CS 1678/2078: Deep Learning

Introduction

Prof. Adriana Kovashka
University of Pittsburgh
January 8, 2024

About the Instructor



Born 1985 in Sofia, Bulgaria



Got BA in 2008 at Pomona College, CA (Computer Science & Media Studies)



Got PhD in 2014 at University of Texas at Austin (Computer Vision)

Course Info

- **Course website:** https://people.cs.pitt.edu/~kovashka/cs1678_sp24
- **Instructor**: Adriana Kovashka (kovashka@cs.pitt.edu)
  → Use "CS1678" at the beginning of your Subject
- **Office**: 5325 Sennott Square
- **Class:** Mon/Wed, 9:30am-10:45am
- **Office hours**: Mon/Wed, 11am-12:30pm
- **TA:** Cagri Gungor (cagri.gungor@pitt.edu)
- **TA's office:** 5501 Sennott Square
- **TA's office hours:** TBD (Do Doodle by end of Jan. 12: https://www.when2meet.com/?22476313-7WWAU)

Course Goals

- To develop intuitions for machine learning techniques and challenges, in the context of deep neural networks
- To learn the basic techniques, including the math behind basic neural network operations
- To become familiar with advances/specialized neural frameworks (e.g. convolutional/recurrent/transformer)
- To understand advantages/disadvantages of methods
- To practice implementing and using these techniques for simple problems
- To develop practical solutions for one real problem via course project (CS 2078 section)

Textbooks

- Ian Goodfellow, Yoshua Bengio, Aaron Courville. *Deep Learning.* online version
- Richard Szeliski. *Computer Vision: Algorithms and Applications.* online version (second edition)
- Aston Zhang, Zack C. Lipton, Mu Li, and Alex J. Smola. *Dive into Deep Learning.* online version
- Additional readings from papers
- Important: Your notes from class are your best study material, slides are not complete with notes

Programming

- Languages/frameworks: Python, NumPy, PyTorch
- NumPy tutorial **(go through at home)**: http://cs231n.github.io/python-numpy-tutorial/
- NumPy for Matlab users: https://docs.scipy.org/doc/numpy/user/numpy-for-matlab-users.html
- The TA will do a PyTorch tutorial
- Computing resource: Google Colab (free tier; can reimburse higher tier for course projects)

Course Structure

- Lectures
- Five programming assignments
- Exams
- Course project (teams of 2-3 students)
  - Proposal, two reports, presentation
  - Check course website for detailed description
- Participation (incl. quizzes on Canvas)

Tips for a successful project

- From your perspective:
  - Learn something
  - Try something out for a real problem

Tips for a successful project

- From your classmates' perspective:
  - Hear about a niche of DL we haven't covered, or learn about a niche of DL in more depth
  - Hear about challenges and how you handled them, that they can use in their own work
  - Listen to an engaging presentation

Tips for a successful project

- From my perspective:
  - Hear about the creative solutions you came up with to handle challenges
  - Hear your perspective on a topic that I care about
  - Co-author a publication with you, potentially with a small amount of follow-up work – looks good on your CV!

Tips for a successful project

- Overall
  - Don’t reinvent the wheel – your audience will be bored
  - But it’s ok to adapt an existing method to a new domain/problem…
  - If you show interesting experimental results…
  - You analyze them and present them in a clear and engaging fashion

Policies and Schedule

See course website!

Should I take this class?

- It will be a lot of work!
  - I expect you'll spend **6-8 hours** on homework or the project each week
  - But you will learn a lot
- Some parts will be hard and require that you pay close attention!
  - Quizzes help ensure we're on the same page
  - Use instructor's and TA's office hours!

Your Homework

- Read entire course website
- Fill out poll for TA’s office hours
- Do first reading
- **Go through NumPy tutorial, ask TA if questions**

Questions?

Plan for Today

- Blitz introductions
- What is deep learning
  - Example problems and challenges
- Machine learning overview
  - ML framework
  - Linear classifiers
  - Elements of a ML algorithm
  - Evaluation and generalization
- Review: Linear algebra and calculus

Blitz introductions (10 sec)

- What is your name?
- What one thing outside of school are you passionate about?
- Have you taken an artificial intelligence class before? Which one(s)?
- What do you hope to get out of this class?

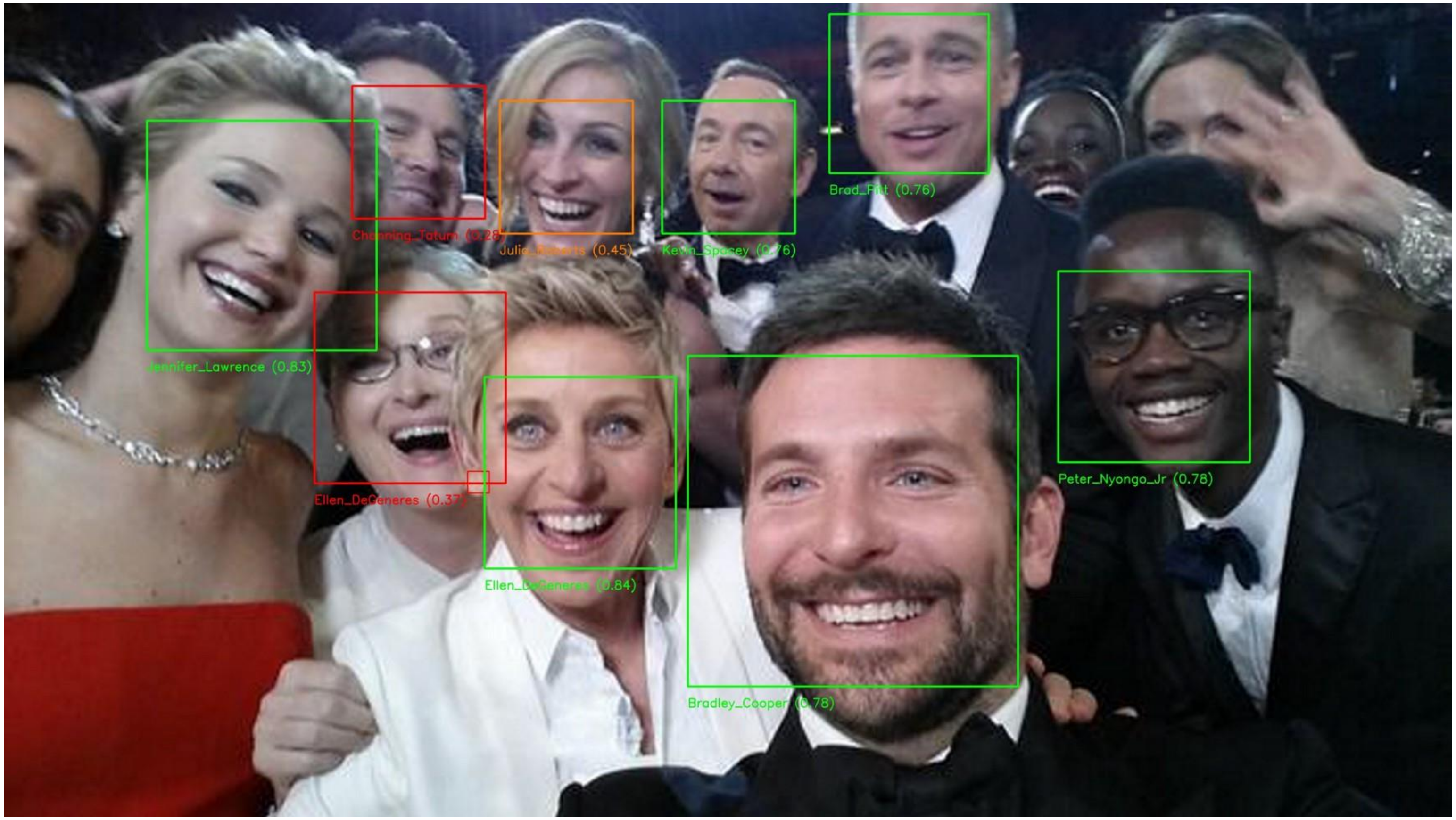
- **When you speak, always say your name**

What is deep learning?

- One approach to finding patterns and relationships in data
- Finding the right representations of the data, that enable correct automatic performance of a given task
- Examples: Learn to predict the category (label) of an image, learn to translate between languages

Example deep learning tasks

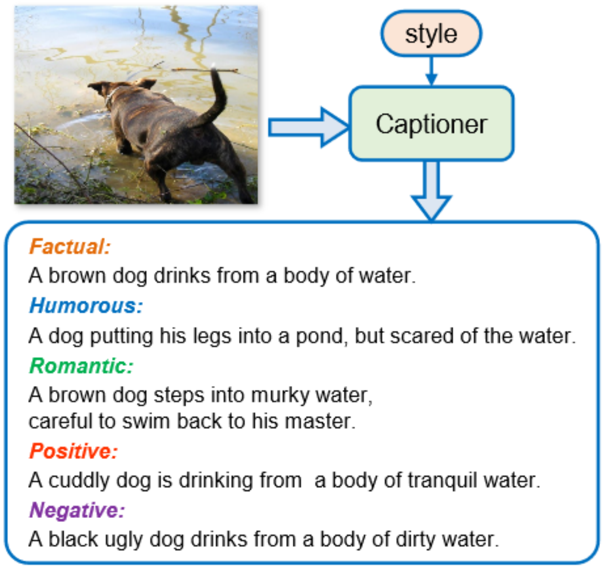
- Face recognition

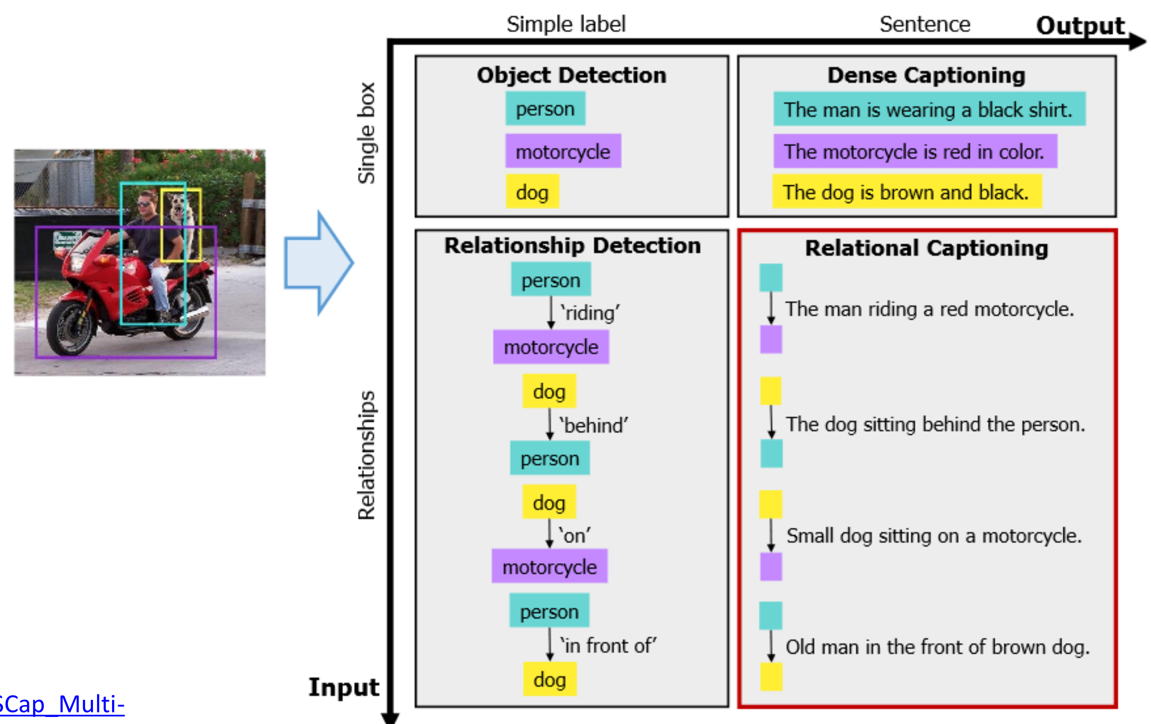


https://towardsdatascience.com/an-intro-to-deep-learning-for-face-recognition-aa8dfbbc51fb

Example deep learning tasks

- Image captioning

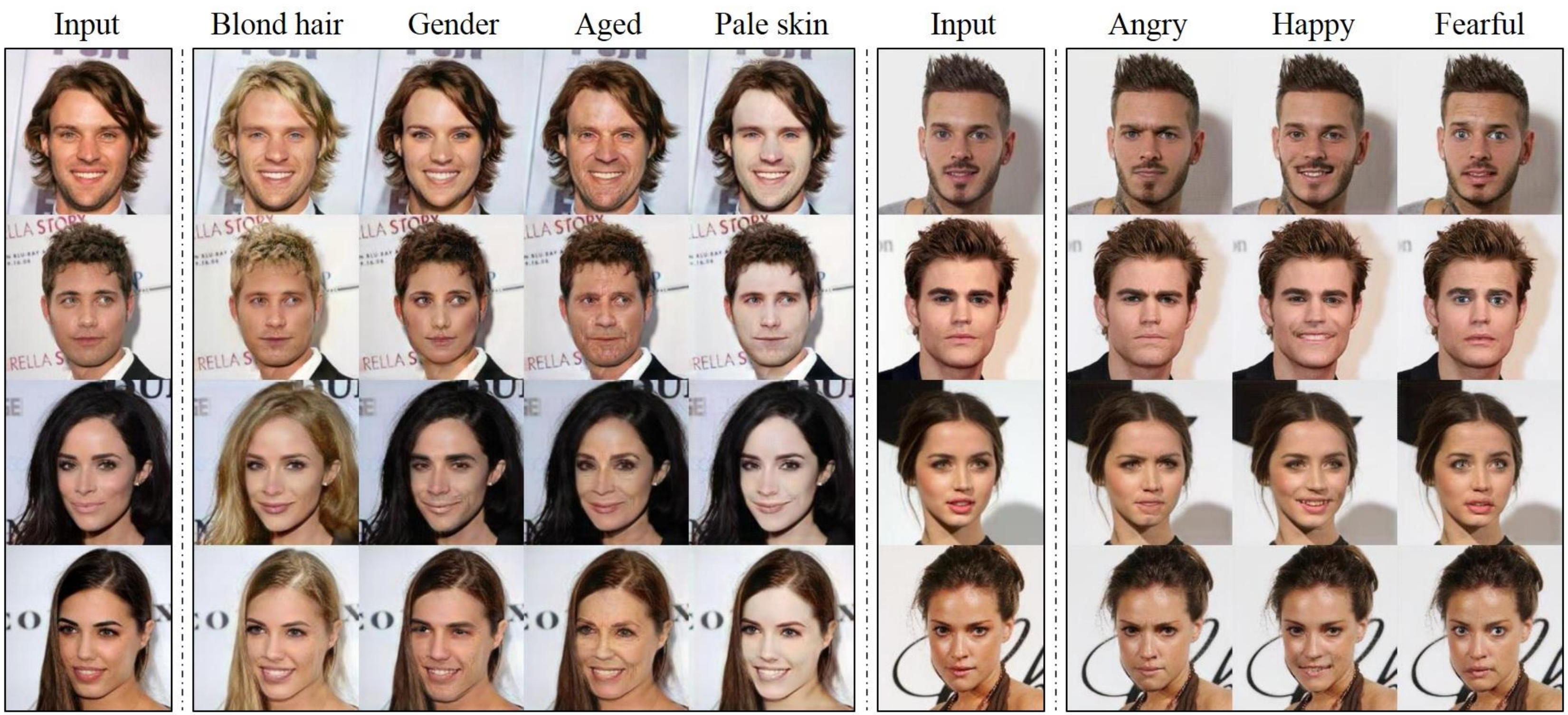




http://openaccess.thecvf.com/content_CVPR_2019/papers/Guo_MSCap_Multi-Style_Image_Captioning_With_Unpaired_Stylized_Text_CVPR_2019_paper.pdf
http://openaccess.thecvf.com/content_CVPR_2019/papers/Kim_Dense_Relational_Captioning_Triple-Stream_Networks_for_Relationship-Based_Captioning_CVPR_2019_paper.pdf

Example deep learning tasks

- Image generation



Choi et al., "StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation", CVPR 2018

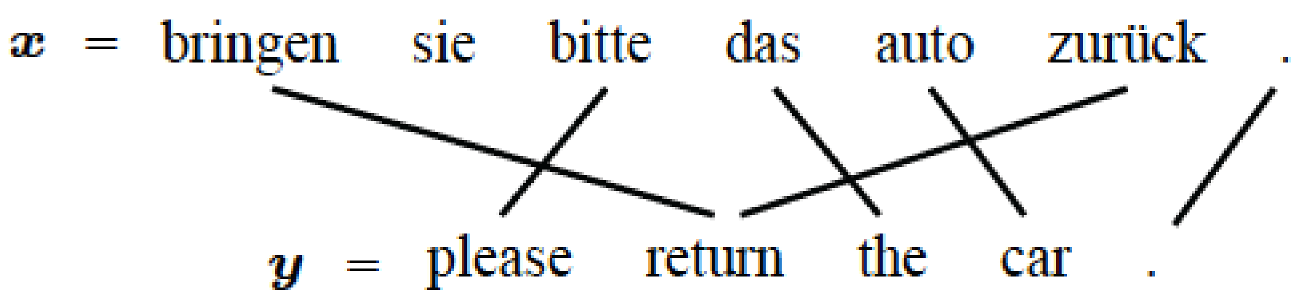
Example deep learning tasks

- Fake news generation

https://www.youtube.com/watch?v=-QvIX3cY4lc

Example deep learning tasks

- Machine translation



Slide credit: Carlos Guestrin

Example deep learning tasks

- Speech recognition

Slide credit: Carlos Guestrin

Example deep learning tasks

- Text generation

```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.

Clown:
Come, sir, I will make did behold your worship.

VIOLA:
I'll drink it.
```

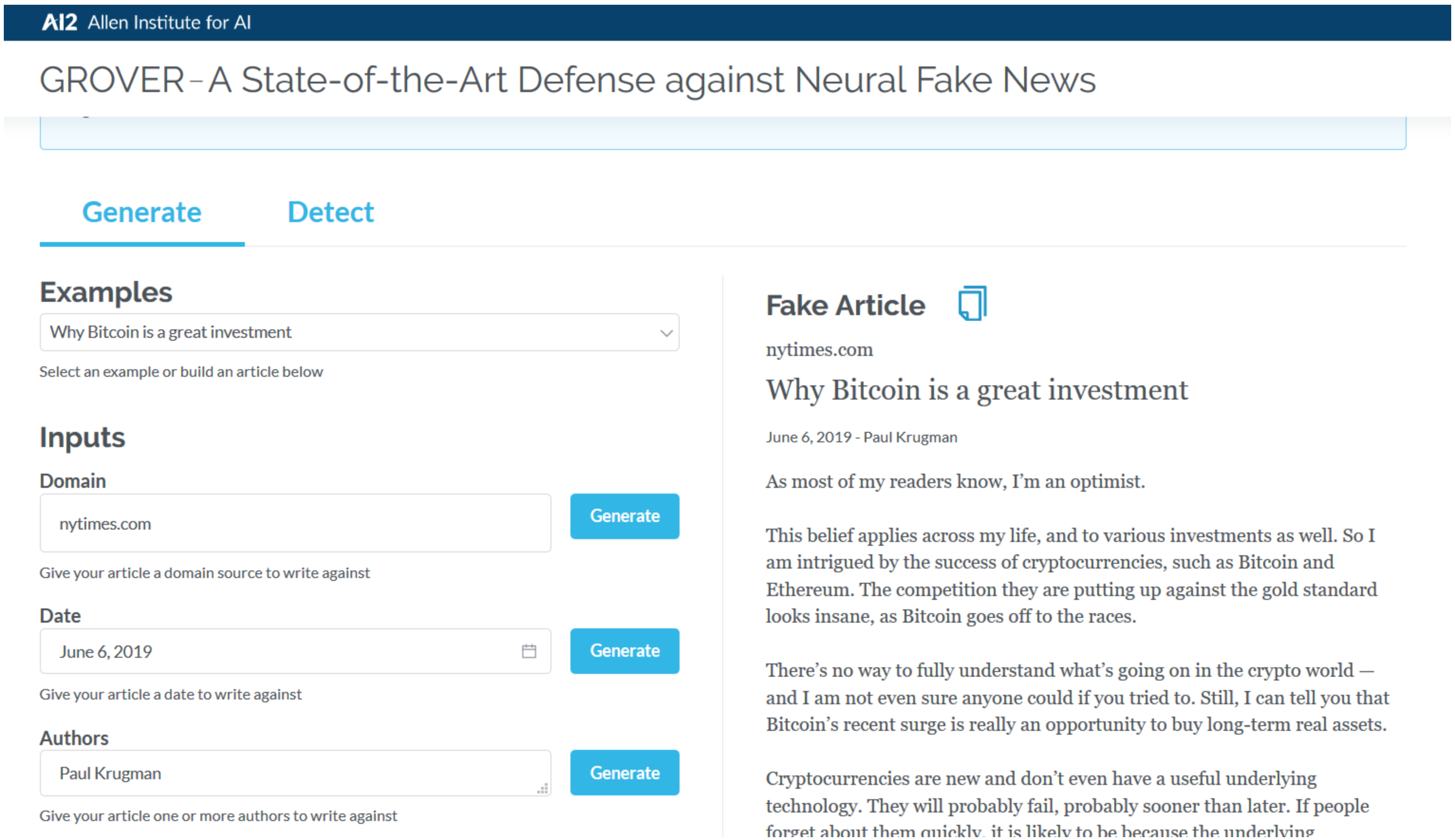
```
VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father's world;
When I was heaven of presence and our fleets,
We spare with hours, but cut thy council I am great,
Murdered and by thy master's ready there
My power to give thee but so much as hell:
Some service in the noble bondman here,
Would show him to her wine.

KING LEAR:
O, if you were a feeble sight, the courtesy of your law,
Your sight and several breath, will wear the gods
With his heads, and my hands are wonder'd at the deeds,
So drop upon your lordship's head, and your opinion
Shall be against your honour.
```

Andrej Karpathy

Example deep learning tasks

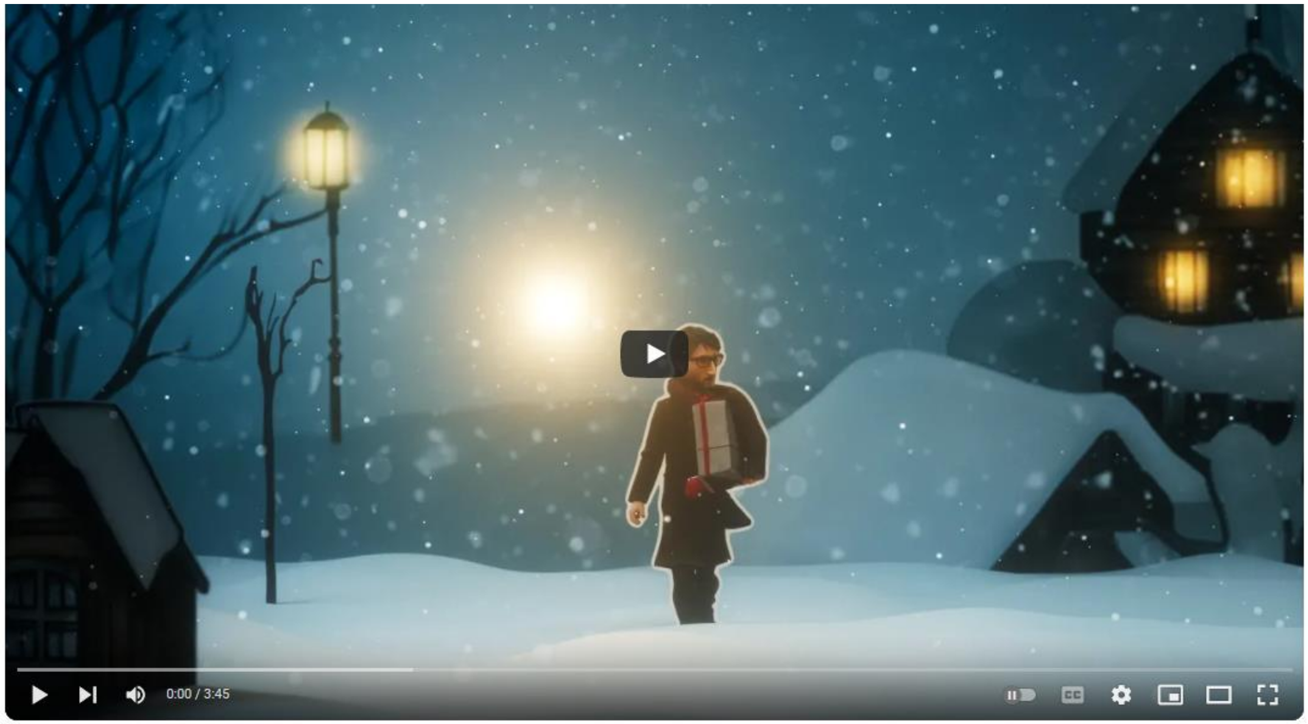
- Fake news generation and detection



https://grover.allenai.org/detect

Example deep learning tasks

- Art?



Steven Wilson - December Skies

Steven Wilson
276K subscribers
Subscribe

9K | Share | Clip | Save

172,117 views Dec 14, 2023
Challenged by a friend last week on why I'd never done a Christmas song, I protested it wasn't really me, and besides I didn't think I had it in me to write suitably Christmassy lyrics. "No problem", he said, "we'll get ChatGPT to do it for you!", upon which he instructed the AI to generate Christmas lyrics in the style of Steven Wilson. Which it proceeded to do so. By the yard, in fact, I was shocked at how instantaneous and easy it was to guide it ("don't actually mention Christmas" we said to it, "make it feel more lonely"...etc)

I'm still not entirely sure what I think about AI, the knee-jerk reaction is to consider it a threat to a musician like me, but it's clearly not going away, so this is my experiment to try embracing it and using it as a tool in the creative process. And of course also to have some fun doing something I would never normally have thought about doing, the archetypal seasonal song. December Skies is the result, swiftly recorded last weekend and written in collaboration with an Artificial Intelligence version of myself. I wonder what people will make of it. I wonder what I make of it.

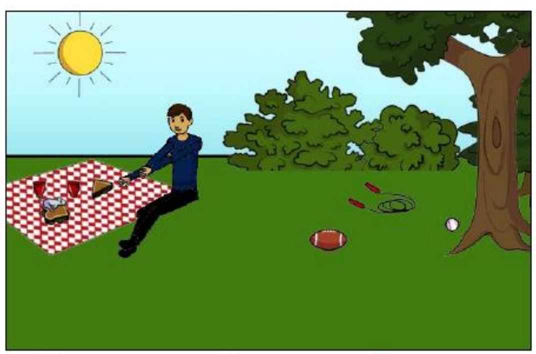
In keeping with the artificial intelligence theme, the visualiser for December Skies has also been generated using a purpose built AI system created by Miles Skarin. The video is based on imagery drawn from some of my previous videos then placed into 2D and 3D backdrops.

Example deep learning tasks

- Question answering

What color are her eyes?
What is the mustache made of?

How many slices of pizza are there?
Is this a vegetarian pizza?



Is this person expecting company?
What is just under the tree?

Does it appear to be rainy?
Does this person have 20/20 vision?

Agrawal et al., "[VQA: Visual Question Answering]", ICCV 2015

From Recognition to Cognition: Visual Commonsense Reasoning

Rowan Zellers♠ Yonatan Bisk♠ Ali Farhadi♠♡ Yejin Choi♠♡

♠Paul G. Allen School of Computer Science & Engineering, University of Washington

♡Allen Institute for Artificial Intelligence

visualcommonsense.com

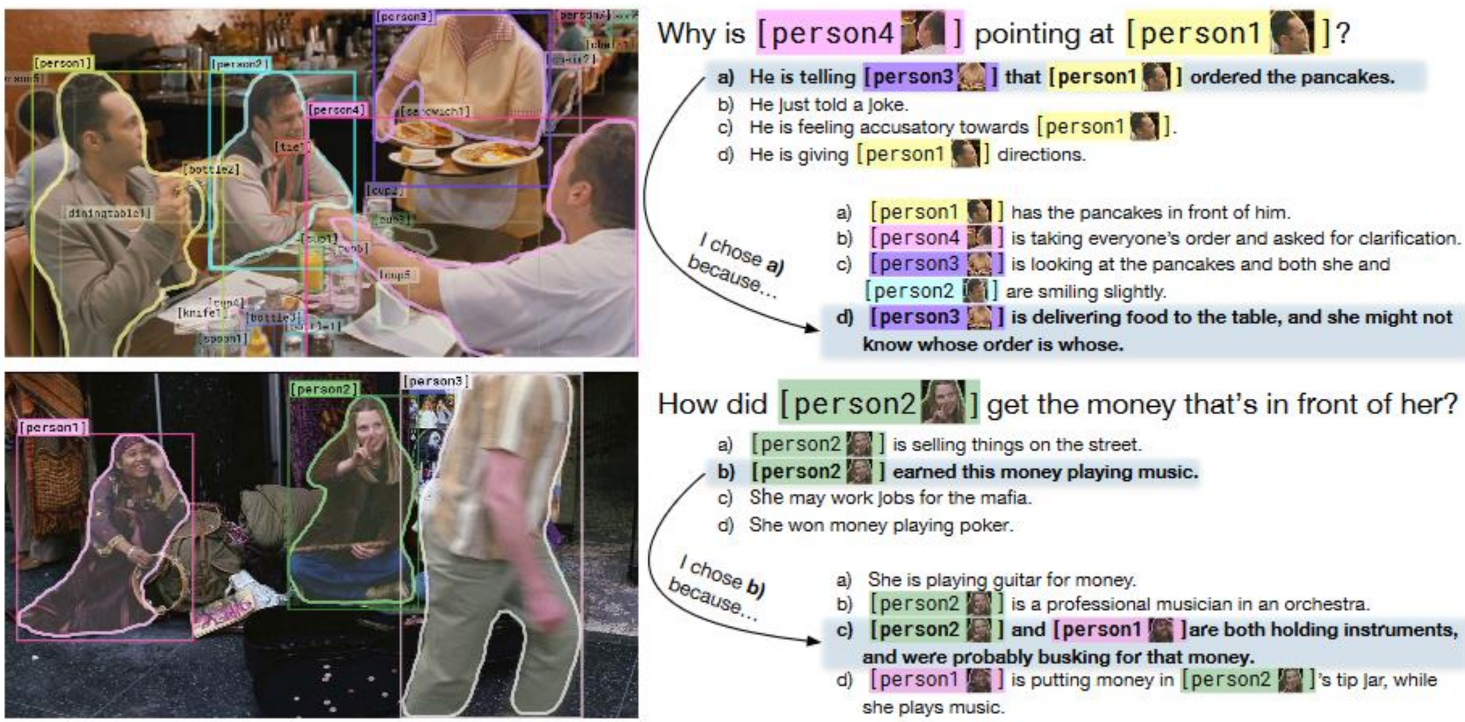


Figure 1: VCR: Given an image, a list of regions, and a question, a model must answer the question and provide a *rationale* explaining why its answer is right. Our questions challenge computer vision systems to go beyond recognition-level understanding, towards a higher-order cognitive and commonsense understanding of the world depicted by the image.

https://visualcommonsense.com/

Example deep learning tasks

- Robotic pets

https://www.youtube.com/watch?v=wE3fmFTtP9g

Example deep learning tasks

- Artificial general intelligence?

https://www.dailymail.co.uk/sciencetech/article-5287647/Humans-robot-second-self.html

Example deep learning tasks

- Why are these tasks challenging?
- What are some problems from everyday life that can be helped by deep learning?
- What are some ethical concerns about using deep learning?

Klingon vs Mlingon Classification

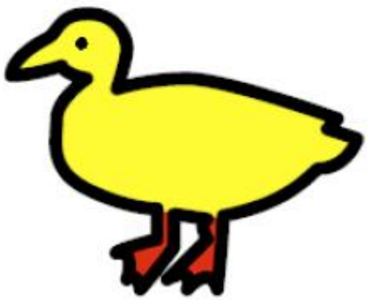
- Training Data
  - Klingon: klix, kour, koop
  - Mlingon: moo, maa, mou

- Testing Data: kap

- Which language? Why?

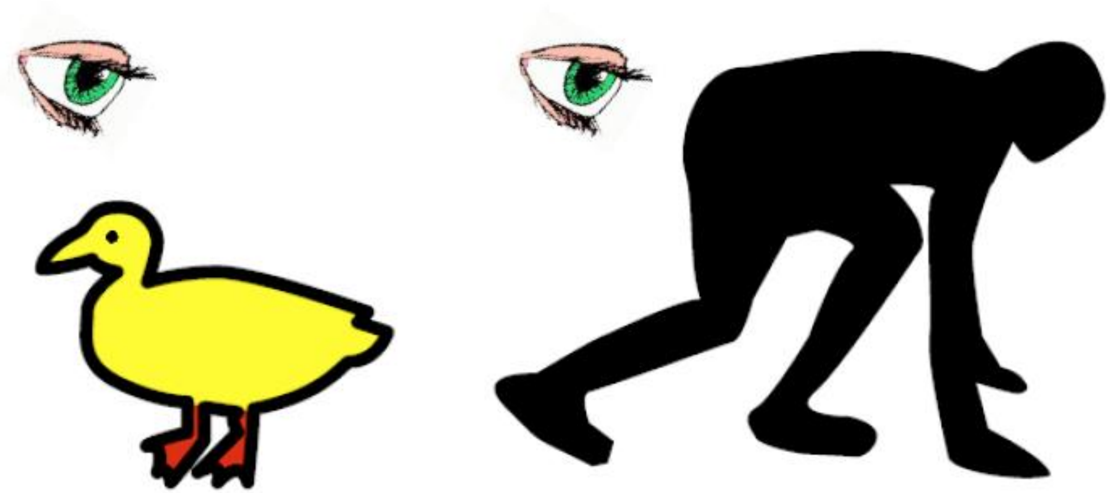
Slide credit: Dhruv Batra

“I saw her duck”



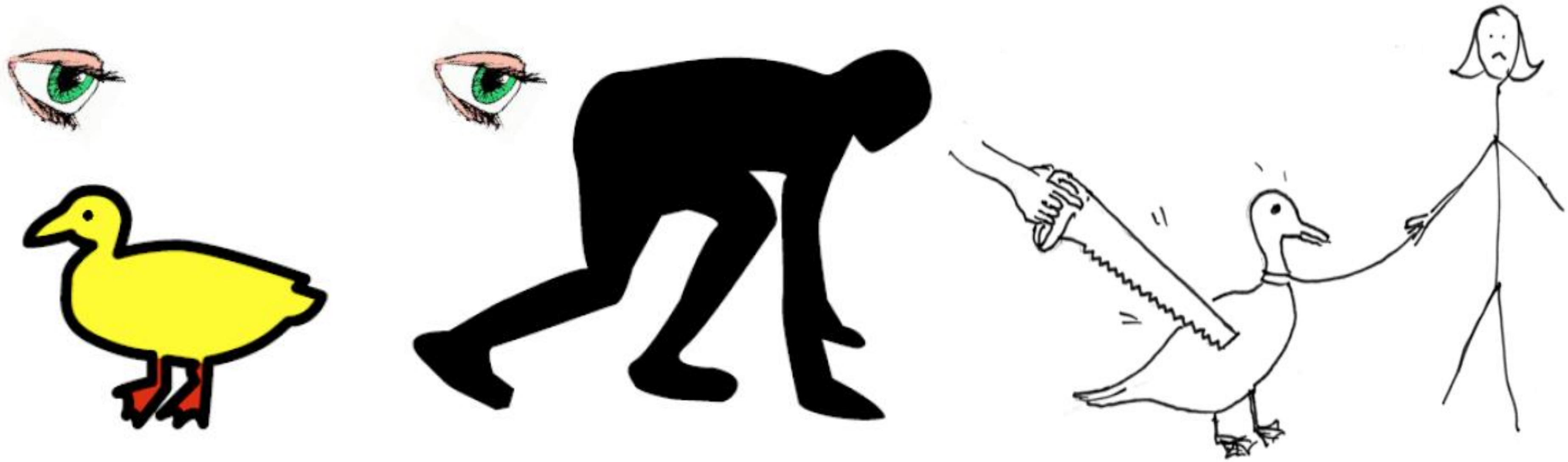
Slide credit: Dhruv Batra, figure credit: Liang Huang

"I saw her duck"



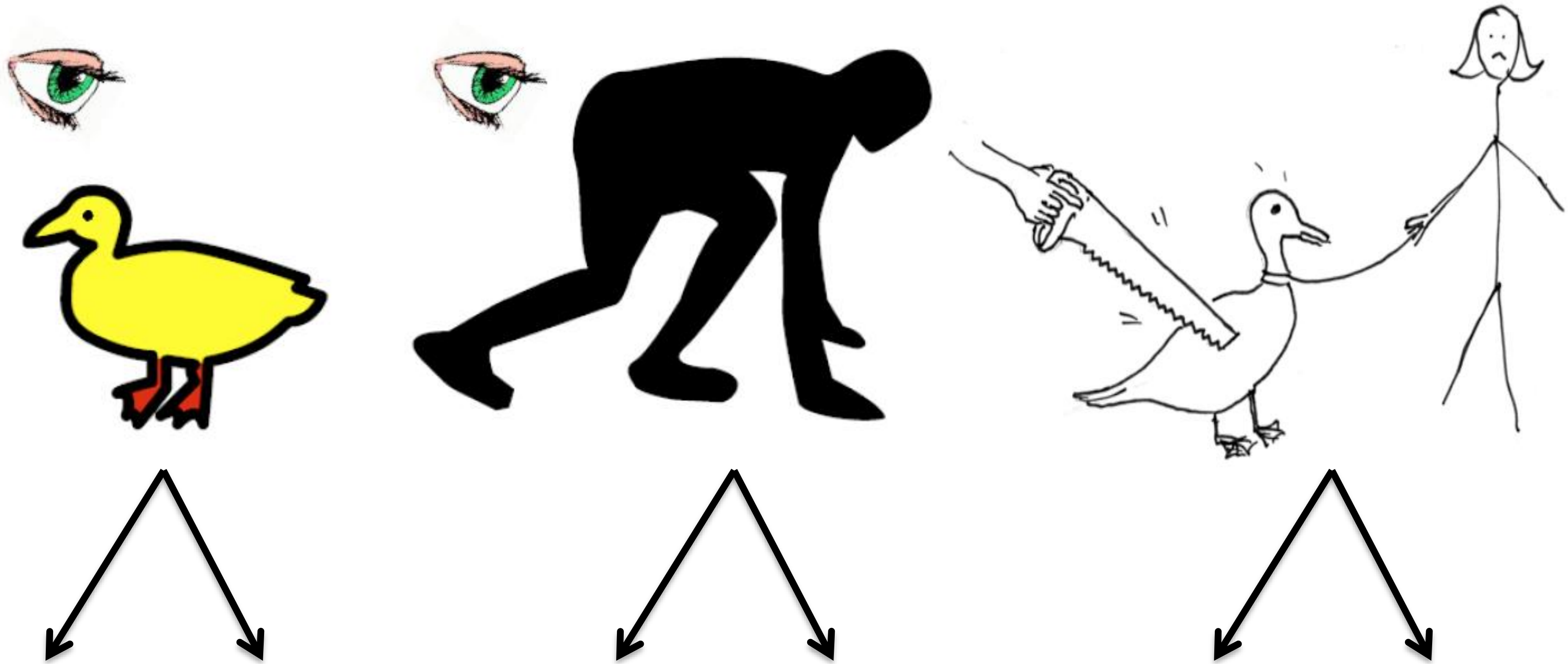
Slide credit: Dhruv Batra, figure credit: Liang Huang

“I saw her duck”



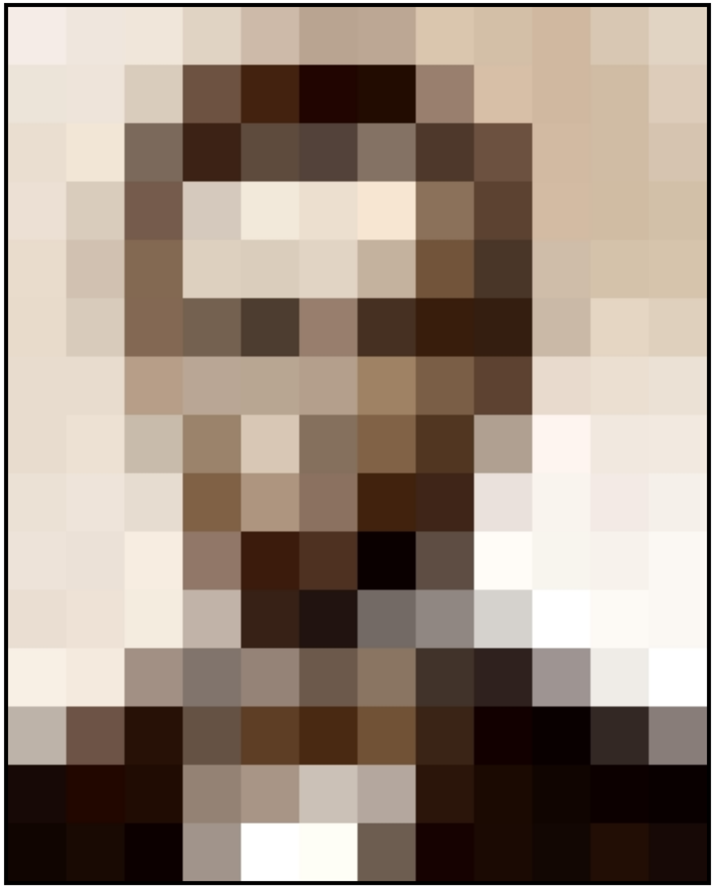
Slide credit: Dhruv Batra, figure credit: Liang Huang

"I saw her duck with a telescope…"



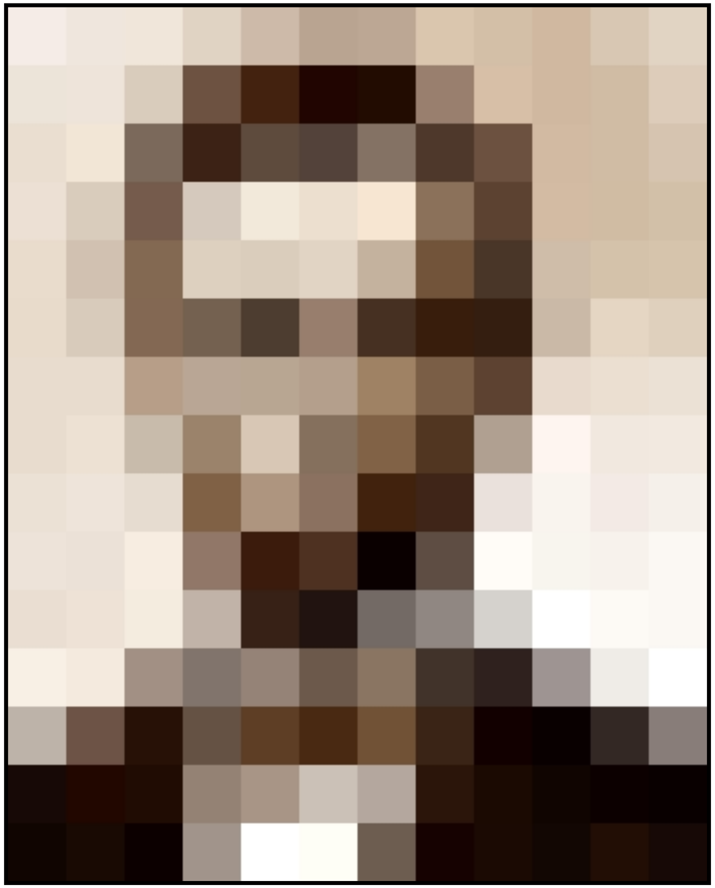
Slide credit: Dhruv Batra, figure credit: Liang Huang

What humans see



Slide credit: Larry Zitnick

What computers see



Slide credit: Larry Zitnick

Challenges

- Some challenges: ambiguity and context
- Machines take data representations too literally
- Humans are much better than machines at generalization, which is needed since test data will rarely look exactly like the training data

Machine Learning Overview

- Deep learning is a specific group of algorithms falling in the broader realm of machine learning
- All ML/DL algorithms roughly match schema:
  - Learn a mapping from input to output f: x → y
  - x: image, text, etc.
  - y: {cat, notcat}, {1, 1.5, 2, …}, etc.
  - f: this is where the magic happens

Machine Learning Overview

$$y' = f(\mathbf{x})$$

output — prediction function — input

- **Training:** given a *training set* of labeled examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, estimate the prediction function f by minimizing the prediction error on the training set
- **Testing:** apply f to a never before seen *test example* $\mathbf{x}$ and output the predicted value $y' = f(\mathbf{x})$

Slide credit: L. Lazebnik

Machine Learning Overview

- Example:
  - Predict whether an email is spam or not:

Sebring, Tracy
To: Batra, Dhruv
ECE 4424 proposal

CUSP has approved ECE 4424 with the following changes: Can you copy of the proposal with these items addressed? (see below)
Thanks!!!
Tracy

vs

nadia bamba January 19, 2015 5:57 AM
To: undisclosed recipients: ; Hide Details
Reply-To: nadia bamba
From Miss Nadia BamBa,

From Miss Nadia BamBa,

Greeting, Permit me to inform you of my desire of going into business relationship with you. I am Nadia BamBa the only Daughter of late Mr and Mrs James BamBa, My father was a director of cocoa merchant in Abidjan, the economic capital of Ivory Coast before he was poisoned to death by his business associates on one of their outing to discus on a business deal. When my mother died on the 21st October 2002, my father took me very special because i am motherless.

Before the death of my father in a private hospital here in Abidjan, He secretly called me on his bedside and told me that he had a sum of $6, 8000.000(SIX Million EIGHT HUNDRED THOUSAND), Dollars) left in a suspense account in a Bank here in Abidjan, that he used my name as his first Daughter for the next of kin in deposit of the fund.

He also explained to me that it was because of this wealth and some huge amount of money That his business associates supposed to balance him from the deal they had that he was poisoned by his business associates, that I should seek for a God fearing foreign partner in a country of my choice where I will transfer this money and use it for investment purposes, (such as real estate Or Hotel management).please i am honourably seeking your assistance in the following ways.

1) To provide a Bank account where this money would be transferred to.
2) To serve as the guardian of this Money since I am a girl of 19 years old.
3)Your private phone number's and your family background' s that we can know each order more.

Figures from Dhruv Batra

Machine Learning Overview

- Example:
  - Predict whether an email is spam or not.
  - **x** = words in the email, *multi-hot* representation of size |V|x1, where V is the full vocabulary and x(j) = 1 iff word *j* is mentioned
  - y = 1 (if spam) or 0 (if not spam)
  - y’ = f(**x**) = $\mathbf{w}^T \mathbf{x}$
    - **w** is a vector of the same size as **x**
    - One weight per dimension of **x** (i.e. one weight per word)
    - Weight can be positive, zero, negative…
    - What might these weights look like?

Simple strategy: Let's count!

nadia bamba
To: undisclosed recipients: ;
Reply-To: nadia bamba
From Miss Nadia BamBa,

From Miss Nadia BamBa,

Greeting, Permit me to inform you of my desire of going i
Nadia BamBa the only Daughter of late Mr and Mrs Jame
cocoa merchant in Abidjan, the economic capital of Ivory
his business associates on one of their outing to discus c
on the 21st October 2002, my father took me very specia

Before the death of my father in a private hospital here i
bedside and told me that he had a sum of $6, 8000.000(S
Dollars) left in a suspense account in a Bank here in Abic
Daughter for the next of kin in deposit of the fund.

Sebring, Tracy
To: Batra, Dhruv
ECE 4424 proposal

CUSP has approved ECE 4424 with the following changes: Can y
copy of the proposal with these items addressed? (see below)
Thanks!!!
Tracy

This is X

$$\begin{pmatrix} \text{free} & 100 \\ \text{money} & 2 \\ \vdots & \vdots \\ \text{account} & 2 \\ \vdots & \vdots \end{pmatrix}$$

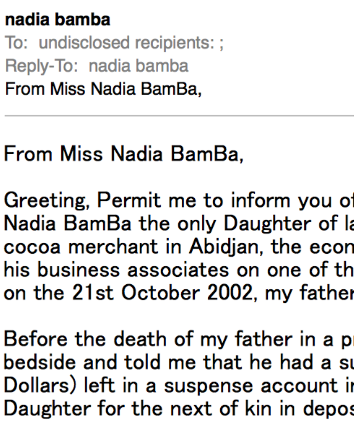
$$\begin{pmatrix} \text{free} & 1 \\ \text{money} & 1 \\ \vdots & \vdots \\ \text{account} & 2 \\ \vdots & \vdots \end{pmatrix}$$

This is Y

= 1 or 0?

Adapted from Dhruv Batra, Fei Sha

Weigh counts and sum to get prediction


nadia bamba
To: undisclosed recipients: ;
Reply-To: nadia bamba
From Miss Nadia BamBa,

From Miss Nadia BamBa,

Greeting, Permit me to inform you of
Nadia BamBa the only Daughter of la
cocoa merchant in Abidjan, the econ
his business associates on one of th
on the 21st October 2002, my father

Before the death of my father in a p
bedside and told me that he had a su
Dollars) left in a suspense account ir
Daughter for the next of kin in depos

$$\begin{pmatrix} 100 \times 0.2 \\ 2 \times 0.3 \\ \vdots \\ 2 \times 0.3 \\ \vdots \end{pmatrix}$$

$$\begin{pmatrix} \text{free} & 100 \\ \text{money} & 2 \\ \vdots & \vdots \\ \text{account} & 2 \\ \vdots & \vdots \end{pmatrix}$$

This is a *linear classifier*

Adapted from Dhruv Batra, Fei Sha

Machine Learning Overview

- Example:
  - Apply a prediction function to an image to get the desired label output:

f() = "apple"

f() = "tomato"

f() = "cow"

Slide credit: L. Lazebnik

Machine Learning Overview

- Example:
  - **x** = pixels of the image (concatenated to form a vector)
  - y = integer (1 = apple, 2 = tomato, etc.)
  - y’ = f(**x**) = $\mathbf{w}^T \mathbf{x}$
    - **w** is a vector of the same size as **x**
    - One weight per each dimension of **x** (i.e. one weight per pixel)

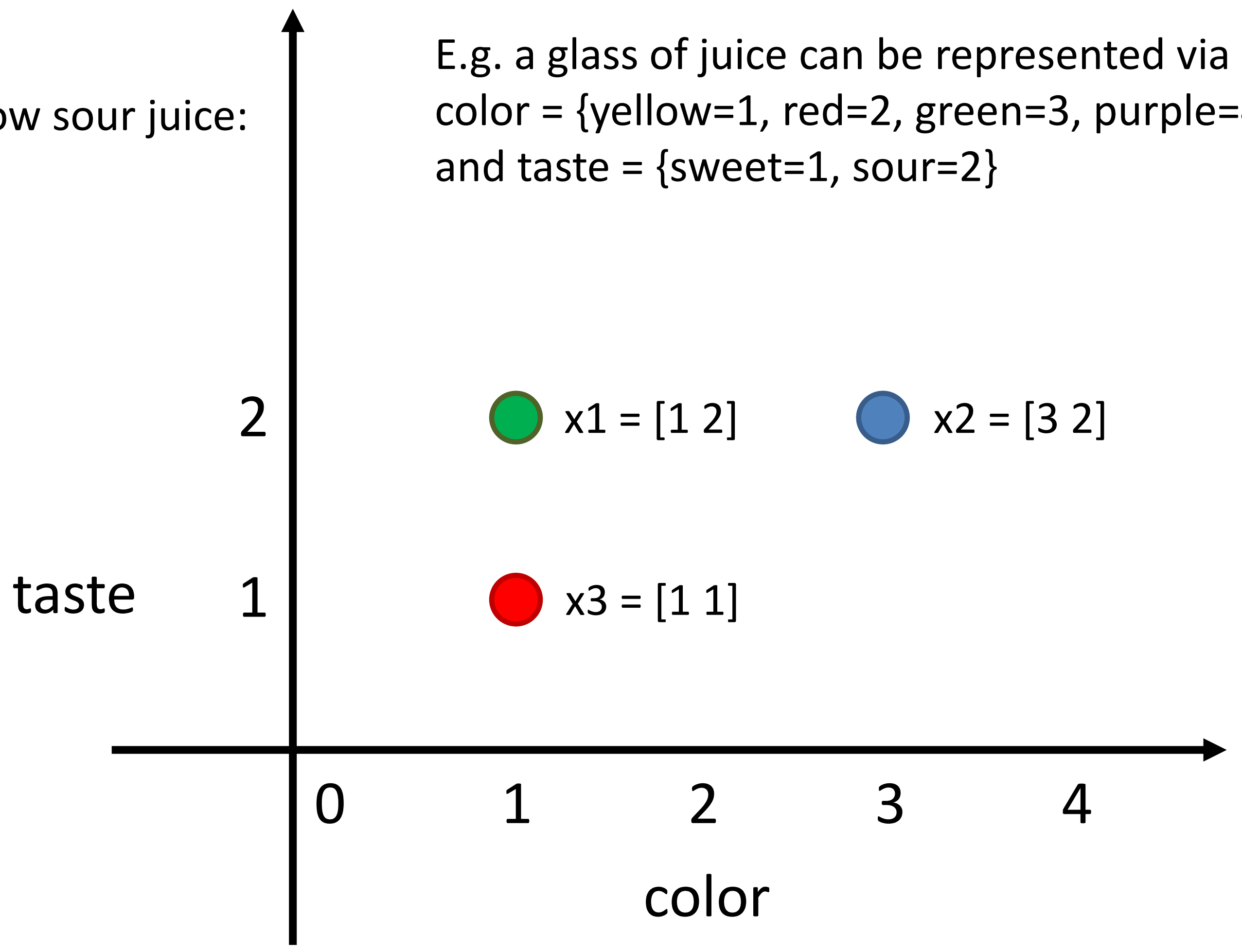
Feature representation (**x**)

- A vector representing measurable characteristics of a data sample we have
- E.g. a glass of juice can be represented via its color = {yellow=1, red=2, green=3, purple=4} and taste = {sweet=1, sour=2}
- For a given glass i, this can be represented as a vector: $\mathbf{x}_i$ = [3 2] represents green sour juice
- For D features, this defines a D-dimensional space where we can measure similarity between samples

Example: Feature representation

E.g. a glass of juice can be represented via its color = {yellow=1, red=2, green=3, purple=4} and taste = {sweet=1, sour=2}

Which is the yellow sour juice: x1, x2 or x3?



Norms

- L1 norm

$$\|\boldsymbol{x}\|_1 := \sum_{i=1}^{n} |x_i|$$

- L2 norm

$$\|\boldsymbol{x}\| := \sqrt{x_1^2 + \cdots + x_n^2}$$

- L^p norm (for real numbers $p \geq 1$)

$$\|\mathbf{x}\|_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

Distances

- L1 (Manhattan) distance

$$d_1(\mathbf{p},\mathbf{q}) = \|\mathbf{p}-\mathbf{q}\|_1 = \sum_{i=1}^{n} |p_i - q_i|,$$

- L2 (Euclidean) distance

$$d(\mathbf{p},\mathbf{q}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

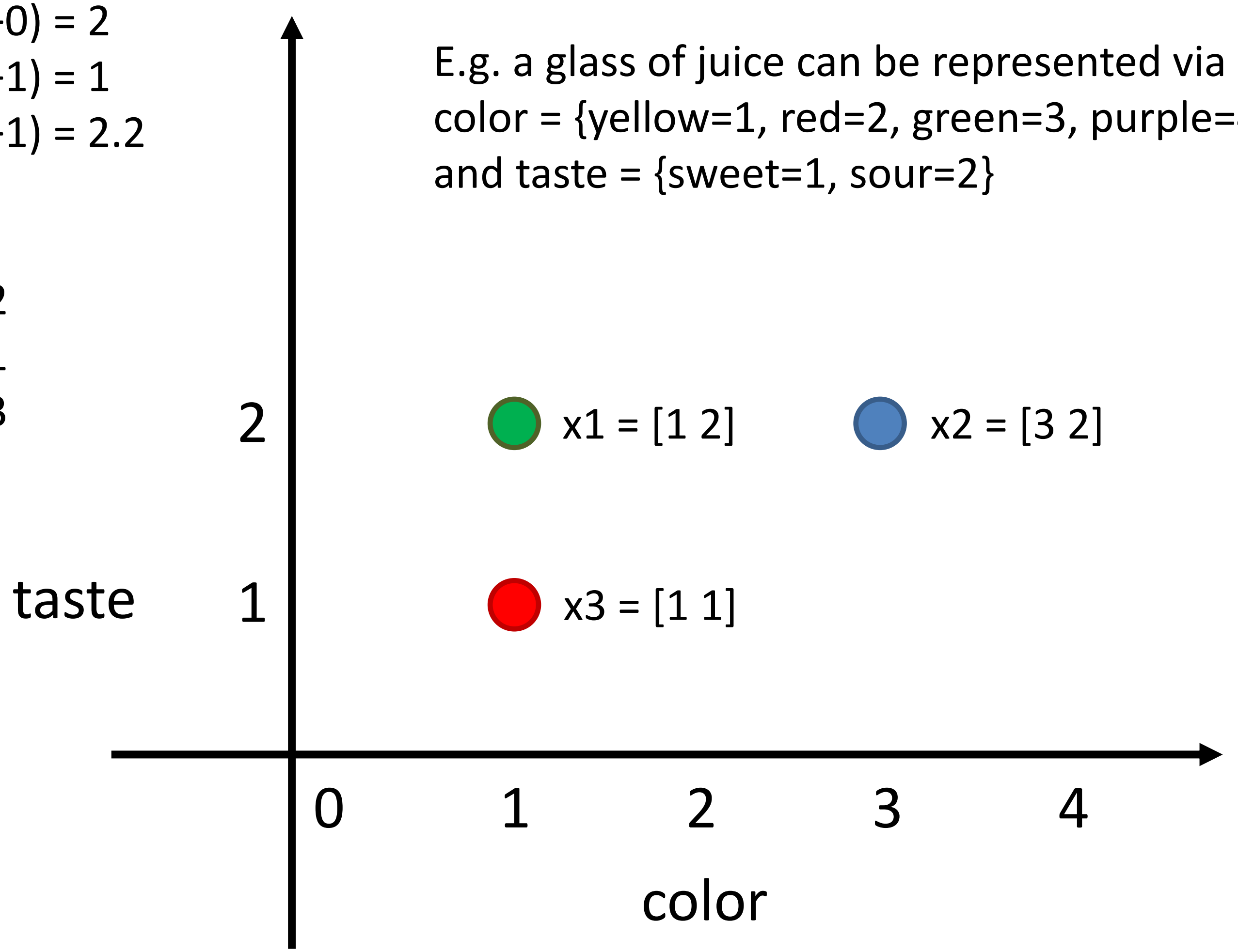
$$d(p,q) = \sqrt{(p_1-q_1)^2 + (p_2-q_2)^2 + \cdots + (p_i-q_i)^2 + \cdots + (p_n-q_n)^2}.$$

Example: Feature representation

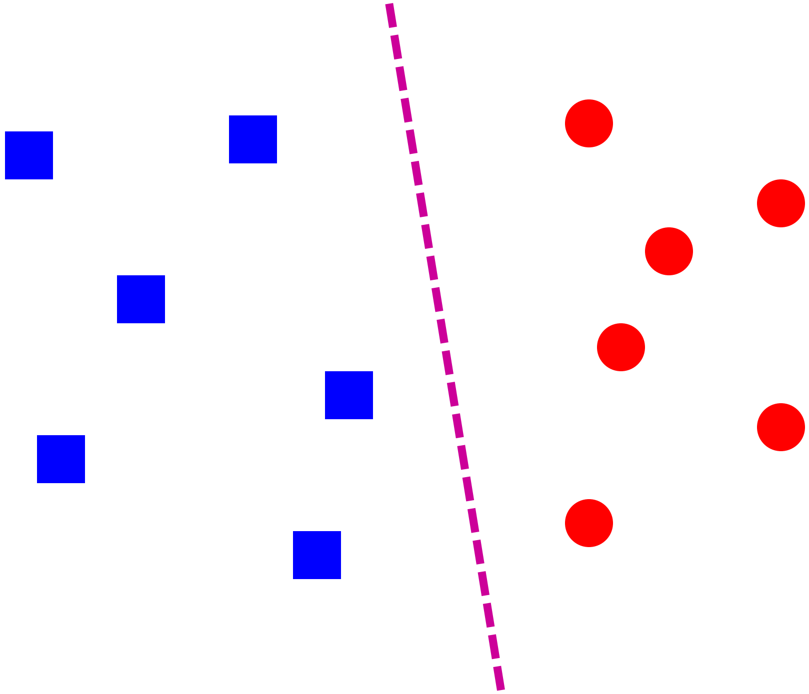
L2 distance:
d(x1, x2) = sqrt(4+0) = 2
d(x1, x3) = sqrt(0+1) = 1
d(x2, x3) = sqrt(4+1) = 2.2

L1 distance:
d(x1, x2) = 2+0 = 2
d(x1, x3) = 0+1 = 1
d(x2, x3) = 2+1 = 3

E.g. a glass of juice can be represented via its color = {yellow=1, red=2, green=3, purple=4} and taste = {sweet=1, sour=2}



Linear classifier



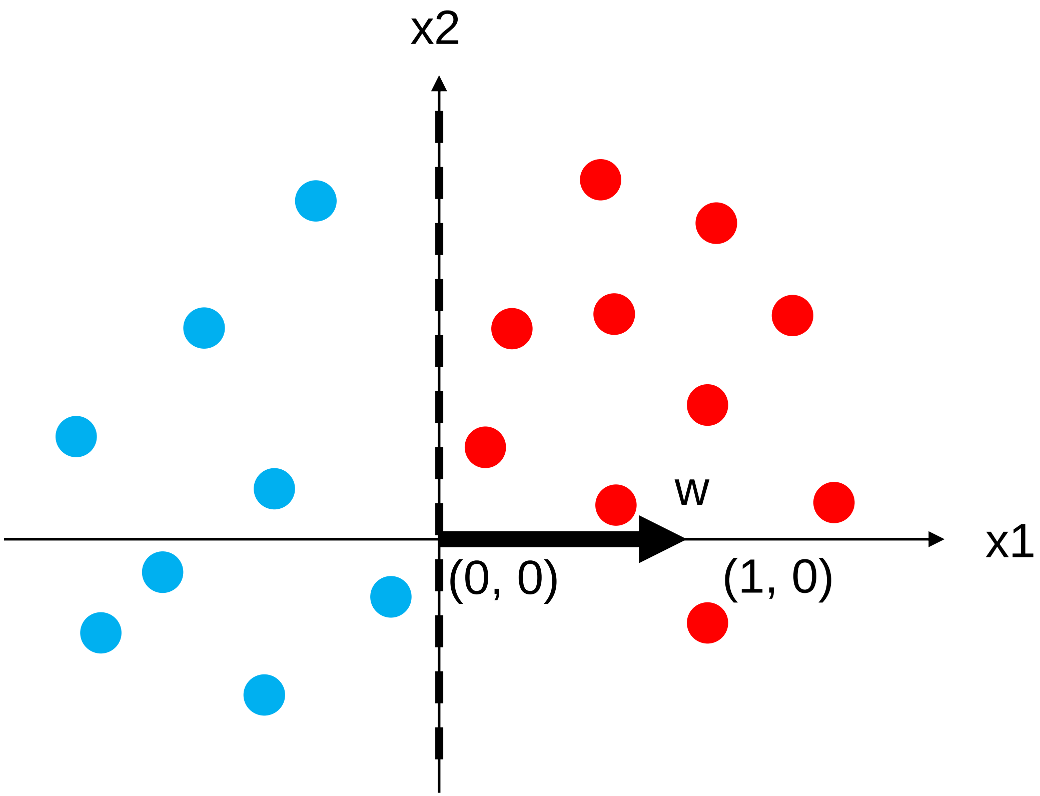
- Find a *linear function* to separate the classes

$$f(\mathbf{x}) = \text{sgn}(w_1x_1 + w_2x_2 + \ldots + w_Dx_D) = \text{sgn}(\mathbf{w}\cdot\mathbf{x})$$
$$= \text{sgn}(\mathbf{w}^T\mathbf{x})$$

Slide credit: L. Lazebnik

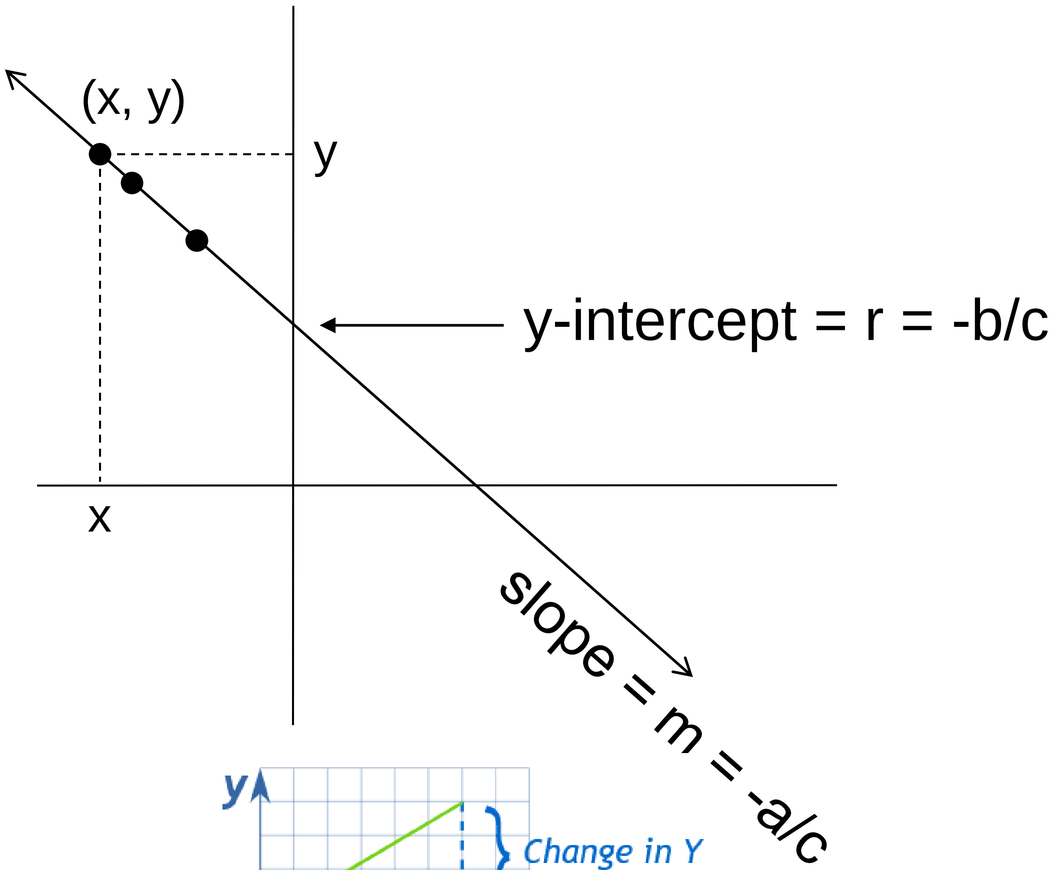
Linear classifier

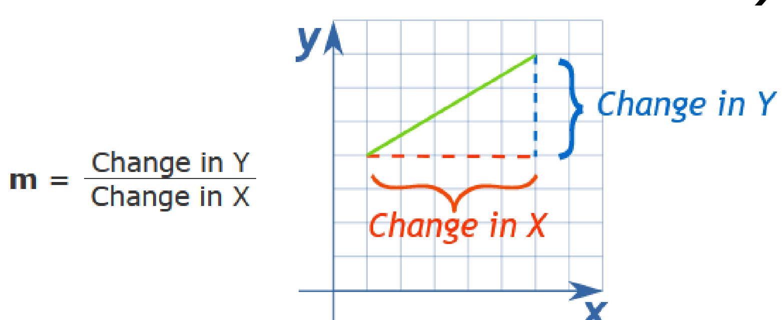
- Decision = sign($\mathbf{w}^T\mathbf{x}$) = sign(w1*x1 + w2*x2)



- What should the weights be?

Lines in R^2





https://www.mathsisfun.com/equation_of_line.html

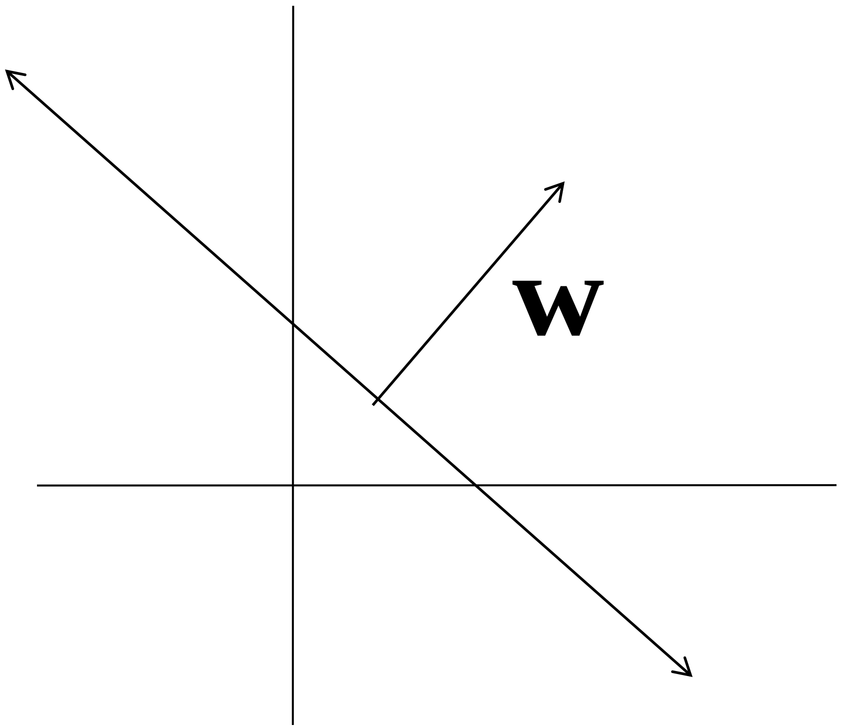
$$ax + cy + b = 0$$

mx + r = y?

ax + cy + b = 0
ax + b = -cy
(-a/c) x + (-b/c) = y

m = -a/c
r = -b/c

Lines in R^2



Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

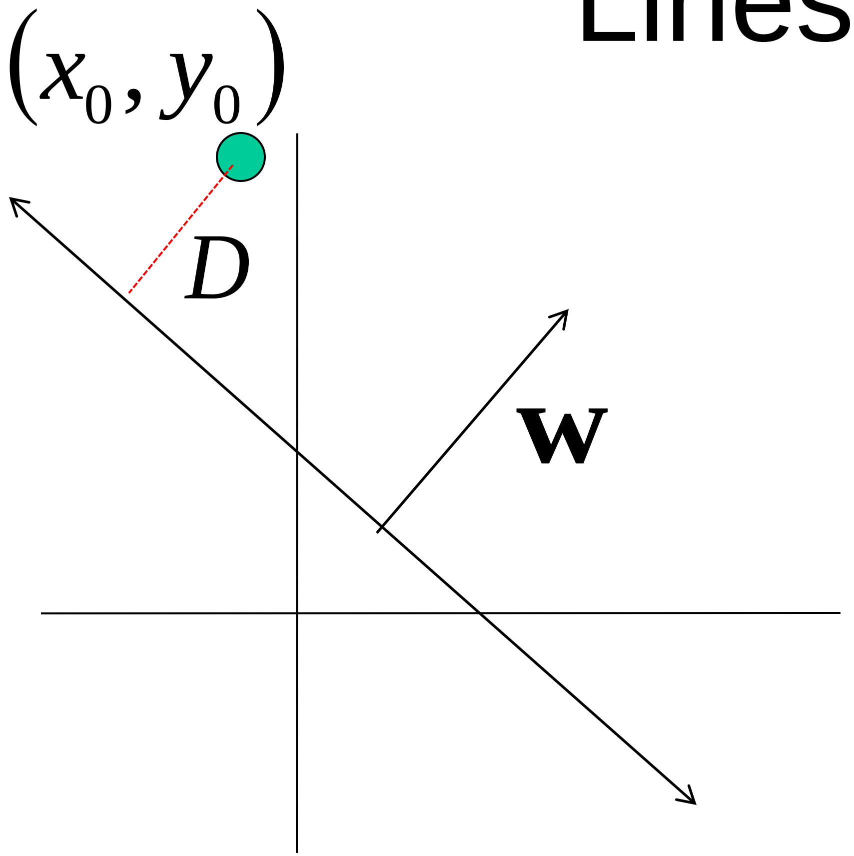
$$ax + cy + b = 0$$

$$\Updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

Kristen Grauman

Lines in R^2



Let $\mathbf{w} = \begin{bmatrix} a \\ c \end{bmatrix}$ $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$ax + cy + b = 0$$

$$\Updownarrow$$

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

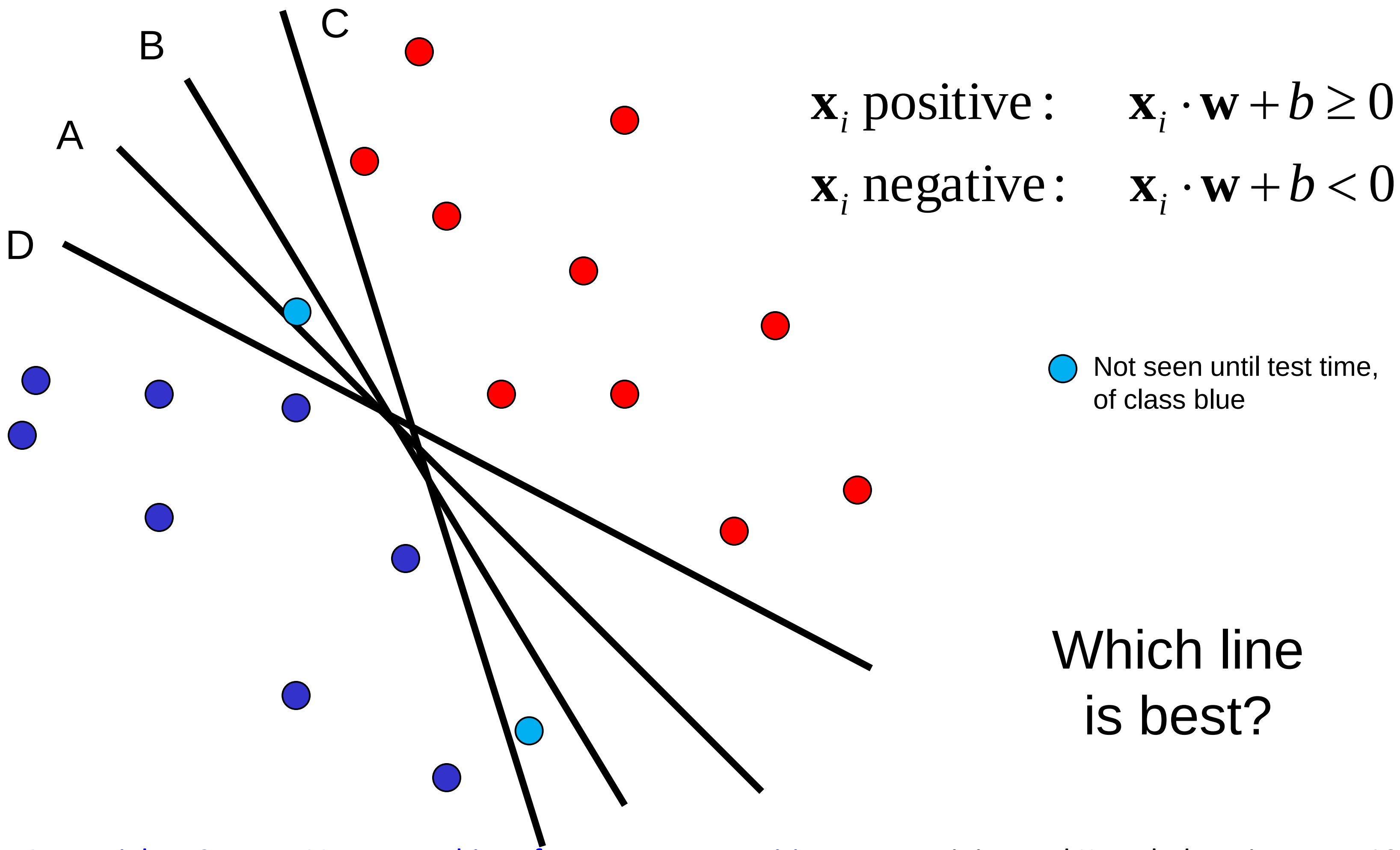
$$D = \frac{|ax_0 + cy_0 + b|}{\sqrt{a^2 + c^2}} = \frac{|\mathbf{w}^{\mathrm{T}}\mathbf{x} + b|}{\|\mathbf{w}\|}$$

distance from point to line

Kristen Grauman

Linear classifiers

- Find linear function to separate positive and negative examples



$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

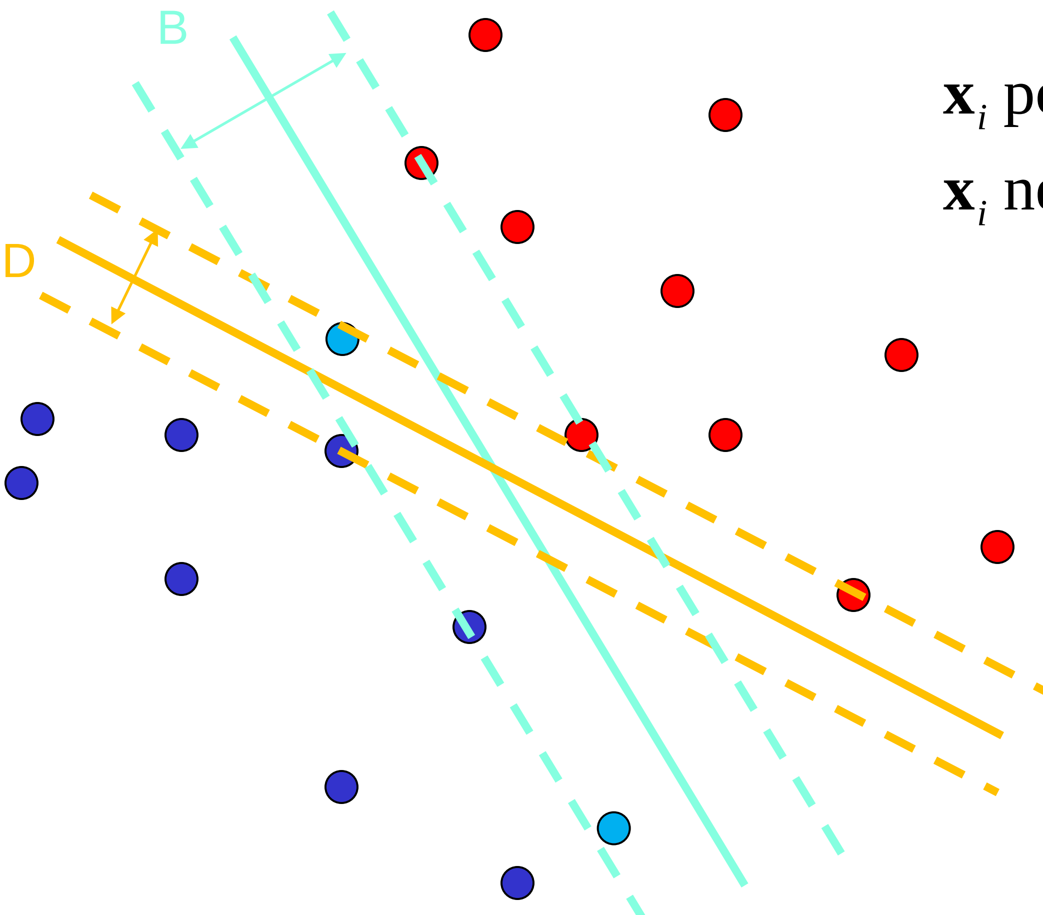
$$\mathbf{x}_i \text{ negative}: \quad \mathbf{x}_i \cdot \mathbf{w} + b < 0$$

Which line is best?

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Linear classifiers

- Find linear function to separate positive and negative examples



$$\mathbf{x}_i \text{ positive}: \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 0$$

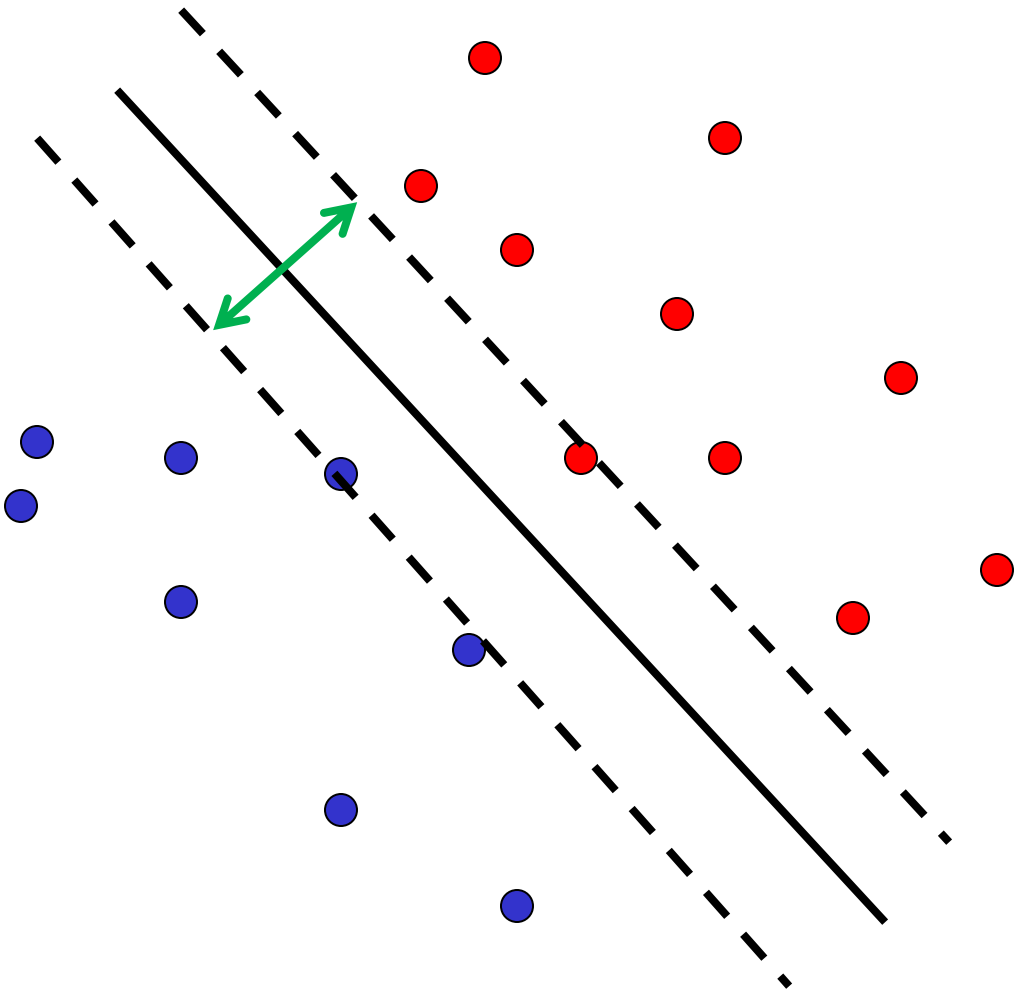
$$\mathbf{x}_i \text{ negative}: \quad \mathbf{x}_i \cdot \mathbf{w} + b < 0$$

Not seen until test time, of class blue

Which line is best?

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

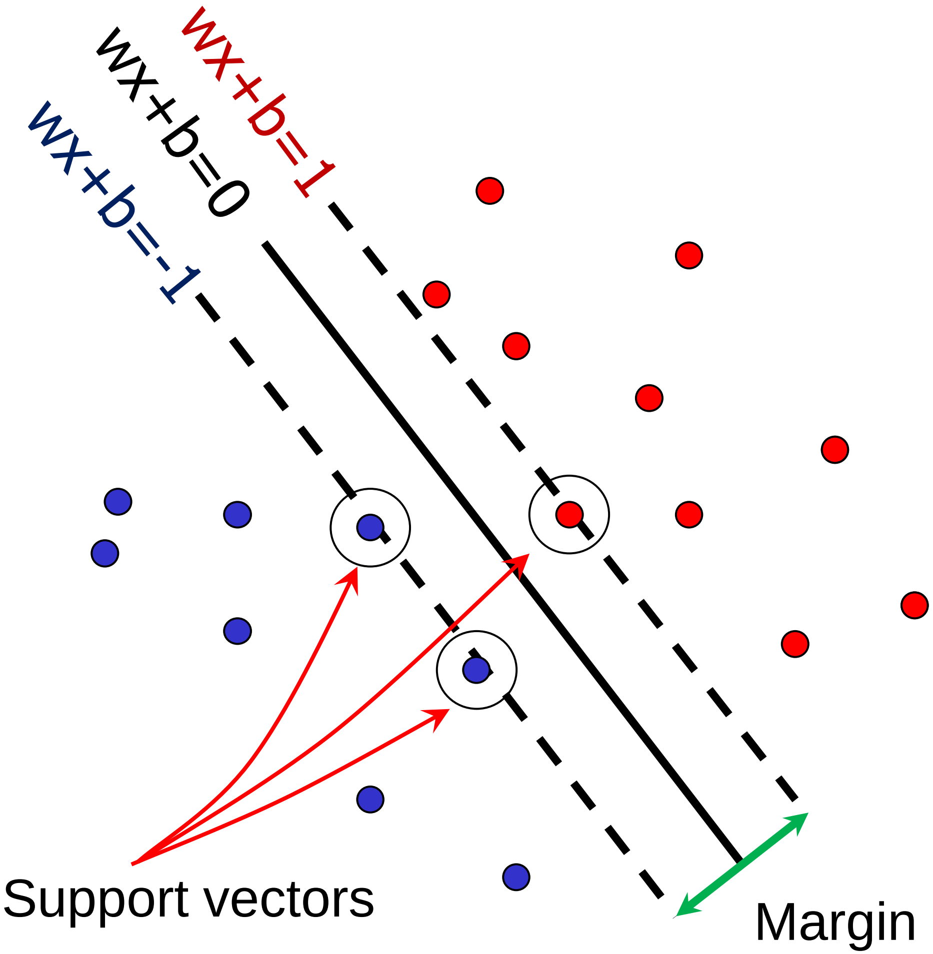


- Discriminative classifier based on *optimal separating line (for 2d case)*
- Maximize the *margin* between the positive and negative training examples

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Support vector machines

- Want line that maximizes the margin.



$\mathbf{x}_i \text{ positive } (y_i = 1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i \text{ negative } (y_i = -1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

For support, vectors, $\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$

Distance between point and line: $\frac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{||\mathbf{w}||}$

For support vectors:

$$\frac{\mathbf{w}^T\mathbf{x} + b}{\|\mathbf{w}\|} = \frac{\pm 1}{\|\mathbf{w}\|} \qquad M = \left| \frac{1}{\|\mathbf{w}\|} - \frac{-1}{\|\mathbf{w}\|} \right| = \frac{2}{\|\mathbf{w}\|}$$

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the maximum margin line

1. Maximize margin $2/||\mathbf{w}||$
2. Correctly classify all training data points:

$$\mathbf{x}_i \text{ positive } (y_i = 1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

$$\mathbf{x}_i \text{ negative } (y_i = -1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

Quadratic optimization problem:

Minimize $\frac{1}{2}\mathbf{w}^T\mathbf{w}$

Subject to $y_i(\mathbf{w}\cdot\mathbf{x}_i+b) \geq 1$

One constraint for each training point.

Note sign trick.

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Finding the maximum margin line

- Solution: $\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$

  $b = y_i - \mathbf{w} \cdot \mathbf{x}_i$ (for any support vector)

- Classification function:

$$f(x) = \text{sign}\,(\mathbf{w} \cdot \mathbf{x} + \text{b})$$

$$= \text{sign}\left(\sum_{\mathrm{i}} \alpha_{\mathrm{i}} y_i \mathbf{x}_i \cdot \mathbf{x} + b\right)$$

If f(x) < 0, classify as negative, otherwise classify as positive.

- Notice that it relies on an *inner product* between the test point $\boldsymbol{x}$ and the support vectors $\boldsymbol{x}_i$
- (Solving the optimization problem also involves computing the inner products $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$ between all pairs of training points)

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Inner product

- **The decision boundary for the SVM and its optimization depend on the inner product of two data points (vectors):**

$$\mathbf{x}_i^T \mathbf{x}_j$$

$$f(x) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + \text{b})$$
$$= \text{sign}\left(\sum_{\text{i}} \alpha_{\text{i}} y_i \mathbf{x}_i \cdot \mathbf{x} + b\right)$$

- **The inner product is equal**

$$(\mathbf{x}_i^T \mathbf{x}) = \|\mathbf{x}_i\| * \|\mathbf{x}_i\| \cos\theta$$

If the angle in between them is 0 then: $(\mathbf{x}_i^T \mathbf{x}) = \|\mathbf{x}_i\| * \|\mathbf{x}_i\|$

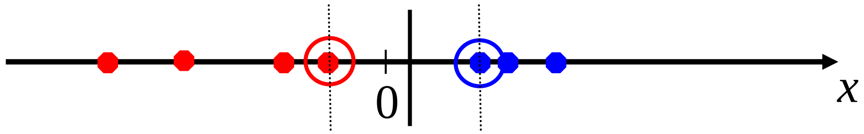
If the angle between them is 90 then: $(\mathbf{x}_i^T \mathbf{x}) = 0$

The inner product measures how similar the two vectors are

Adapted from Milos Hauskrecht

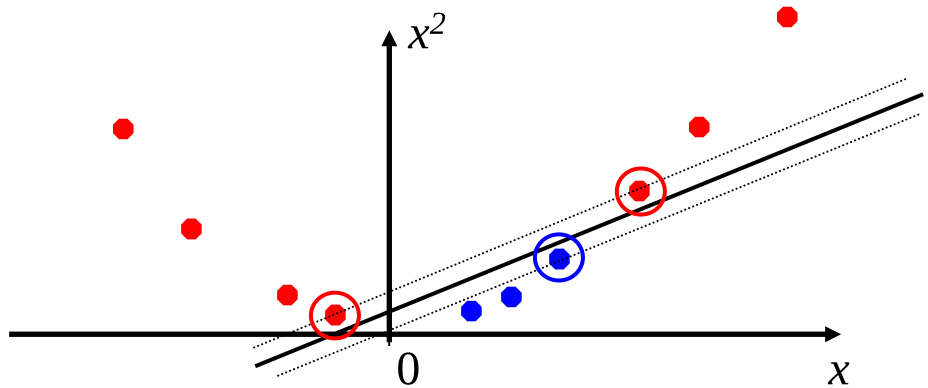
Nonlinear SVMs

- Datasets that are linearly separable work out great:



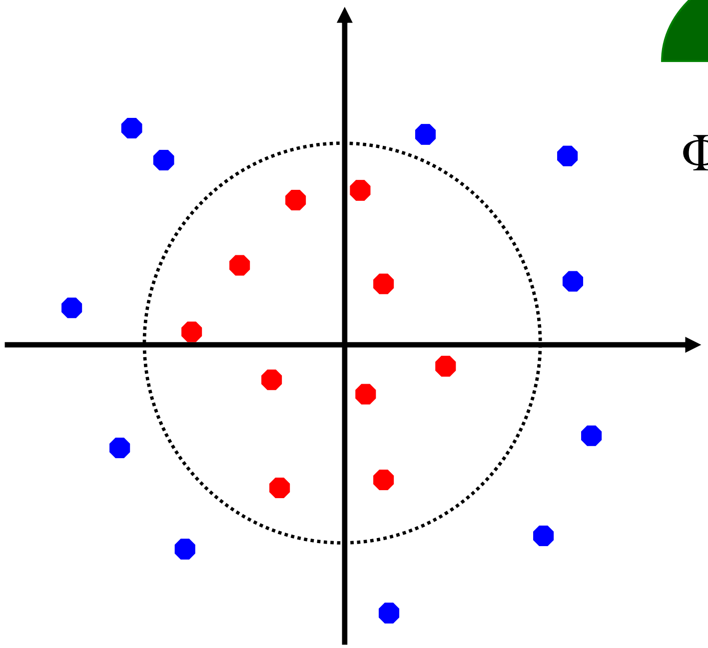
- But what if the dataset is just too hard?

- We can map it to a higher-dimensional space:

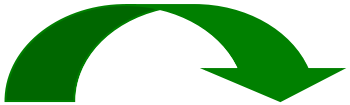


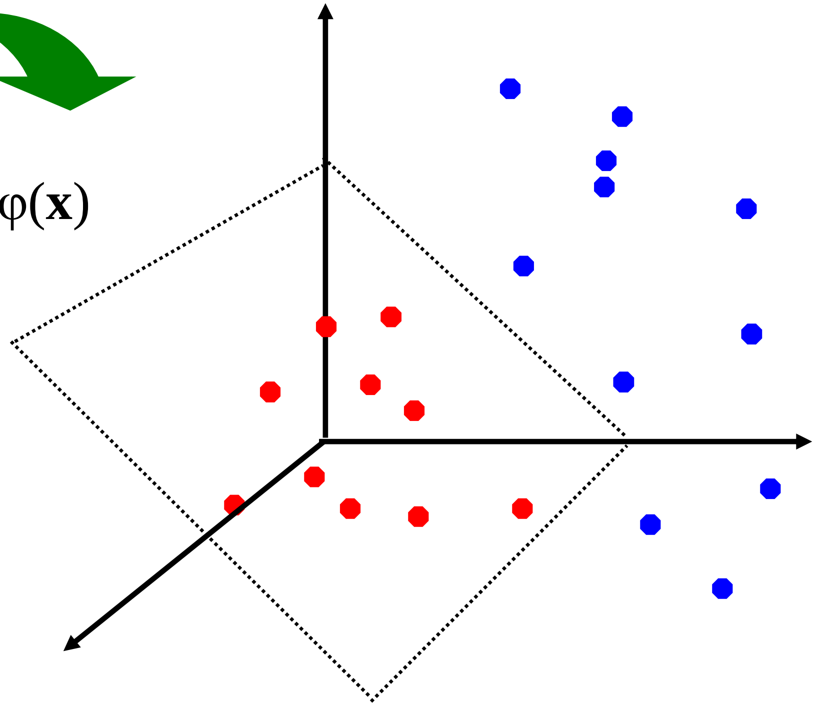
Nonlinear SVMs

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



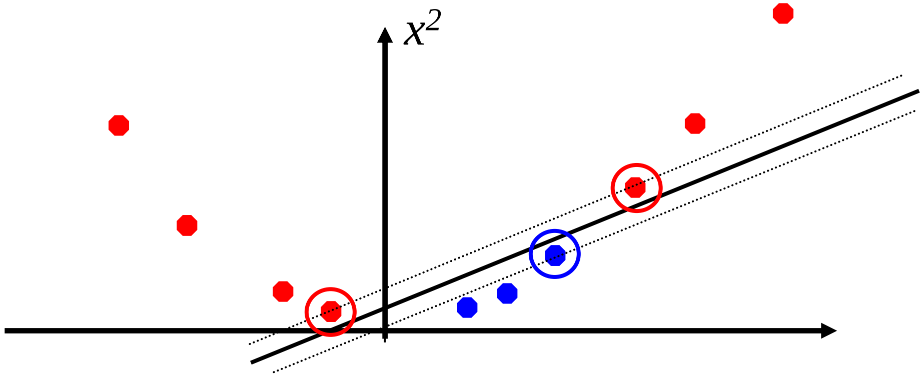
Φ: $\mathbf{x} \rightarrow \varphi(\mathbf{x})$





Nonlinear kernel: Example

- Consider the mapping $\varphi(x) = (x, x^2)$



$$\varphi(x) \cdot \varphi(y) = (x, x^2) \cdot (y, y^2) = xy + x^2 y^2$$

$$K(x, y) = xy + x^2 y^2$$

Svetlana Lazebnik

The “Kernel Trick”

- The linear classifier relies on dot product between vectors $K(\mathbf{x}_i,\mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j$
- If every data point is mapped into high-dimensional space via some transformation *Φ*: $\mathbf{x}_i \rightarrow \varphi(\mathbf{x}_i)$, the dot product becomes: $K(\mathbf{x}_i,\mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$
- A *kernel function* is similarity function that corresponds to an inner product in some expanded feature space
- *The kernel trick*: instead of explicitly computing the lifting transformation $\varphi(\mathbf{x})$, define a kernel function K such that: $K(\mathbf{x}_i,\mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)$

Examples of kernel functions

- Linear: $K(x_i, x_j) = x_i^T x_j$
- Polynomials of degree up to *d*:

$$K(x_i, x_j) = (x_i^T x_j + 1)^d$$

- Gaussian RBF:

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

- Histogram intersection:

$$K(x_i, x_j) = \sum_k \min(x_i(k), x_j(k))$$

The benefit of the “kernel trick”

- Example: Polynomial kernel for 2-dim features

$$
\begin{aligned}
k(\mathbf{x}, \mathbf{z}) &= \left(1 + \mathbf{x}^{\mathrm{T}}\mathbf{z}\right)^2 = (1 + x_1 z_1 + x_2 z_2)^2 \\
&= 1 + 2x_1 z_1 + 2x_2 z_2 + x_1^2 z_1^2 + 2x_1 z_1 x_2 z_2 + x_2^2 z_2^2 \\
&= (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, \sqrt{2}x_1 x_2, x_2^2)(1, \sqrt{2}z_1, \sqrt{2}z_2, z_1^2, \sqrt{2}z_1 z_2, z_2^2)^{\mathrm{T}} \\
&= \phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{z}).
\end{aligned}
\tag{7.42}
$$

- … lives in 6 dimensions
- With the kernel trick, we directly compute an inner product in 2-dim space, obtaining a scalar that we add 1 to and exponentiate

Hard-margin SVMs

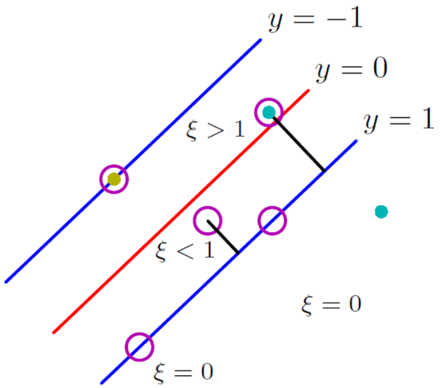
$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\|\boldsymbol{w}\|^2$$

The *w* that minimizes…

Maximize margin

$$\text{subject to} \quad y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1 \quad , \quad \forall i = 1, \ldots, N$$

Soft-margin SVMs



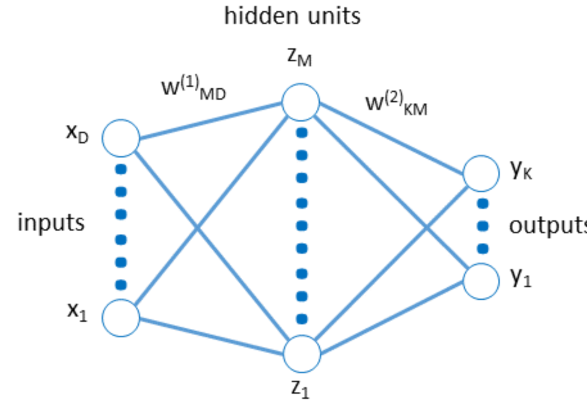
$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

- $\min_{\boldsymbol{w}}$: The *w* that minimizes…
- C: Misclassification cost
- N: # data samples
- ξ_i: Slack variable
- $\frac{1}{2}\|\boldsymbol{w}\|^2$: Maximize margin
- $C\sum_{i=1}^{N}\xi_i$: Minimize misclassification

$$\text{subject to} \quad y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1 - \xi_i,$$
$$\xi_i \geq 0, \quad \forall i = 1, \ldots, N$$

Figure from Chris Bishop

Deep Learning in a Nutshell



- Input → network → outputs
- Input X is raw (e.g. raw image, one-hot representation of text)
- Network extracts features: abstraction of input (e.g. not pixels, but edges)
- Output is the labels Y
- All parameters of the network *learned* (during *training*) by checking how well predicted/true Y agree, using labels in the training set

Elements of Machine *Learning*

- Every machine learning algorithm has:
  – Data representation (x, y)
  – Problem representation (network)
  – Evaluation / objective function
  – Optimization (solve for parameters of network)

Adapted from Pedro Domingos

Data representation

- Let's brainstorm what our "X" should be for various "Y" prediction tasks…

- Weather prediction?
- Movie ratings predictions?

Problem representation

- Instances
- Decision trees
- Sets of rules / Logic programs
- Support vector machines
- Graphical models (Bayes/Markov nets)
- **Neural networks**
- Model ensembles
- Etc.

Slide credit: Pedro Domingos

Evaluation / objective function

- Accuracy
- Precision and recall
- Squared error
- Likelihood
- Posterior probability
- Cost / Utility
- Margin
- Entropy
- K-L divergence
- Etc.

Slide credit: Pedro Domingos

Loss functions

- Measure error
- Can be defined for discrete or continuous outputs
- E.g. if task is classification – could use cross-entropy loss
- If task is regression – use L2 loss i.e. $||y-y'||$

Optimization

- Optimization means we need to solve for the parameters **w** of the model
- For a (non-linear) neural network, there is no closed-form solution to solve for **w**; cannot set up linear system with **w** as the unknowns
- Thus, optimization solutions look like this:
  1. Initialize **w** (e.g. randomly)
  2. Check error (ground-truth vs predicted labels on training set) under current model
  3. Use gradient (derivative) of error wrt **w** to update **w**
  4. Repeat from 2 until convergence

Types of Learning

- Supervised learning
  - Training data includes desired outputs
- Unsupervised learning
  - Training data does not include desired outputs
- Weakly or Semi-supervised learning
  - Training data includes a few desired outputs, or contains labels that only approximate the labels desired at test time (noisy)
- Reinforcement learning
  - Rewards from sequence of actions

Adapted from: Dhruv Batra

Types of Prediction Tasks

Supervised Learning

x → Classification → y   Discrete

x → Regression → y   Continuous

Unsupervised Learning

x → Clustering → x'   Discrete ID

x → Dimensionality Reduction → x'   Continuous

Adapted from Dhruv Batra

Validation strategies

- Ultimately, for our application, what do we want?
  - High accuracy on training data?
  - No, high accuracy on *unseen/new/test data!*
  - Why is this tricky?
- Training data
  - Features (x) and labels (y) used to learn mapping f
- Test data
  - Features used to make a prediction
  - Labels only used to see how well we've learned f!!!
- Validation data
  - Held-out set of the *training data*
  - Can use both features and labels to tune model *hyperparameters*
  - *Hyperparameters* are "knobs" of the algorithm tuned by the designer: number of iterations for learning, learning rate, etc.
  - We train multiple model (one per hyperparameter setting) and choose the best one, on the validation set

Validation strategies

Idea #1: Choose hyperparameters that work best on the data

BAD: Overfitting; e.g. in K-nearest neighbors, K = 1 always works perfectly on training data

Your Dataset

Idea #2: Split data into **train** and **test**, choose hyperparameters that work best on test data

BAD: No idea how algorithm will perform on new data; cheating

train | test

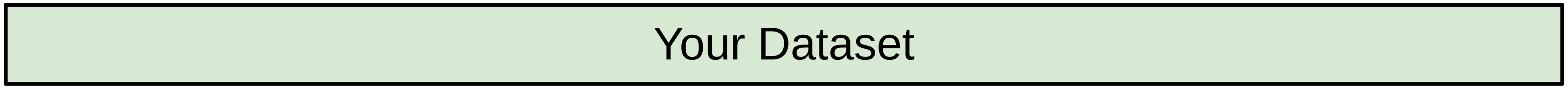
Idea #3: Split data into **train**, **val**, and **test**; choose hyperparameters on val and evaluate on test

Better!

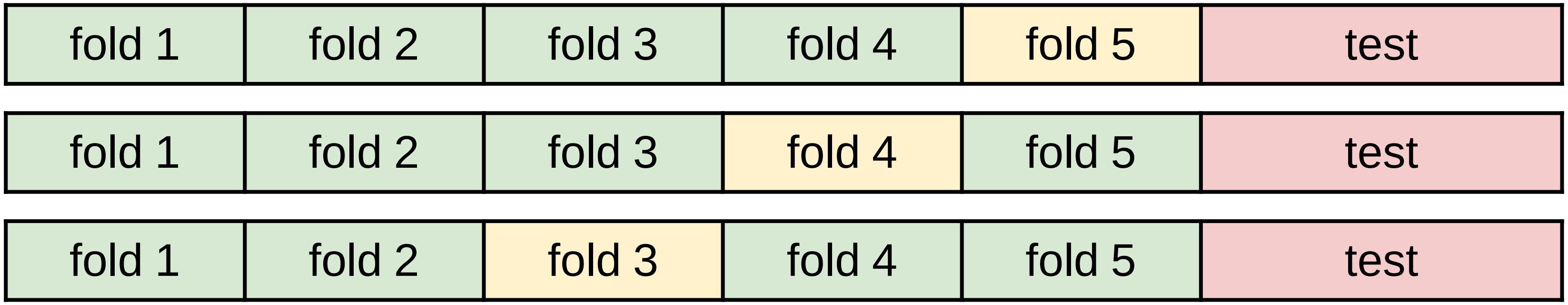
train | validation | test

Adapted from Fei-Fei, Johnson, Yeung

Validation strategies



Idea #4: **Cross-Validation**: Split data into **folds**,
try each fold as validation and average the results



Useful for small datasets, but not used too frequently in deep learning

Adapted from Fei-Fei, Johnson, Yeung

Why do we hope this would work?

- Statistical estimation view:
  - x and y are *random variables*
  - $D = (x_1,y_1), (x_2,y_2), \ldots, (x_N,y_N) \quad \sim P(X,Y)$
  - Both training & testing data sampled IID from P(X,Y)
    - IID: Independent and Identically Distributed
  - Learn on training set, have some hope of *generalizing* to test set

Adapted from Dhruv Batra

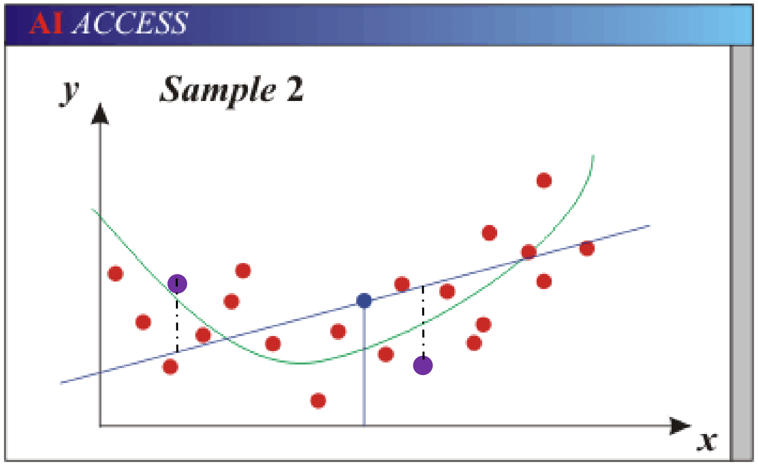
Generalization

Training set (labels known)

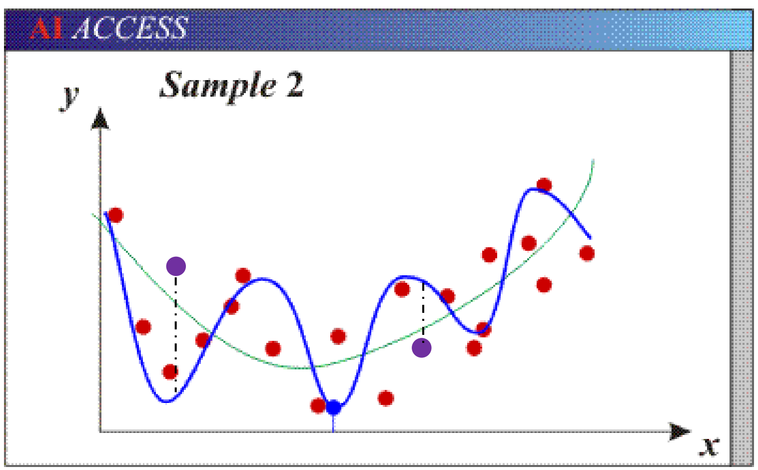
Test set (labels unknown)

- How well does a learned model generalize from the data it was trained on to a new test set?

Generalization



- Underfitting: Models with too few parameters are inaccurate because of a large bias (not enough flexibility).



- Overfitting: Models with too many parameters are inaccurate because of a large variance (too much sensitivity to the sample).

Purple dots = possible test points

Red dots = training data (all that we see before we ship off our model!)

Green curve = true underlying model

Blue curve = our predicted model/fit

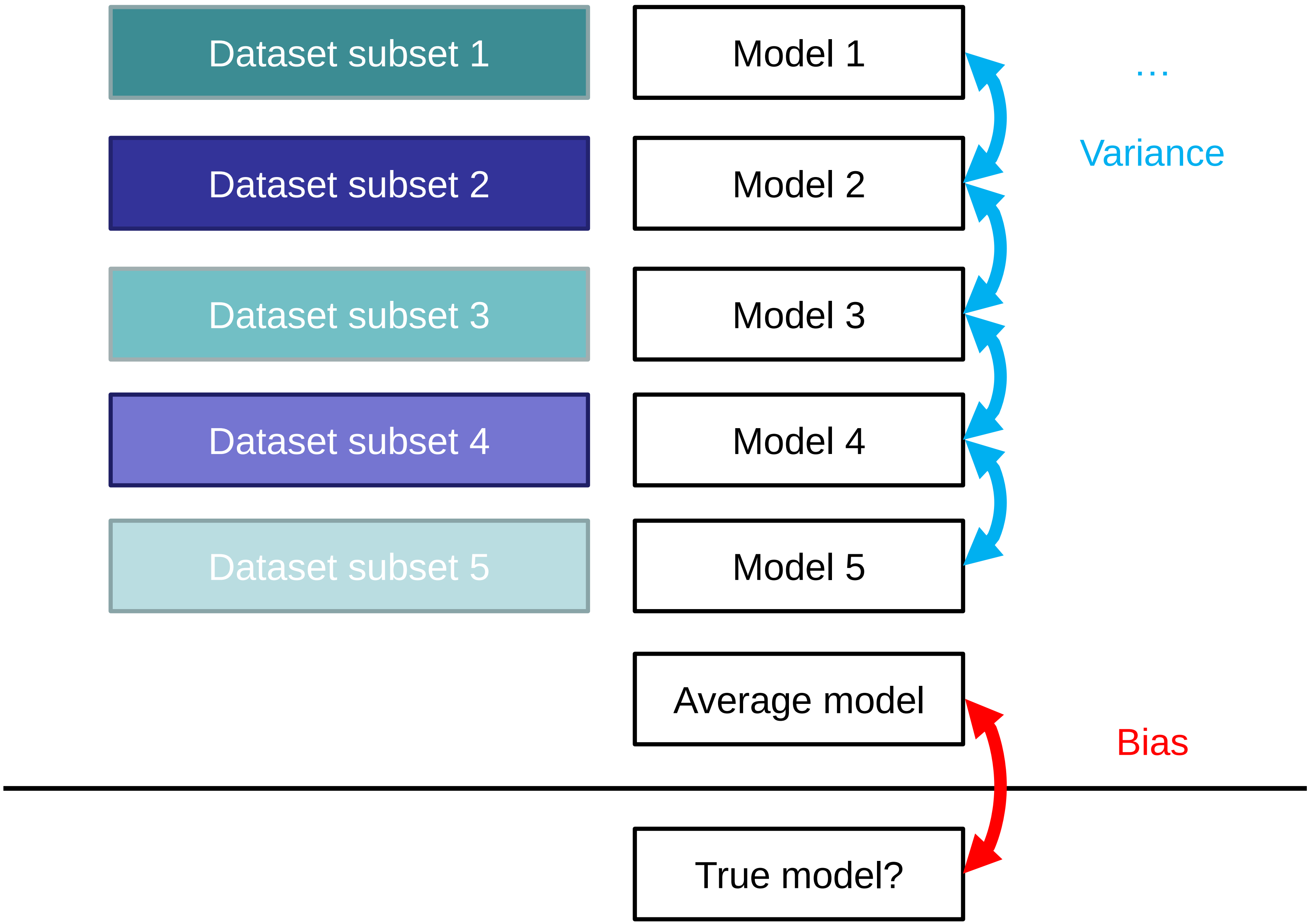
Adapted from D. Hoiem

Generalization

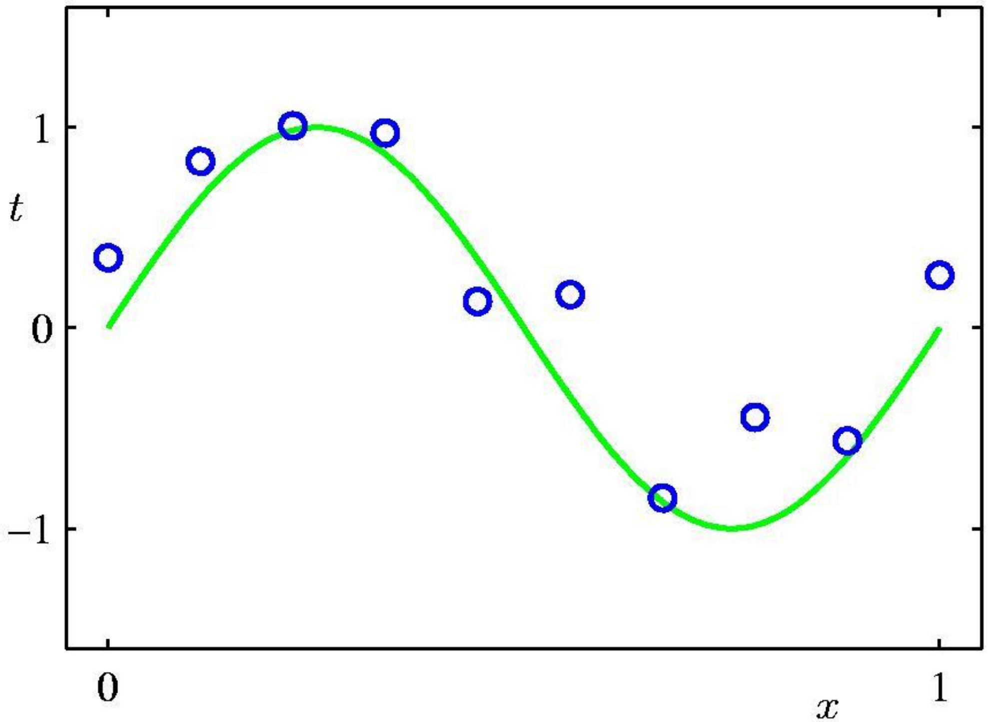
- Components of generalization error
  - **Noise** in our observations: unavoidable
  - **Bias:** due to inaccurate assumptions/simplifications by model
  - **Variance:** models estimated from different training sets differ greatly rom each other
- **Underfitting:** model is too “simple” to represent all the relevant class characteristics
  - High bias and low variance
  - High training error and high test error
- **Overfitting:** model is too “complex” and fits irrelevant characteristics (noise) in the data
  - Low bias and high variance
  - Low training error and high test error

Slide credit: L. Lazebnik

Generalization



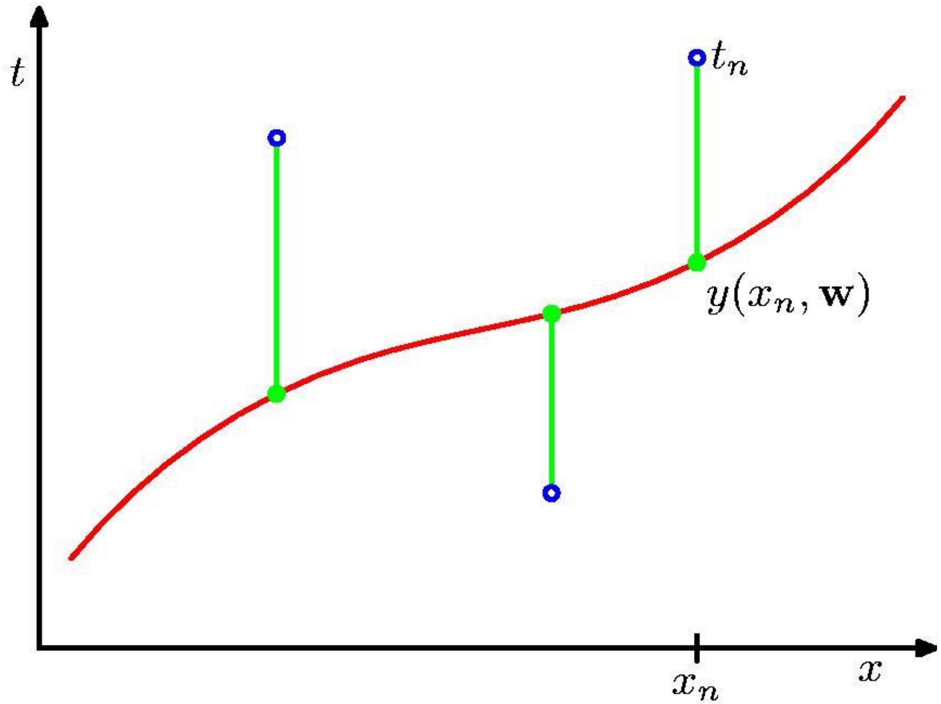
Polynomial Curve Fitting



$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

Slide credit: Chris Bishop

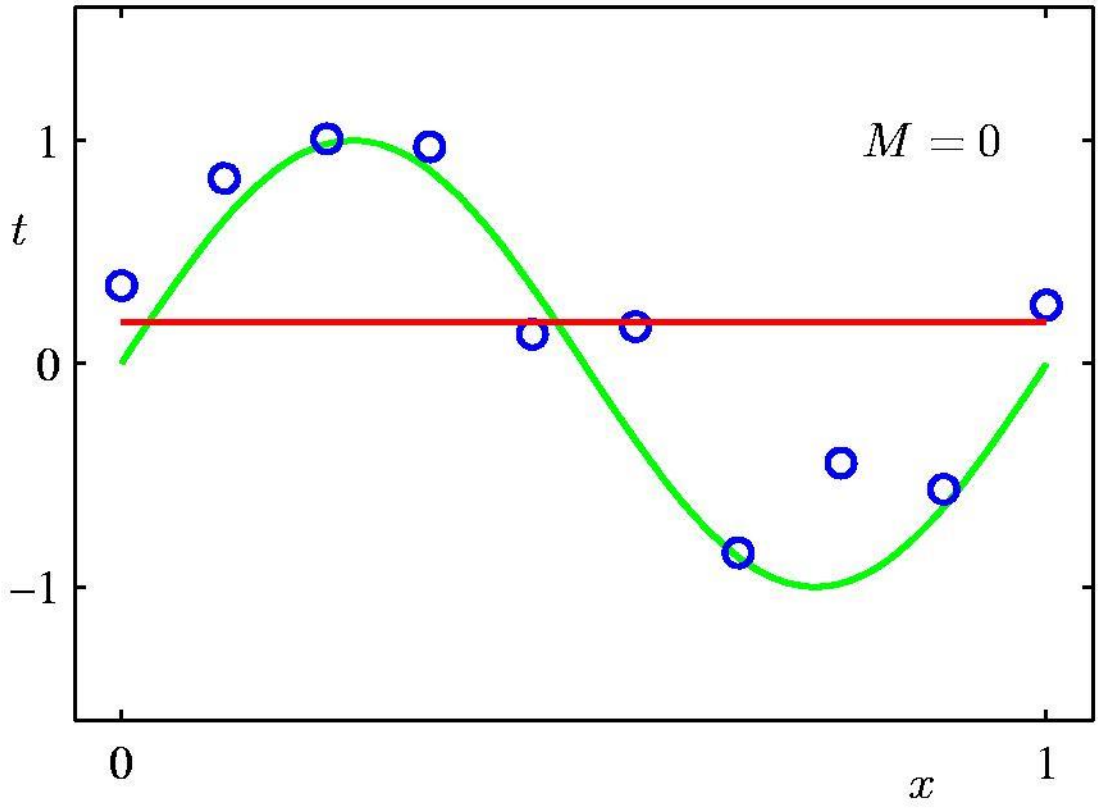
Sum-of-Squares Error Function



$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

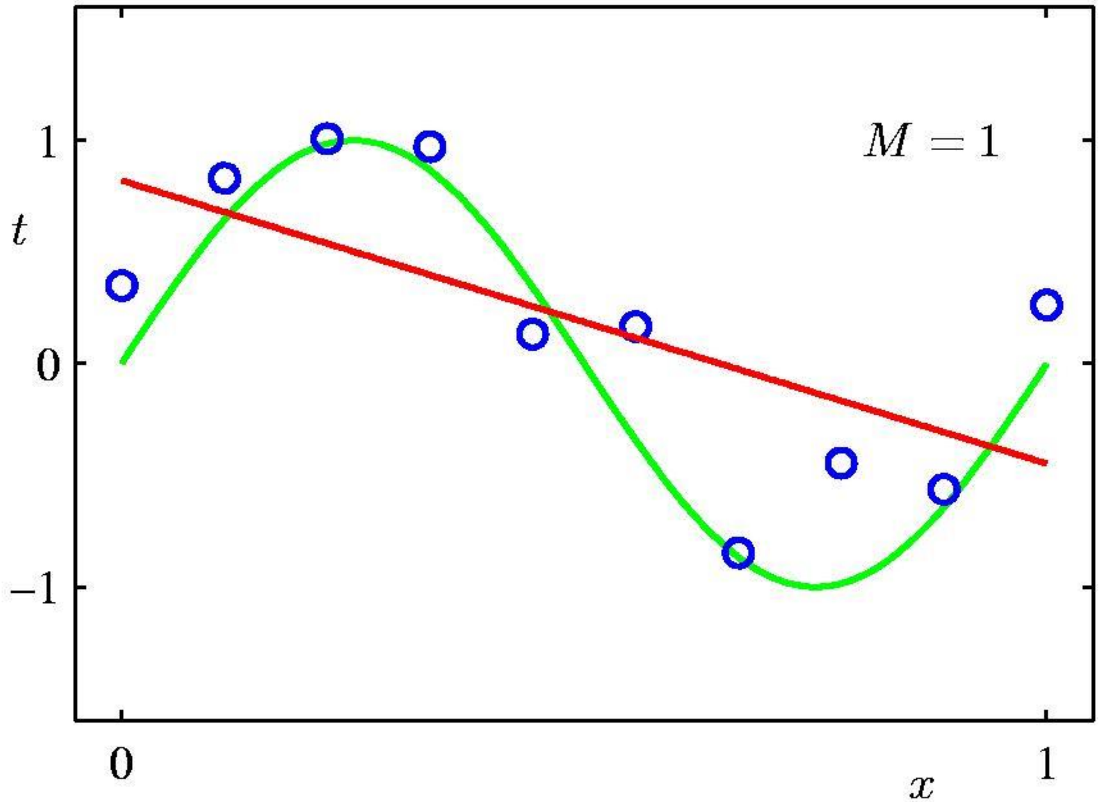
Slide credit: Chris Bishop

0^{th} Order Polynomial



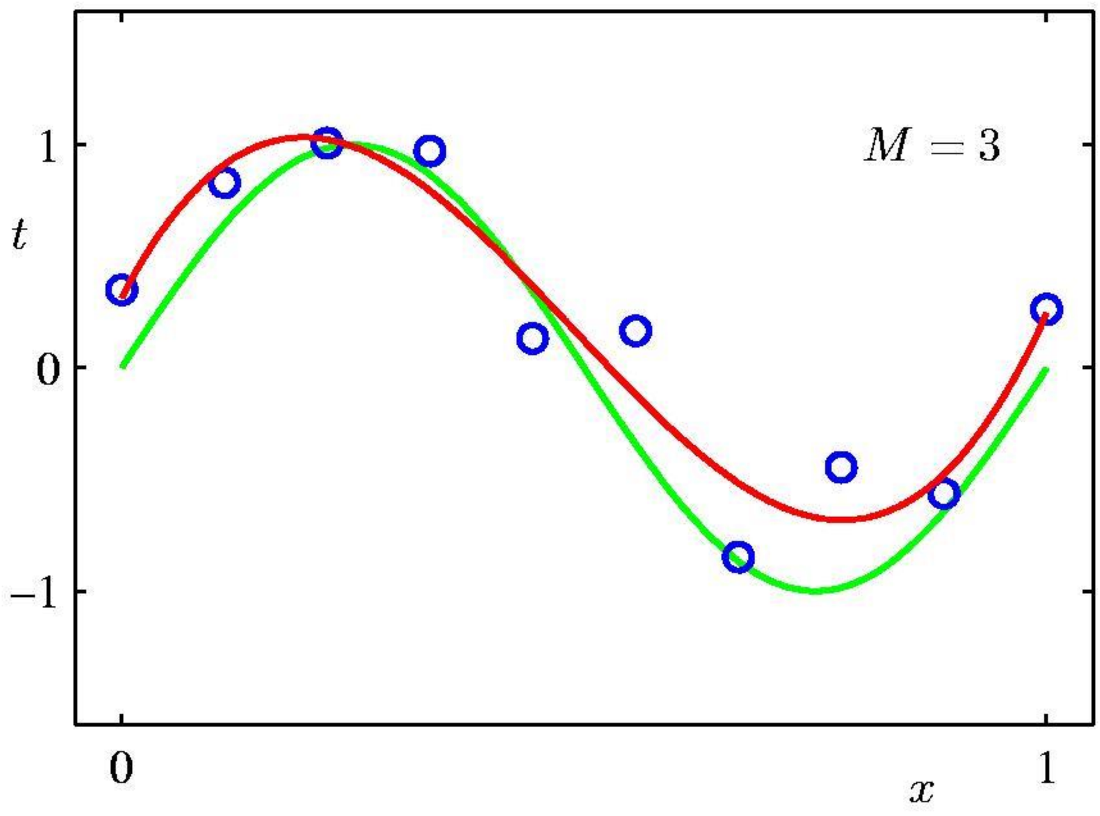
Slide credit: Chris Bishop

1st Order Polynomial



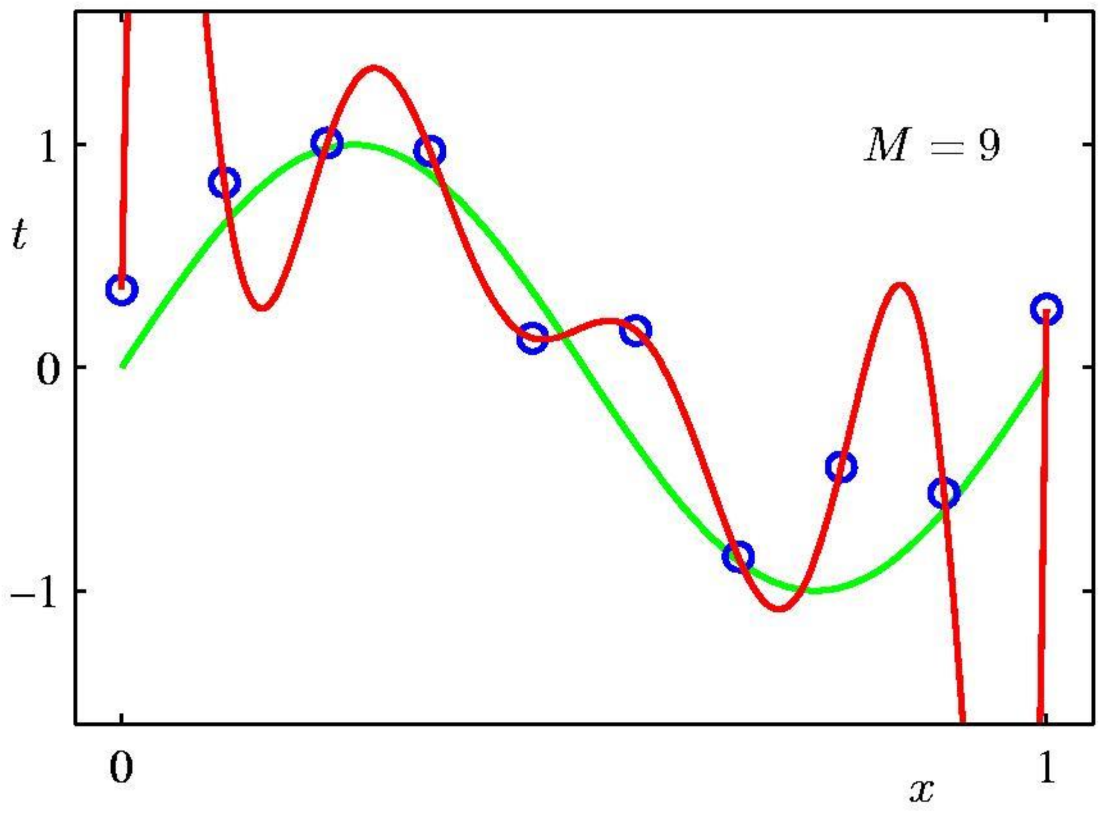
Slide credit: Chris Bishop

3rd Order Polynomial



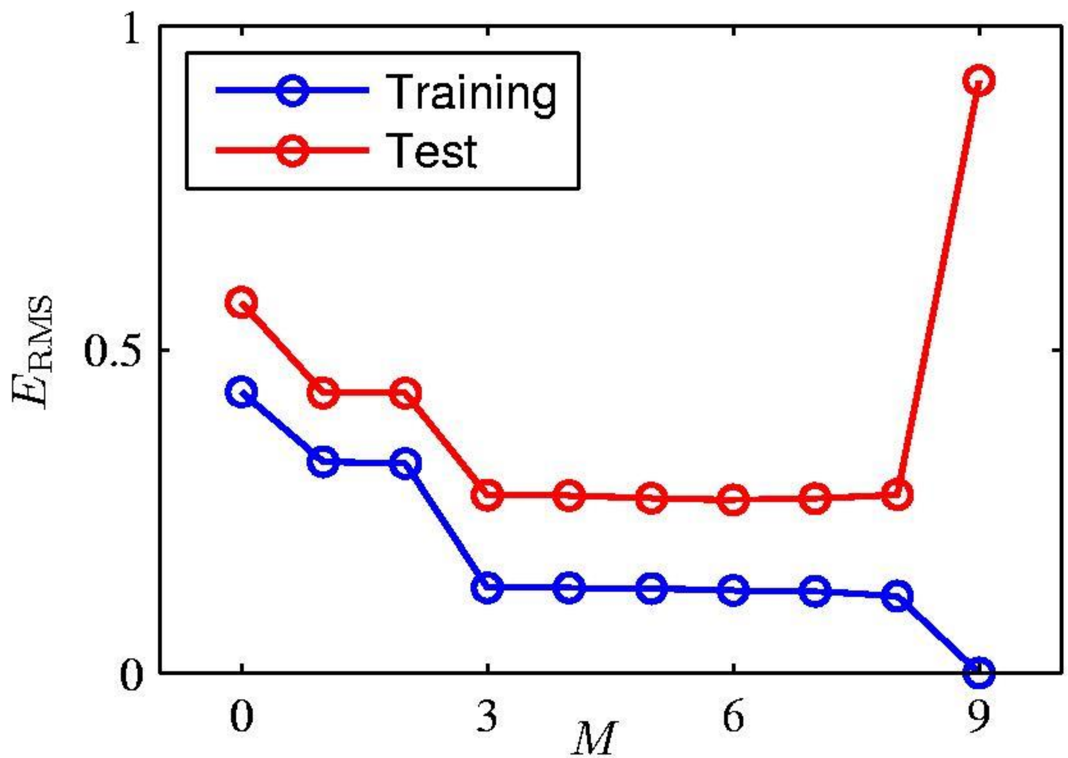
Slide credit: Chris Bishop

9th Order Polynomial



Slide credit: Chris Bishop

Over-fitting

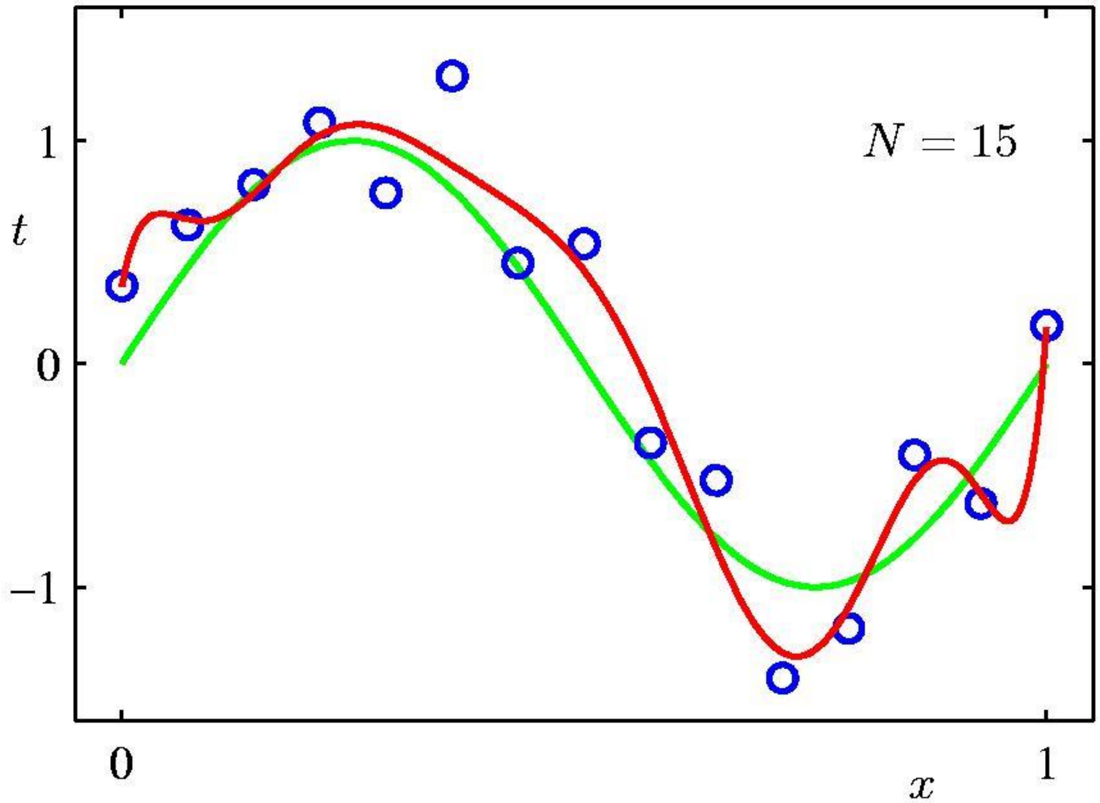


Root-Mean-Square (RMS) Error: $E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^{\star})/N}$

Slide credit: Chris Bishop

Data Set Size: $N = 15$

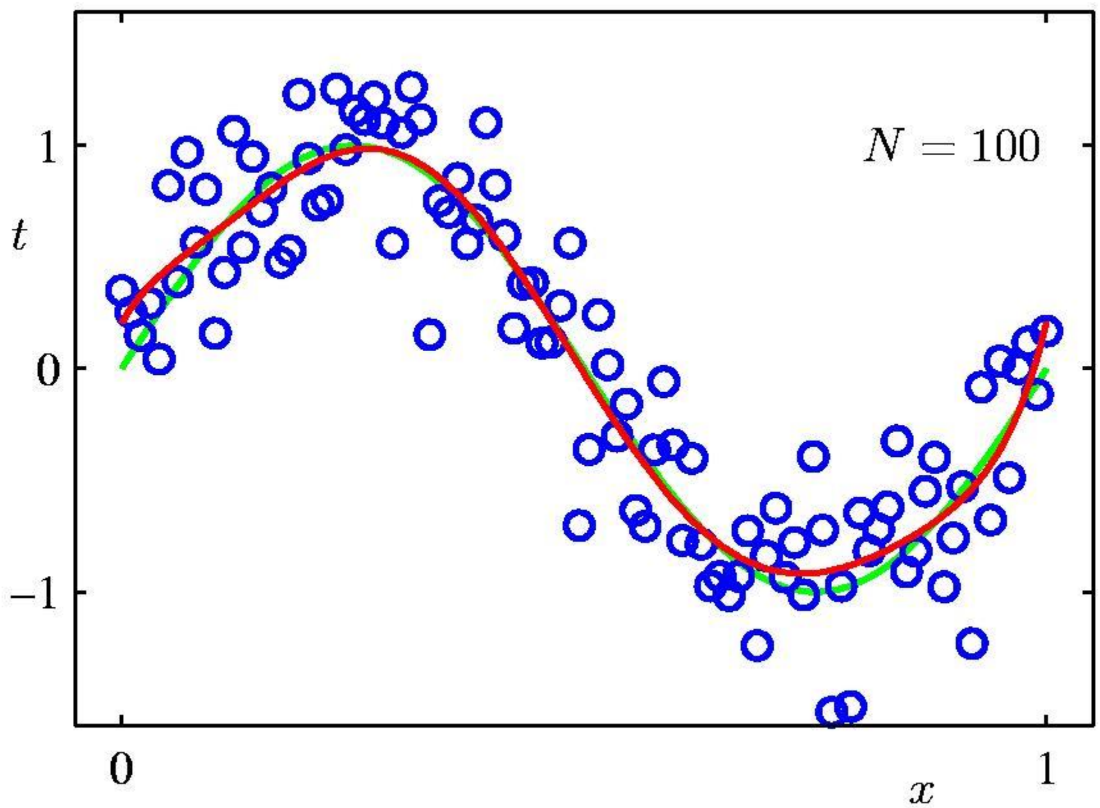
9th Order Polynomial



Slide credit: Chris Bishop

Data Set Size: $N = 100$

9th Order Polynomial



Slide credit: Chris Bishop

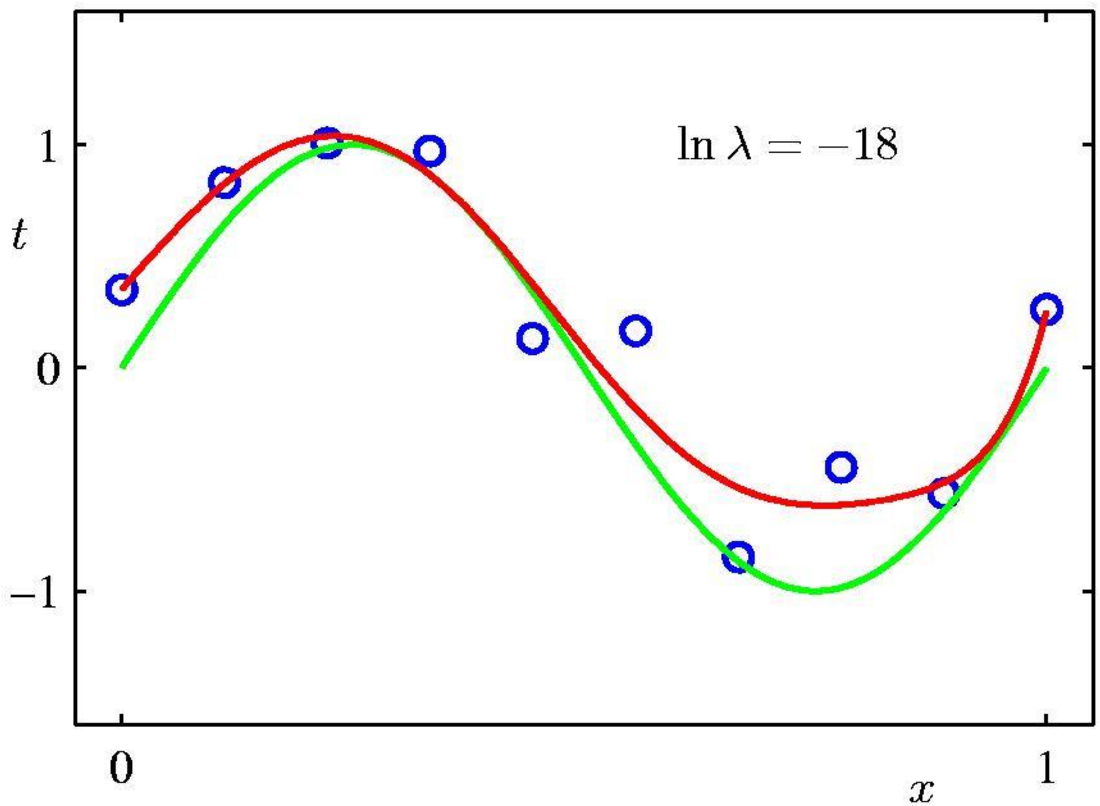
Regularization

Penalize large coefficient values

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

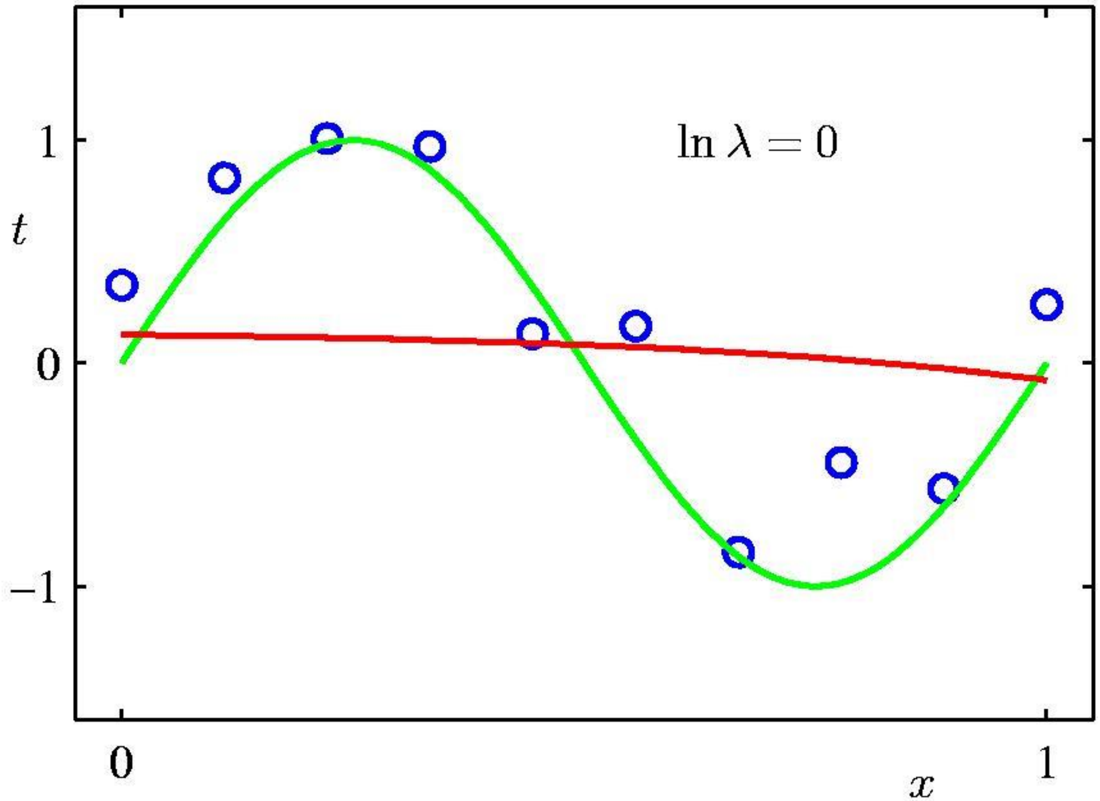
(Remember: We want to minimize this expression.)

Regularization: $\ln \lambda = -18$



Slide credit: Chris Bishop

Regularization: $\ln \lambda = 0$



Slide credit: Chris Bishop

Polynomial Coefficients

	$M=0$	$M=1$	$M=3$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

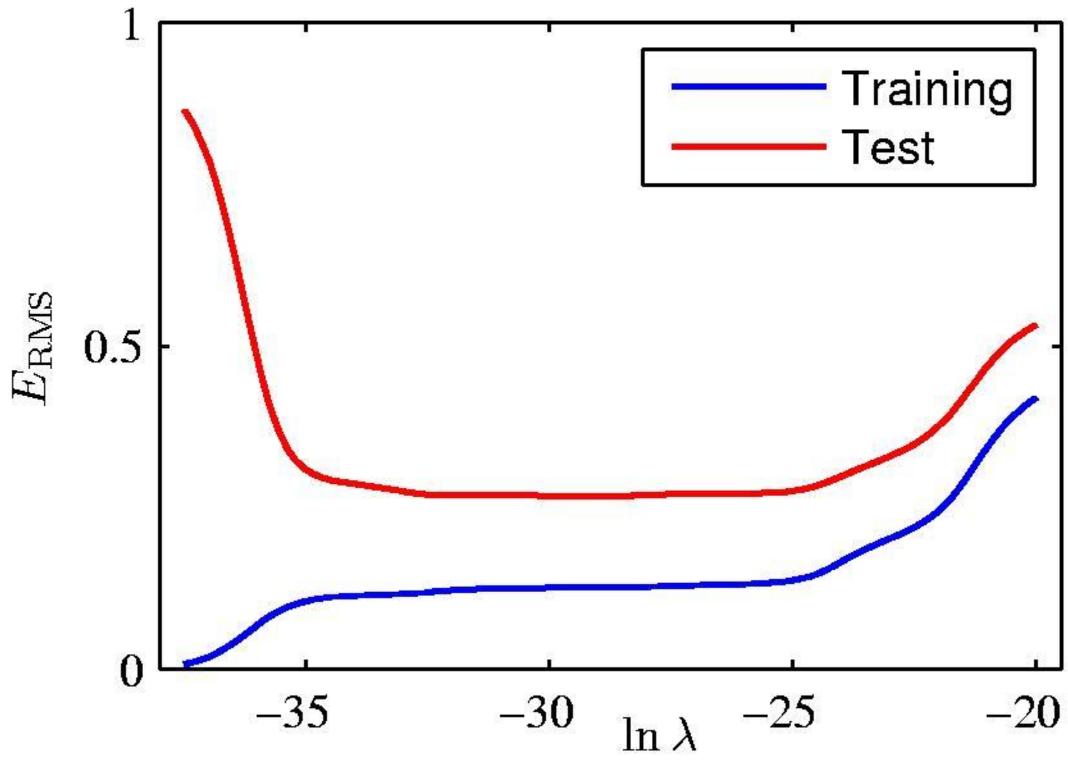
Slide credit: Chris Bishop

Polynomial Coefficients

	No regularization		Huge regularization
	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

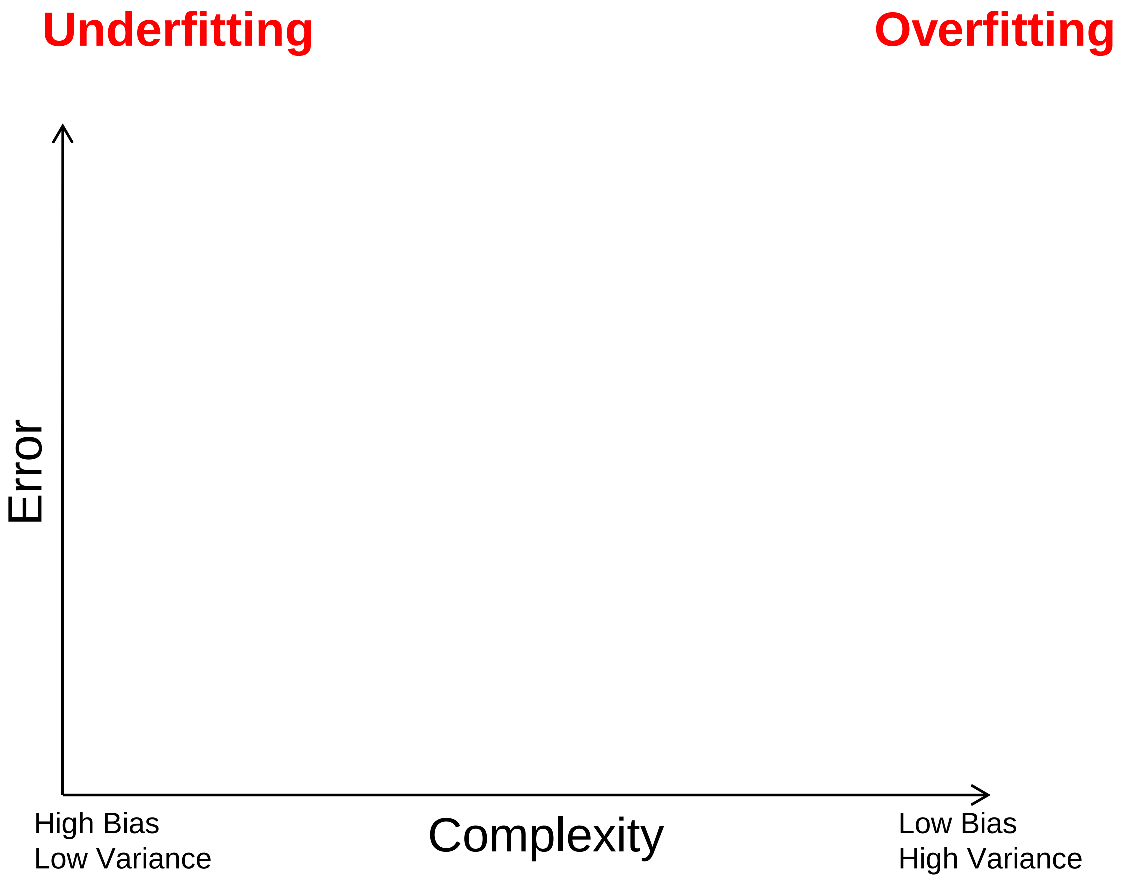
Adapted from Chris Bishop

Regularization: E_{RMS} vs. $\ln \lambda$



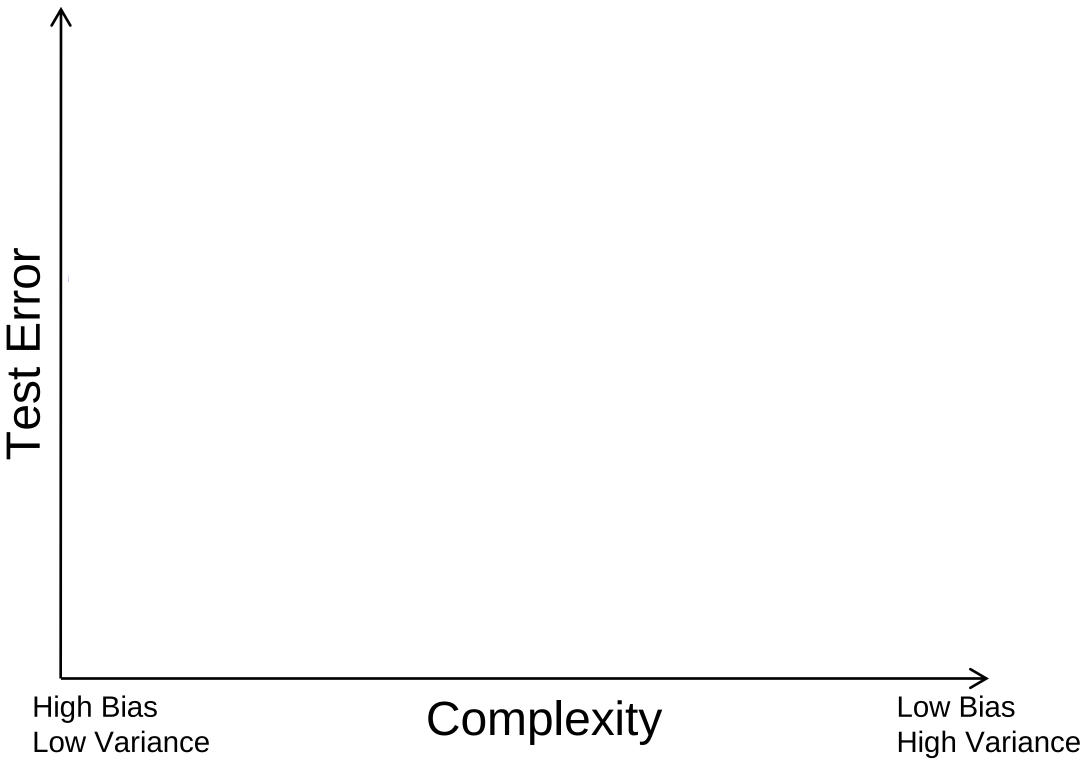
Slide credit: Chris Bishop

Training vs test error



Slide credit: D. Hoiem

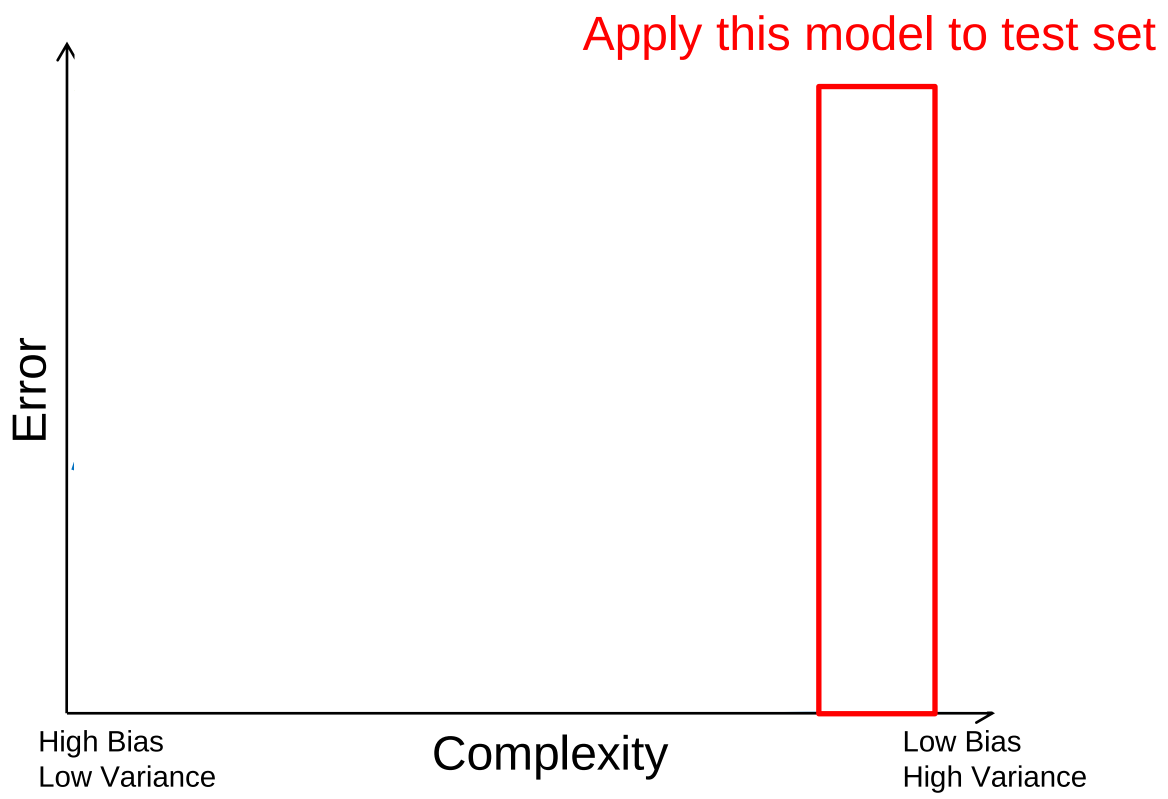
The effect of training set size



Slide credit: D. Hoiem

Choosing the trade-off between bias and variance

- Need validation set (separate from the test set)



Slide credit: D. Hoiem

Summary of generalization

- Try simple classifiers first
- Better to have smart features and simple classifiers than simple features and smart classifiers
- Use increasingly powerful classifiers with more training data
- As an additional technique for reducing variance, try regularizing the parameters

Slide credit: D. Hoiem

Linear algebra review

See http://cs229.stanford.edu/section/cs229-linalg.pdf for more

Vectors and Matrices

- Vectors and matrices are just collections of ordered numbers that represent something: movements in space, scaling factors, word counts, movie ratings, pixel brightnesses, etc.
- We'll define some common uses and standard operations on them.

Vector

- A column vector $\mathbf{v} \in \mathbb{R}^{n \times 1}$ where

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

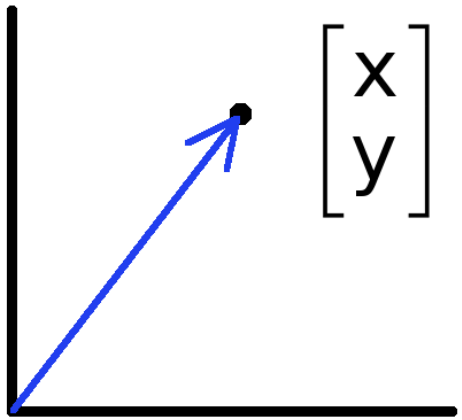
- A row vector $\mathbf{v}^T \in \mathbb{R}^{1 \times n}$ where

$$\mathbf{v}^T = \begin{bmatrix} v_1 & v_2 & \dots & v_n \end{bmatrix}$$

T denotes the transpose operation

- You need to keep track of orientation

Vectors have two main uses



- Vectors can represent an offset in 2D or 3D space
- Points are just vectors from the origin

- Data can also be treated as a vector
- Such vectors don't have a geometric interpretation, but calculations like "distance" still have value

Matrix

- A matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ is an array of numbers with size $m \downarrow$ by $n \rightarrow$, i.e. m rows and n columns.

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \vdots & & & & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{bmatrix}$$

- If $m = n$, we say that $\mathbf{A}$ is square.

Matrix Operations

- Addition

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} + \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} a+1 & b+2 \\ c+3 & d+4 \end{bmatrix}$$

  – Can only add a matrix with matching dimensions, or a scalar.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} + 7 = \begin{bmatrix} a+7 & b+7 \\ c+7 & d+7 \end{bmatrix}$$

- Scaling

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \times 3 = \begin{bmatrix} 3a & 3b \\ 3c & 3d \end{bmatrix}$$

Inner vs outer vs matrix vs element-wise product

- ***x, y*** = column vectors (nx1)
- ***X, Y*** = matrices (mxn)
- *x, y* = scalars (1x1)

- $\mathbf{x} \cdot \mathbf{y} = \mathbf{x}^T \mathbf{y}$ = inner product (1xn x nx1 = scalar)
- $\mathbf{x} \otimes \mathbf{y} = \mathbf{x}\,\mathbf{y}^T$ = outer product (nx1 x 1xn = matrix)

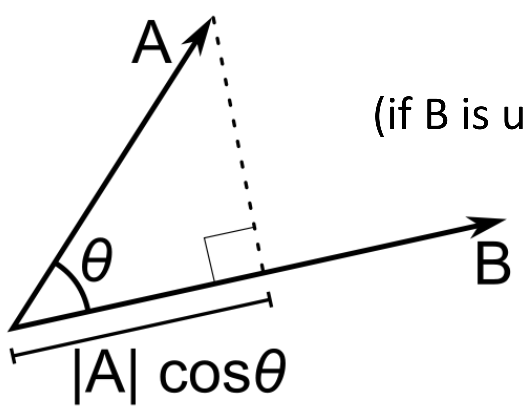
- $\mathbf{X} * \mathbf{Y}$ = matrix product
  - Watch out: could also be element-wise product in NumPy, if class is array rather than matrix– see tutorial

Inner Product

- Multiply corresponding entries of two vectors and add up the result

$$\mathbf{x}^T\mathbf{y} = \begin{bmatrix} x_1 & \dots & x_n \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \sum_{i=1}^{n} x_i y_i \qquad \text{(scalar)}$$

- x·y is also |x||y|Cos(*angle between x and y*)
- If B is a unit vector, then A·B gives the length of A which lies in the direction of B (projection)



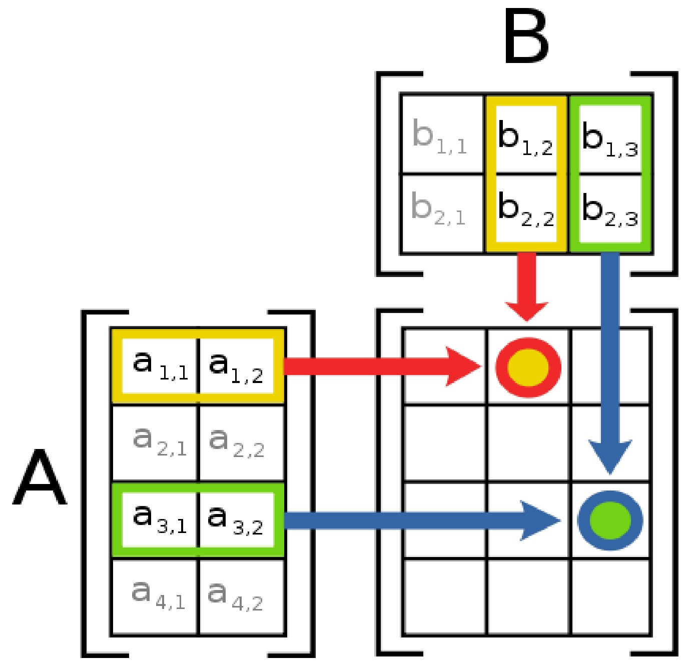
(if B is unit-length hence norm is 1)

Matrix Multiplication

- Let X be an *a*x*b* matrix, Y be an *b*x*c* matrix
- Then Z = X*Y is an *a*x*c* matrix
- Second dimension of first matrix, and first dimension of second matrix have to be the same, for matrix multiplication to be possible
- Practice: Let X be an 10x5 matrix. Let's factorize it into 3 matrices…

Matrix Multiplication

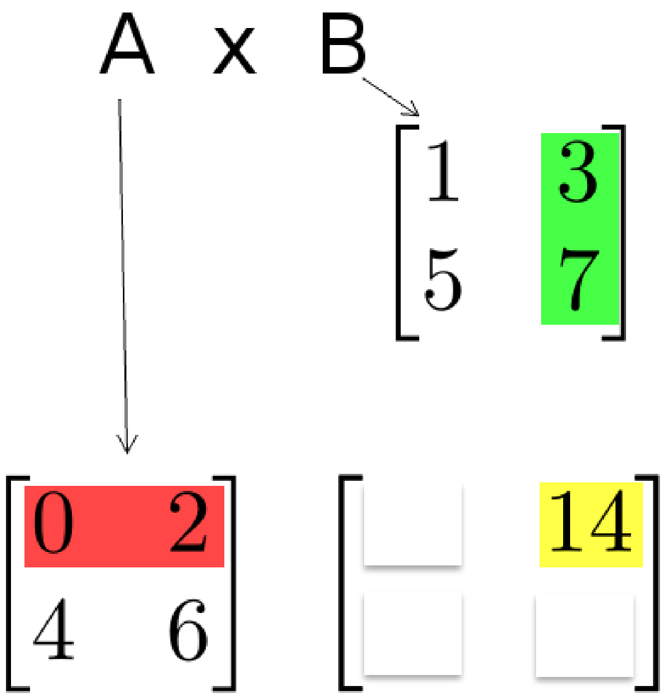
- The product AB is:



- Each entry in the result is (that row of A) dot product with (that column of B)

Matrix Multiplication

- Example:



$$0 \cdot 3 + 2 \cdot 7 = 14$$

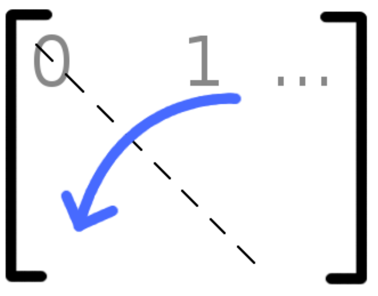
– Each entry of the matrix product is made by taking the dot product of the corresponding row in the left matrix, with the corresponding column in the right one.

Matrix Operation Properties

- Matrix addition is commutative and associative
  - A + B = B + A
  - A + (B + C) = (A + B) + C
- Matrix multiplication is associative and distributive but *not* commutative
  - A(B*C) = (A*B)C
  - A(B + C) = A*B + A*C
  - A*B != B*A

Matrix Operations

- Transpose – flip matrix, so row 1 becomes column 1



$$\begin{bmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{bmatrix}^T = \begin{bmatrix} 0 & 2 & 4 \\ 1 & 3 & 5 \end{bmatrix}$$

- A useful identity:

$$(ABC)^T = C^T B^T A^T$$

Inverse

- Given a matrix **A**, its inverse $\mathbf{A}^{-1}$ is a matrix such that $\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$

- E.g. $\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{3} \end{bmatrix}$

- Inverse does not always exist. If $\mathbf{A}^{-1}$ exists, **A** is *invertible* or *non-singular*. Otherwise, it's *singular*.

Special Matrices

- Identity matrix **I**
  - Square matrix, 1's along diagonal, 0's elsewhere
  - **I** · [another matrix] = [that matrix]

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Diagonal matrix
  - Square matrix with numbers along diagonal, 0's elsewhere
  - A diagonal · [another matrix] scales the rows of that matrix

$$\begin{bmatrix} 3 & 0 & 0 \\ 0 & 7 & 0 \\ 0 & 0 & 2.5 \end{bmatrix}$$

Special Matrices

- Symmetric matrix

$$\mathbf{A}^T = \mathbf{A}$$

$$\begin{bmatrix} 1 & 2 & 5 \\ 2 & 1 & 7 \\ 5 & 7 & 1 \end{bmatrix}$$

Norms

- L1 norm

$$\|\boldsymbol{x}\|_1 := \sum_{i=1}^{n} |x_i|$$

- L2 norm

$$\|\boldsymbol{x}\| := \sqrt{x_1^2 + \cdots + x_n^2}$$

- L^p norm (for real numbers $p \geq 1$)

$$\|\mathbf{x}\|_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

System of Linear Equations

- MATLAB example

// linalg.solve or lingalg.lstsq in Python

$$AX = B$$

$$A = \begin{bmatrix} 2 & 2 \\ 3 & 4 \end{bmatrix}, B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

```
>> x = A\B
x =
    1.0000
   -0.5000
```

Matrix Rank

- Column/row rank

$$\text{col-rank}(\mathbf{A}) = \text{ the maximum number of linearly independent column vectors of } \mathbf{A}$$
$$\text{row-rank}(\mathbf{A}) = \text{ the maximum number of linearly independent row vectors of } \mathbf{A}$$

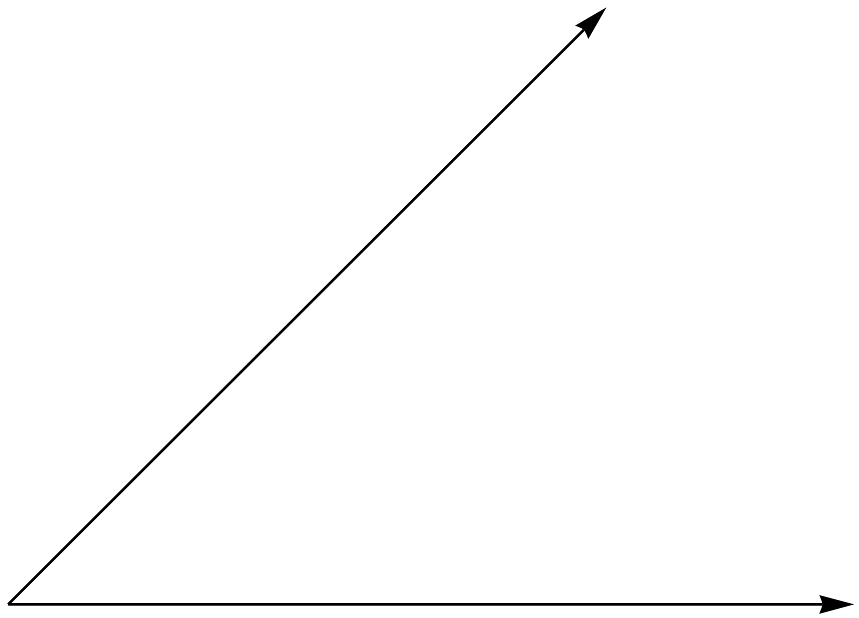
- Column rank always equals row rank
- Matrix rank $\text{rank}(\mathbf{A}) \triangleq \text{col-rank}(\mathbf{A}) = \text{row-rank}(\mathbf{A})$
- If a matrix is not full rank, inverse doesn't exist
  - Inverse also doesn't exist for non-square matrices

Linear independence

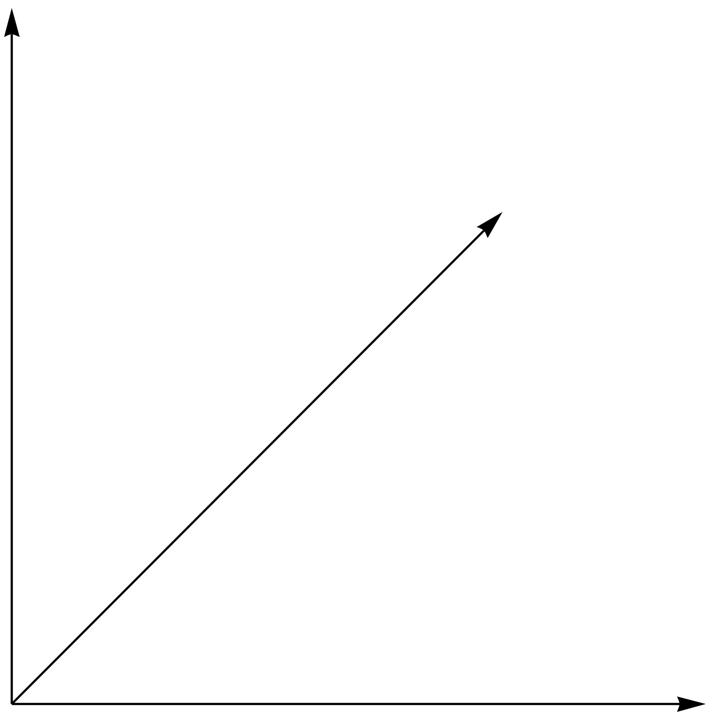
- Suppose we have a set of vectors $\mathbf{v_1, \ldots, v_n}$
- If we can express $\mathbf{v}_1$ as a linear combination of the other vectors $\mathbf{v}_2 \ldots \mathbf{v}_\mathrm{n}$, then $\mathbf{v}_1$ is linearly *dependent* on the other vectors.
  - The direction $\mathbf{v}_1$ can be expressed as a combination of the directions $\mathbf{v}_2 \ldots \mathbf{v}_\mathrm{n}$. (E.g. $\mathbf{v}_1 = .7\ \mathbf{v}_2 - .5\ \mathbf{v}_4$)
- If no vector is linearly dependent on the rest of the set, the set is linearly *independent*.
  - Common case: a set of vectors $\mathbf{v_1, \ldots, v_n}$ is always linearly independent if each vector is perpendicular to every other vector (and non-zero)

Linear independence

Linearly independent set



Not linearly independent



Singular Value Decomposition (SVD)

- There are several computer algorithms that can “factor” a matrix, representing it as the product of some other matrices
- The most useful of these is the Singular Value Decomposition
- Represents any matrix **A** as a product of three matrices: $\mathbf{U\Sigma V^T}$

Singular Value Decomposition (SVD)

$$\mathbf{U\Sigma V^T = A}$$

- Where **U** and **V** are rotation matrices, and **Σ** is a scaling matrix. For example:

$$\overset{U}{\begin{bmatrix} -.40 & .916 \\ .916 & .40 \end{bmatrix}} \times \overset{\Sigma}{\begin{bmatrix} 5.39 & 0 \\ 0 & 3.154 \end{bmatrix}} \times \overset{V^T}{\begin{bmatrix} -.05 & .999 \\ .999 & .05 \end{bmatrix}} = \overset{A}{\begin{bmatrix} 3 & -2 \\ 1 & 5 \end{bmatrix}}$$

Singular Value Decomposition (SVD)

- In general, if **A** is *m* x *n*, then **U** will be *m* x *m*, **Σ** will be *m* x *n*, and $\mathbf{V}^{\mathbf{T}}$ will be *n* x *n*.

$$\overset{U}{\begin{bmatrix} -.39 & -.92 \\ -.92 & .39 \end{bmatrix}} \times \overset{\Sigma}{\begin{bmatrix} 9.51 & 0 & 0 \\ 0 & .77 & 0 \end{bmatrix}} \times \overset{V^T}{\begin{bmatrix} -.42 & -.57 & -.70 \\ .81 & .11 & -.58 \\ .41 & -.82 & .41 \end{bmatrix}} = \overset{A}{\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}}$$

Singular Value Decomposition (SVD)

- **U** and **V** are always rotation matrices.
  - Geometric rotation may not be an applicable concept, depending on the matrix. So we call them “unitary” matrices – each column is a unit vector.
- **Σ** is a diagonal matrix
  - The number of nonzero entries = rank of **A**
  - The algorithm always sorts the entries high to low

$$\overset{U}{\begin{bmatrix} -.39 & -.92 \\ -.92 & .39 \end{bmatrix}} \times \overset{\Sigma}{\begin{bmatrix} 9.51 & 0 & 0 \\ 0 & .77 & 0 \end{bmatrix}} \times \overset{V^T}{\begin{bmatrix} -.42 & -.57 & -.70 \\ .81 & .11 & -.58 \\ .41 & -.82 & .41 \end{bmatrix}} = \overset{A}{\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}}$$

Singular Value Decomposition (SVD)

$$\mathbf{M = U\Sigma V^T}$$

Illustration from Wikipedia

Calculus review

Differentiation

The derivative provides us information about the rate of change of a function.

The derivative of a function is also a function.

Example:

> The derivative of the rate function is the acceleration function.

Derivative = rate of change

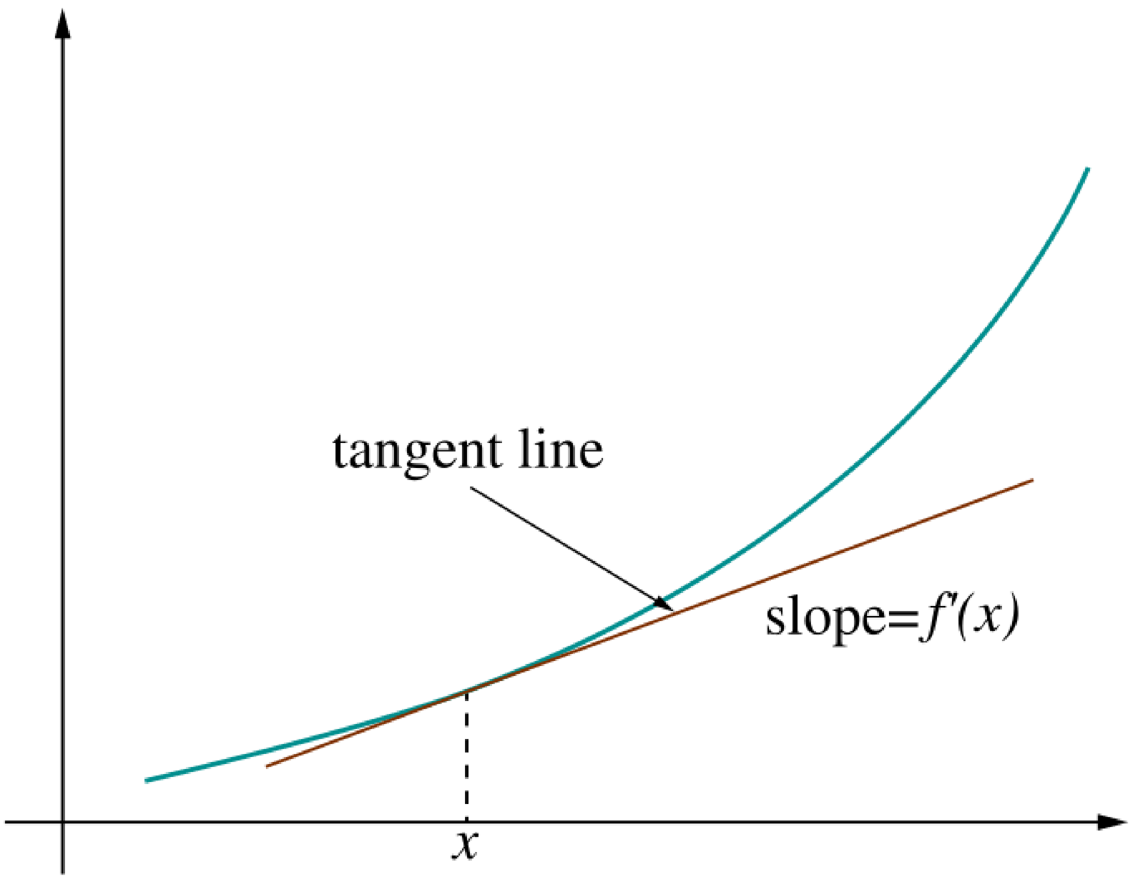


Image: Wikipedia

Derivative = rate of change

- Linear function y = mx + b
- Slope $\boldsymbol{m = \frac{\text{change in } y}{\text{change in } x} = \frac{\Delta y}{\Delta x}}$,

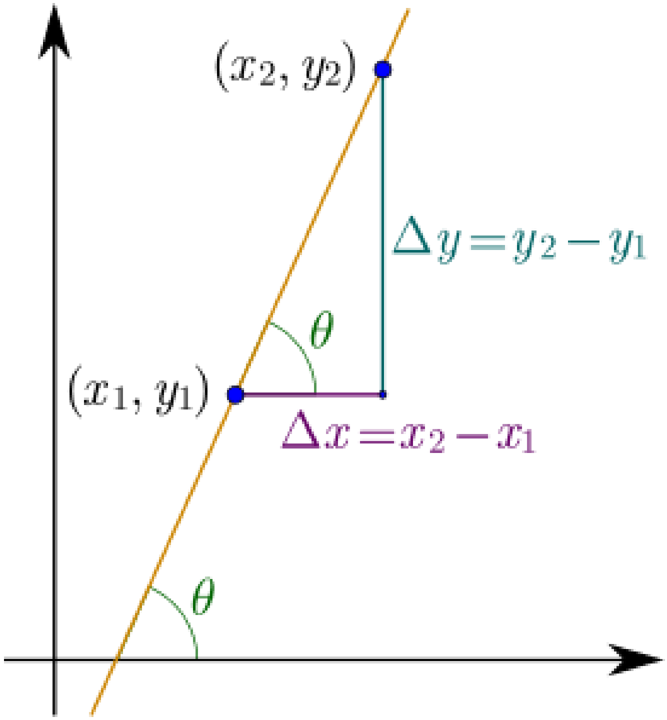


Image: Wikipedia

Ways to Write the Derivative

Given the function f(x), we can write its derivative in the following ways:

- $f'(x)$

- $\frac{d}{dx}f(x)$

The derivative of x is commonly written dx.

Differentiation Formulas

The following are common differentiation formulas:

- The derivative of a constant is 0.

$$\frac{d}{du}c = 0$$

- The derivative of a sum is the sum of the derivatives.

$$\frac{d}{du}(f(u) + g(u)) = f'(u) + g'(u)$$

Examples

- The derivative of a constant is 0.

$$\frac{d}{du}7 =$$

- The derivative of a sum is the sum of the derivatives.

$$\frac{d}{dt}(t + 4) =$$

More Formulas

- The derivative of *u* to a constant power:

$$\frac{d}{du}u^n = n*u^{n-1}du$$

- The derivative of *e*:

$$\frac{d}{du}e^u = e^u du$$

- The derivative of *log*:

$$\frac{d}{du}\log(u) = \frac{1}{u}du$$

More Examples

- The derivative of *u* to a constant power:

$$\frac{d}{dx}3x^3 =$$

- The derivative of *e*:

$$\frac{d}{dy}e^{4y} =$$

- The derivative of *log*:

$$\frac{d}{dx}3\log(x) =$$

Product and Quotient

The product rule and quotient rules are commonly used in differentiation.

- Product rule:

$$\frac{d}{du}(f(u)*g(u)) = f(u)g'(u) + g(u)f'(u)$$

- Quotient rule:

$$\frac{d}{du}\left(\frac{f(u)}{g(u)}\right) = \frac{g(u)f'(u) - f(u)g'(u)}{(g(u))^2}$$

Chain Rule

The chain rule allows you to combine any of the differentiation rules we have already covered.

- First, do the derivative of the outside and then do the derivative of the inside.

$$\frac{d}{du} f(g(u)) = f'(g(u)) * g'(u) * du$$

Try These

$$f(z) = z+11$$

$$s(y) = 4ye^{2y}$$

$$g(y) = 4y^3 + 2y$$

$$p(x) = \frac{\log(x^2)}{x}$$

$$h(x) = e^{3x}$$

$$q(z) = (e^z - z)^3$$

Solutions

$$f'(z) = 1$$

$$s'(y) = 8ye^{2y} + 4e^{2y}$$

$$g'(y) = 12y^2 + 2$$

$$p'(x) = \frac{2 - \log(x^2)}{x^2}$$

$$h'(x) = 3e^{3x}$$

$$q'(z) = 3(e^z - z)^2(e^z - 1)$$