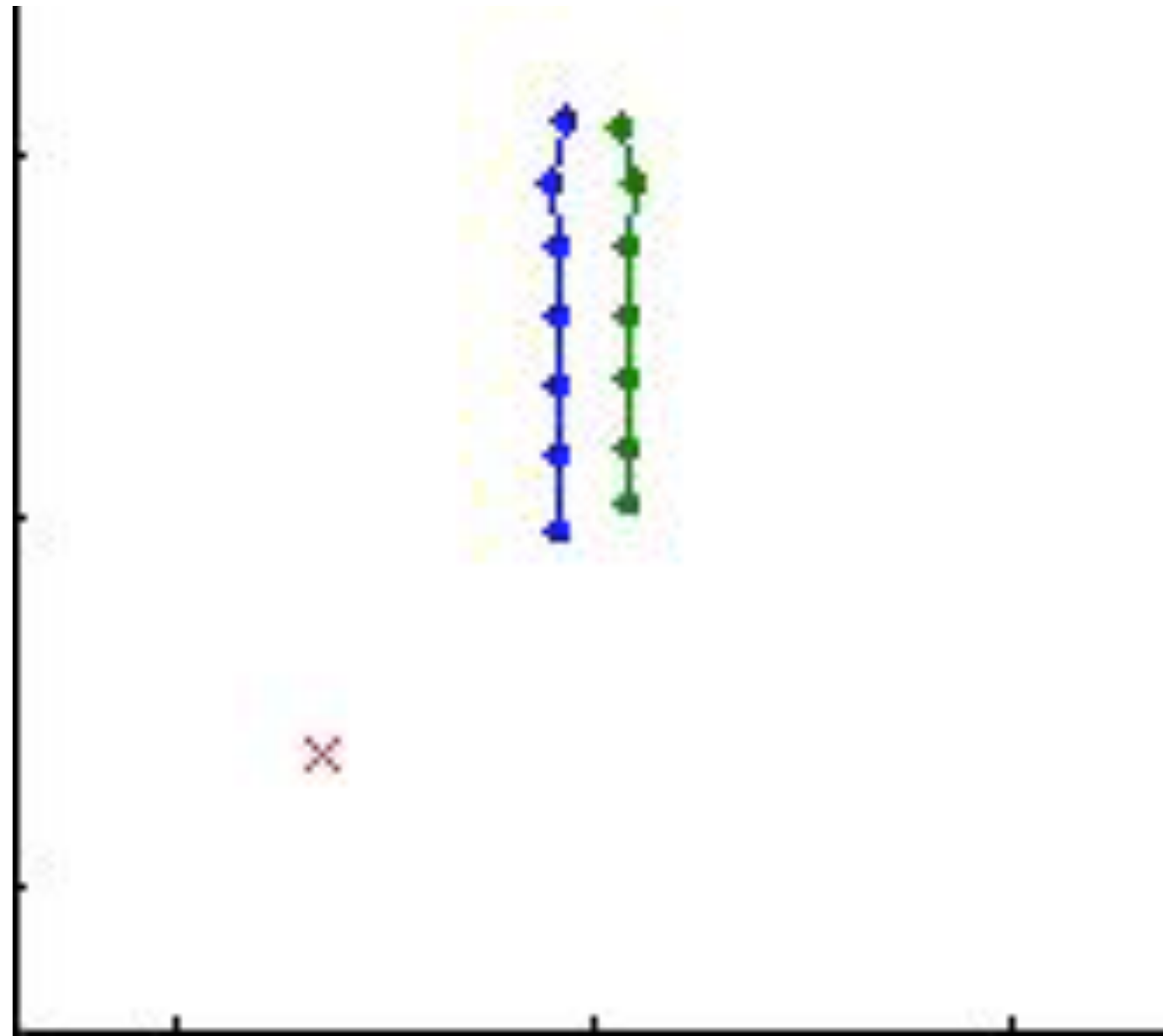
$$\Gamma(t|k, \theta) = \frac{t^{k-1} e^{-\frac{t}{\theta}}}{\theta^k \Gamma(k)}$$

Weather prediction (time series)



- Imagine we want to predict the probability of rain tomorrow, 2 days from now, 3 days, ...
- One common strategy for time series is to use the last p points as features to predict the next point, 2 points into future, etc.
- What other strategies can you imagine?
- How do you predict a probability value, rather than say a binary value (0 or 1) or a real value?
 - Hint: these are things we will learn

Octopus arm (control)



Predict (long-term) value of action given current observations about position of arm points