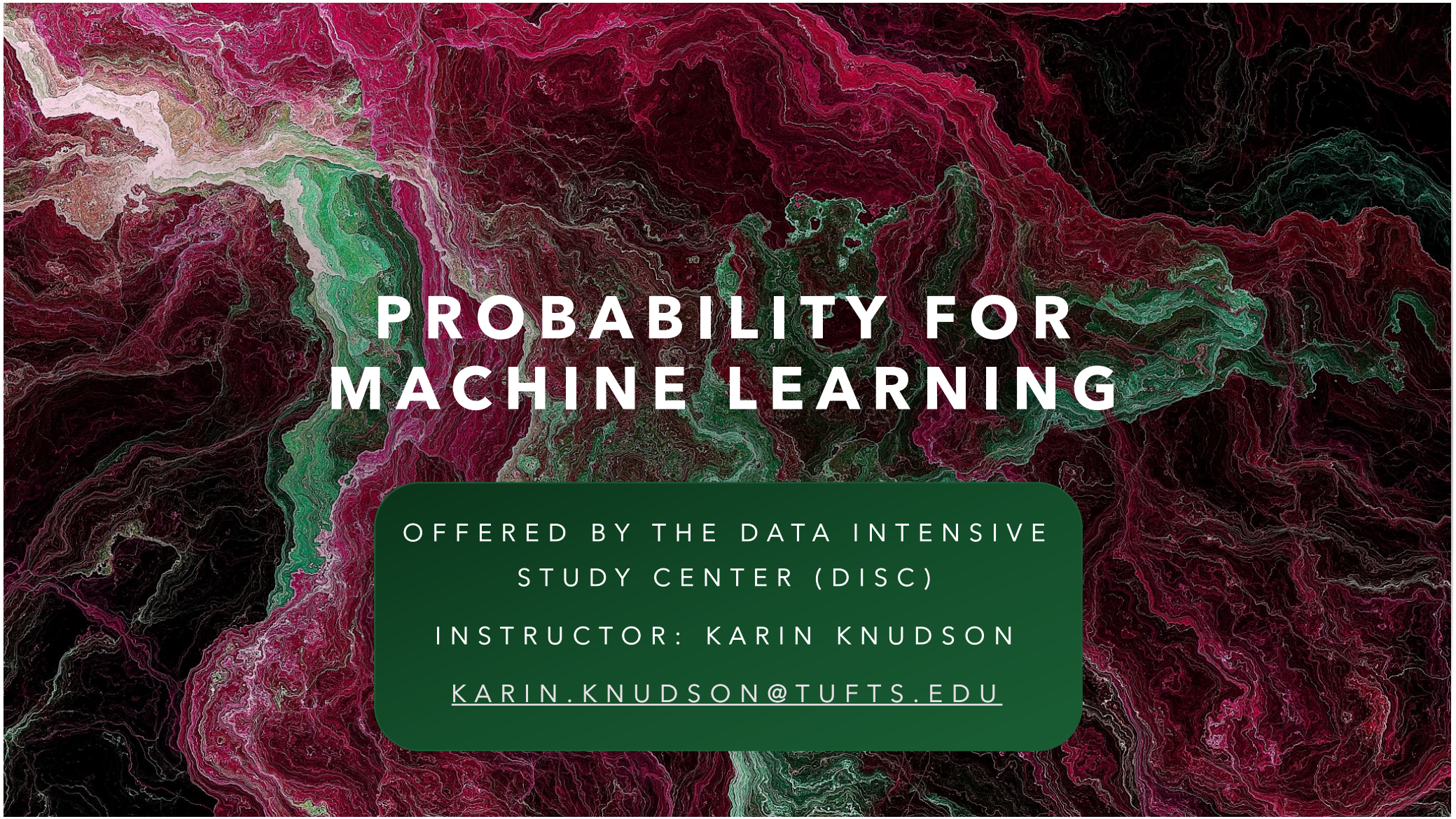
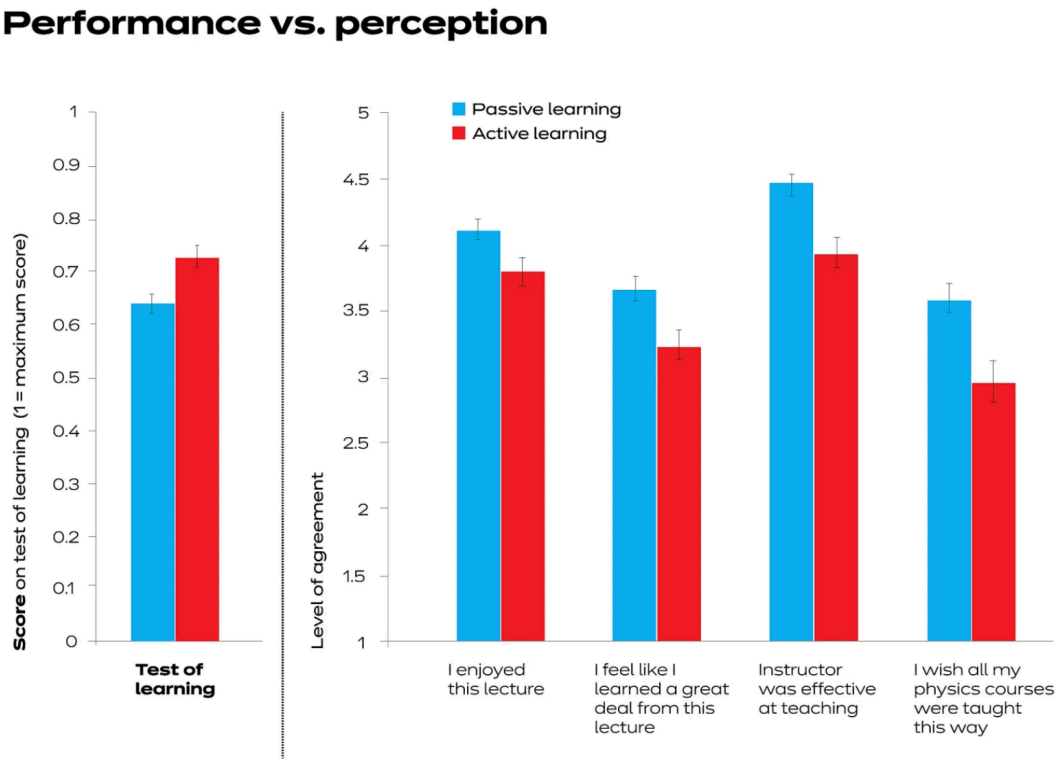

PROBABILITY FOR MACHINE LEARNING
OFFERED BY THE DATA INTENSIVE STUDY CENTER (DISC)
INSTRUCTOR: KARIN KNUDSON
KARIN.KNUDSON@TUFTS.EDU

DAILY FORMAT

- Whole group instruction
- Hands-on practice in breakout groups
- Breaks

Source: "Measuring actual learning versus feeling of learning in response to being actively engaged in the classroom," Louis Deslauriers, Logan S. McCarty, Kelly Miller, Kristina Callaghan, and Greg Kestin



https://news.harvard.edu/gazette/story/2019/09/study-shows-that-students-learn-more-when-taking-part-in-classrooms-that-employ-active-learning-strategies/

WEEK SCHEDULE

- Monday: Probability foundations
- Tuesday: Regression and classification*
- Wednesday: Regression and classification*
- Thursday: Unsupervised learning*
- Friday: Misc: (time series, deep learning, other topics)*

* Will be both exploring specific techniques and using these techniques to illustrate broader probabilistic principles

"The probability of flipping heads is 50%"

– what does this mean?

"The probability of flipping heads is 50%" – what does this mean?

- If we flip the coin many times, we expect about half of the flips to be heads
- 50% quantifies our degree of belief that the coin will be heads the next time we flip
- We might pay 50 cents to play a game where the coin is flipped once and we get 1 dollar if it is heads

RANDOM VARIABLES

- Discrete random variable – countable number of possible values:

  Coin flip: {heads, tails}

  Die roll: {1, 2, 3, 4, 5, 6}

  Number of times a neuron fires in a time interval: {0, 1, 2, …}

- Continuous random variable – uncountable number of possible values:

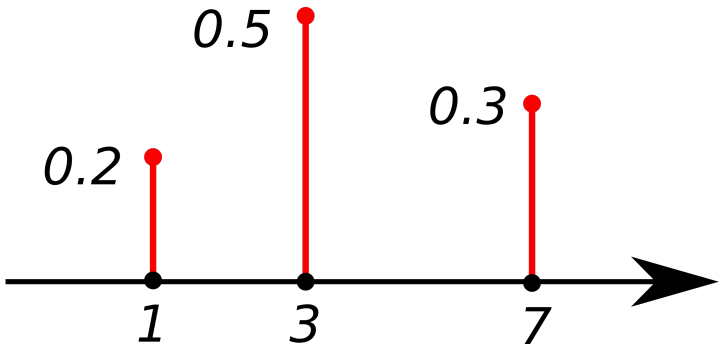
  Temperature outside

  Bias of a coin: [0, 1]

- Distinction can blur– e.g. approximating the count of vehicles on a road in a day with a continuous distribution
- Set of possible values a random variables: support

DISCRETE

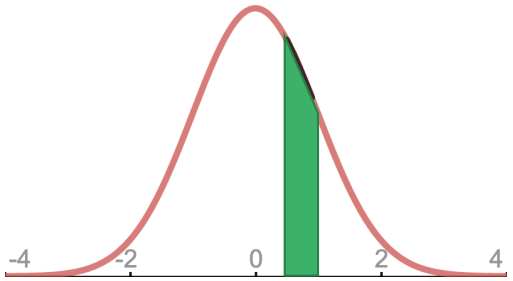
probability mass function (pmf)



$$\sum p(x_i) = 1$$
$$p(x_i) = P(X = x_i) \geq 0$$

CONTINUOUS

probability density function pdf



$$P(X \in (a, b)) = \int_a^b p(x)\, dx$$
$$\int p(x)\, dx = 1$$
$$p(x) \geq 0$$

EXPECTATION

$$\mathbb{E}(X) = \sum xp(x)$$
$$\mathbb{E}(f(x)) = \sum f(x)p(x)$$

$$\mathbb{E}(X) = \int xp(x)\,dx$$
$$\mathbb{E}(f(x)) = \int f(x)p(x)\,dx$$

Weighted average (weighted by probability)

Linear: $\mathbb{E}(aX + b) = \sum (ax + b)p(x)$

$$= \sum axp(x) + \sum bp(x)$$
$$= a\sum xp(x) + b\sum p(x)$$
$$= a\mathbb{E}(X) + b$$

VARIANCE

$$\mathrm{var}(X) = \mathbb{E}[(X - \mathbb{E}(X))^2]$$

Expected squared difference from the mean

$$\mathrm{var}(X) = \mathbb{E}(X^2) - [\mathbb{E}(X)]^2$$

Proof: (exercise)

$$\mathrm{var}(aX + b) = \ ??\ \ (\mathrm{exercise})$$

COVARIANCE

$$\begin{aligned}\operatorname{cov}(\mathrm{X},\mathrm{Y}) &= \mathbb{E}[(X-\mathbb{E}(X))(Y-\mathbb{E}(Y))] \\ &= \mathbb{E}(XY) - \mathbb{E}(X)\mathbb{E}(Y)\end{aligned}$$

	x = 1	x=2	
y=1	.1	.3	
y=2	.2	.1	
y=3	.2	0	
y=4	0	.1	

Joint probability
$P(X=2,\ Y=1)$

	x = 1	x=2	
y=1	.1	.3	.4
y=2	.2	.1	.3
y=3	.2	0	.2
y=4	0	.1	.1
	.5	.5	

Joint probability
P(X=2, Y= 1)

Marginal probability
P(Y=2)
$= \sum_x p(Y = 2, X = x)$

	x = 1	x=2	
y=1	.1	.3	.4
y=2	.2	.1	.3
y=3	.2	0	.2
y=4	0	.1	.1
	.5	.5	

Marginal distribution of Y

Joint probability
P(X=2, Y= 1)

Marginal probability
P(Y=2)
$= \sum_x p(Y = 2, X = x)$

	x = 1	x=2	
y=1	.1	.3	.4
y=2	.2	.1	.3
y=3	.2	0	.2
y=4	0	.1	.1
	.5	.5	

Joint probability
P(X=2, Y= 1)

Marginal probability
P(Y=2)

$$= \sum_x p(Y = 2, X = x)$$

Marginal probability
P(X=2)

$$= \sum_y p(X = 2, Y = y)$$

	x = 1	x=2	
y=1	.1	.3	.4
y=2	.2	.1	.3
y=3	.2	0	.2
y=4	0	.1	.1
	.5	.5	

Joint probability
P(X=2, Y= 1)

Marginal probability
P(Y=2)

$$= \sum_x p(Y = 2, X = x)$$

Marginal probability
P(X=2)

$$= \sum_y p(X = 2, Y = y)$$

	x = 1	x=2	
y=1	.1	.3	.4
y=2	.2	.1	.3
y=3	.2	0	.2
y=4	0	.1	.1
	.5	.5	

Joint probability
P(X=2, Y= 1)

Marginal probability
P(Y=2)

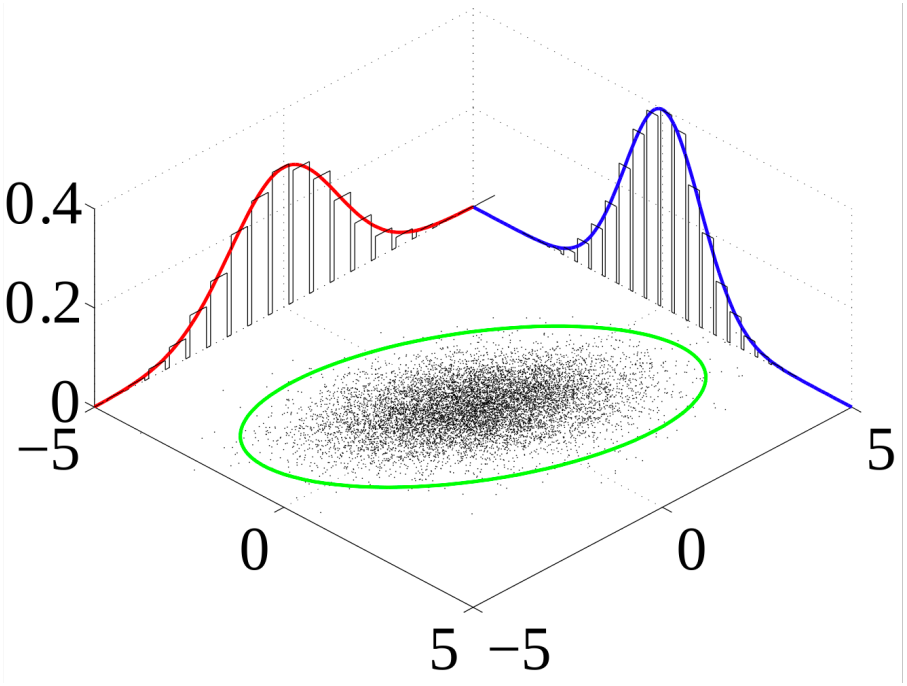
$$= \sum_x p(Y = 2, X = x)$$

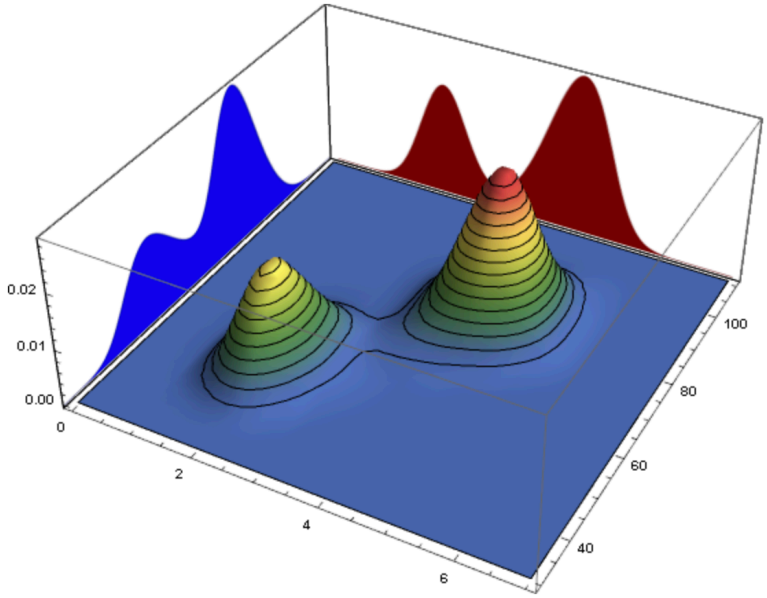
Marginal probability
P(X=2)

$$= \sum_y p(X = 2, Y = y)$$

Conditional probability:

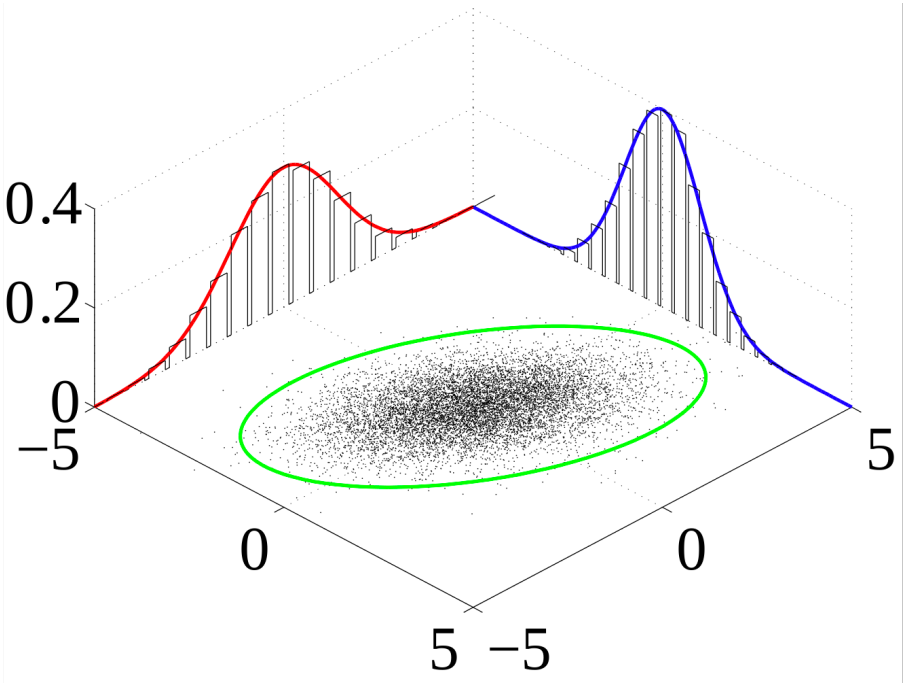
$$P(X = 2|Y = 2) = \frac{P(X=2,Y=2)}{P(Y=2)} = \frac{.1}{.3} = \frac{1}{3}$$

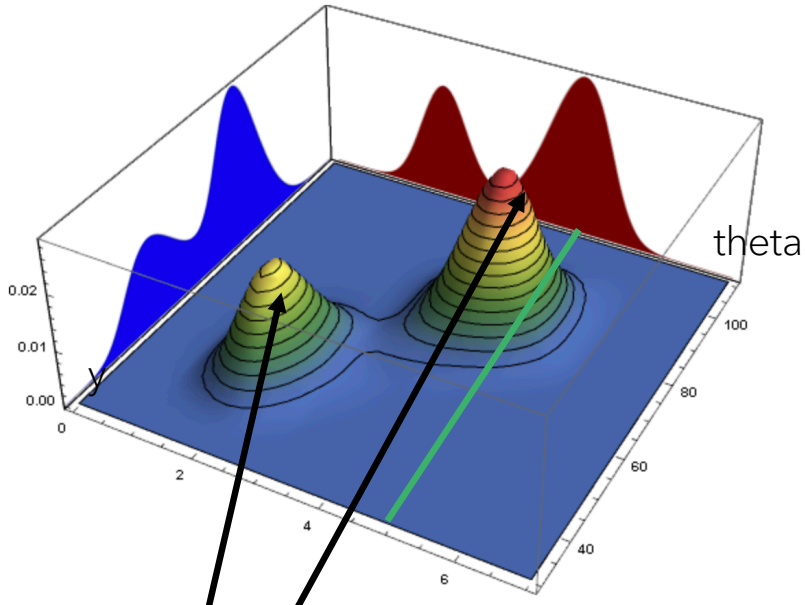




https://www.wolfram.com/mathematica/new-in-8/statistical-visualization/visualize-marginal-distributions-for-estimated-dis.html

CONTINUOUS MULTIVARIATE DISTRIBUTIONS





https://www.wolfram.com/mathematica/new-in-8/statistical-visualization/visualize-marginal-distributions-for-estimated-dis.html

modes

(this is a **multimodal** distribution)

SUM RULE

$$P(X) = \sum_{Y} P(X, Y)$$

$$p(X) = \int p(X, Y)\, dY$$

PRODUCT RULE

$$P(X, Y) = P(X|Y)P(Y)$$

Conditional probability

$$P(X|Y) = \frac{P(X, Y)}{P(Y)}$$

INDEPENDENCE

We say X and Y are **independent** if $P(X, Y) = P(X)\,P(Y)$

BAYES THEOREM

How to Think Like an Epidemiologist

Don't worry, a little Bayesian analysis won't hurt you.

https://www.nytimes.com/2020/08/04/science/coronavirus-bayes-statistics-math.html

BAYES THEOREM

$$P(\theta|Y) = \frac{P(Y|\theta)P(\theta)}{P(Y)}$$

BAYES THEOREM

$$P(\theta|Y) = \frac{P(Y|\theta)P(\theta)}{P(Y)}$$

$$= \frac{P(Y|\theta)P(\theta)}{\sum_\theta P(Y|\theta)P(\theta)}$$

BAYES THEOREM

$$\overbrace{P(\theta|Y)}^{\text{posterior}} = \frac{\overbrace{P(Y|\theta)}^{\text{likelihood}}\overbrace{P(\theta)}^{\text{prior}}}{P(Y)}$$

Y = observed data
θ= parameters of interest (unknown)

$$= \frac{P(Y|\theta)P(\theta)}{\sum_\theta P(Y|\theta)P(\theta)}$$

EX: ESTIMATING BIAS OF A COIN

$$\overbrace{P(\theta|Y)}^{\text{posterior}} = \frac{\overbrace{P(Y|\theta)}^{\text{likelihood}}\overbrace{P(\theta)}^{\text{prior}}}{P(Y)}$$

Y = observed data = # of heads observed in 20 flips
θ = parameter of interest = probability that the coin lands heads

EX: ESTIMATING BIAS OF A COIN

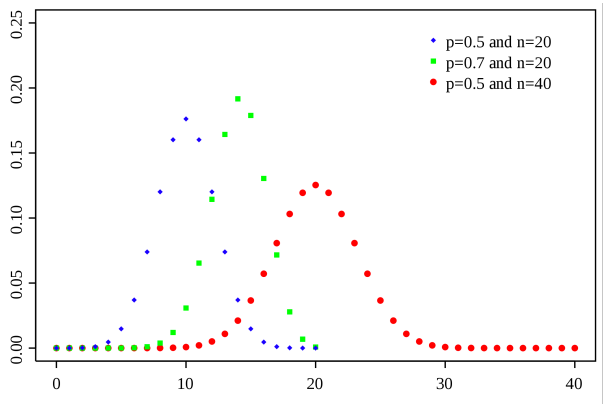
$$\underbrace{P(\theta|Y)}_{\text{posterior}} = \frac{\overbrace{P(Y|\theta)}^{\text{likelihood}}\overbrace{P(\theta)}^{\text{prior}}}{P(Y)}$$

Y = observed data = # of heads observed in 20 flips
θ = parameter of interest = probability that the coin lands heads

likelihood

$$Y|\theta \sim \text{Binomial}(20, \theta)$$

$$p(Y = y|\theta) = \binom{20}{y}\theta^y(1-\theta)^{20-y}$$



EX: ESTIMATING BIAS OF A COIN

$$\underbrace{P(\theta|Y)}_{\text{posterior}} = \frac{\overbrace{P(Y|\theta)}^{\text{likelihood}}\overbrace{P(\theta)}^{\text{prior}}}{P(Y)}$$

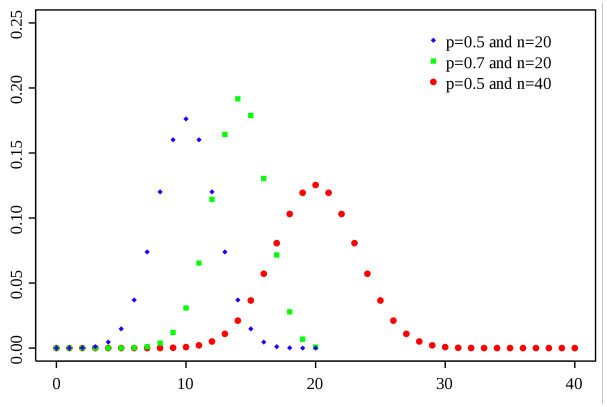
Y = observed data = # of heads observed in 20 flips

θ = parameter of interest = probability that the coin lands heads

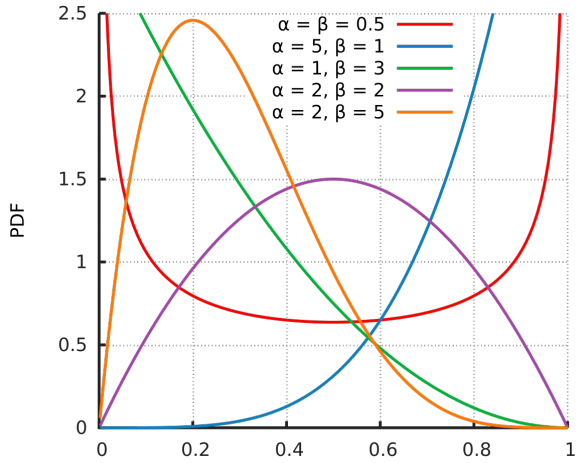
likelihood

$$Y|\theta \sim \text{Binomial}(20, \theta)$$

$$p(Y = y|\theta) = \binom{20}{y}\theta^y(1-\theta)^{20-y}$$



prior



$$\theta \sim \text{Beta}(a, b)$$

$$p(\theta) = \frac{\theta^{a-1}(1-\theta)^{b-1}}{B(a,b)}$$

EX: ESTIMATING BIAS OF A COIN

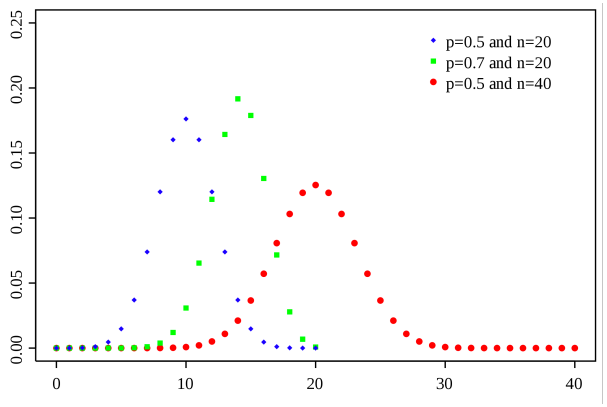
$$\underbrace{P(\theta|Y)}_{\text{posterior}} = \frac{\overbrace{P(Y|\theta)}^{\text{likelihood}}\overbrace{P(\theta)}^{\text{prior}}}{P(Y)}$$

Y = observed data = # of heads observed in 20 flips
θ = parameter of interest = probability that the coin lands heads

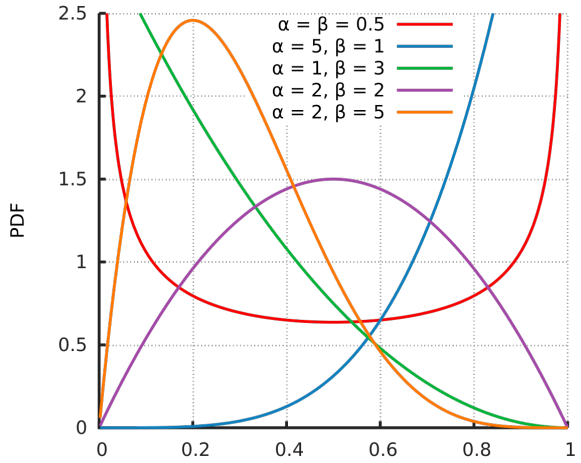
likelihood

$$Y|\theta \sim \text{Binomial}(20, \theta)$$

$$p(Y = y|\theta) = \binom{20}{y}\theta^y(1-\theta)^{20-y}$$



prior



$$\theta \sim \text{Beta}(a, b) \qquad \theta \sim \text{Beta}(2, 2)$$

$$p(\theta) = \frac{\theta^{a-1}(1-\theta)^{b-1}}{B(a,b)} \qquad p(\theta) = \frac{\theta(1-\theta)}{B(2,2)}$$

EX: ESTIMATING BIAS OF A COIN

$$\underbrace{P(\theta|Y)}_{\text{posterior}} = \frac{\overbrace{P(Y|\theta)}^{\text{likelihood}}\overbrace{P(\theta)}^{\text{prior}}}{P(Y)}$$

Y = observed data = # of heads observed in 20 flips
θ = parameter of interest = probability that the coin lands heads

likelihood

$$Y \sim \text{Binomial}(20, \theta)$$

$$p(Y = y|\theta) = \binom{20}{y}\theta^y(1-\theta)^{20-y}$$

prior

$$\theta \sim \text{Beta}(a, b) \qquad \theta \sim \text{Beta}(2, 2)$$

$$p(\theta) = \frac{\theta^{a-1}(1-\theta)^{b-1}}{B(a,b)} \qquad p(\theta) = \frac{\theta(1-\theta)}{B(2,2)}$$

posterior

EX: ESTIMATING BIAS OF A COIN

$$\underbrace{P(\theta|Y)}_{\text{posterior}} = \frac{\overbrace{P(Y|\theta)}^{\text{likelihood}}\overbrace{P(\theta)}^{\text{prior}}}{P(Y)}$$

Y = observed data = # of heads observed in 20 flips
θ = parameter of interest = probability that the coin lands heads

likelihood

$$Y \sim \text{Binomial}(20, \theta)$$

$$p(Y = y|\theta) = \binom{20}{y}\theta^y(1-\theta)^{20-y}$$

prior

$$\theta \sim \text{Beta}(a, b) \qquad \theta \sim \text{Beta}(2, 2)$$

$$p(\theta) = \frac{\theta^{a-1}(1-\theta)^{b-1}}{B(a,b)} \qquad p(\theta) = \frac{\theta(1-\theta)}{B(2,2)}$$

posterior

$$p(\theta|Y = y) = \frac{p(Y=y|\theta)p(\theta)}{p(Y=y)} \propto p(Y = y|\theta)p(\theta)$$

EX: ESTIMATING BIAS OF A COIN

$$\underbrace{P(\theta|Y)}_{\text{posterior}} = \frac{\overbrace{P(Y|\theta)}^{\text{likelihood}}\overbrace{P(\theta)}^{\text{prior}}}{P(Y)}$$

Y = observed data = # of heads observed in 20 flips
θ = parameter of interest = probability that the coin lands heads

likelihood

$$Y \sim \text{Binomial}(20, \theta)$$

$$p(Y = y|\theta) = \binom{20}{y}\theta^y(1-\theta)^{20-y}$$

prior

$$\theta \sim \text{Beta}(a, b) \qquad \theta \sim \text{Beta}(2, 2)$$

$$p(\theta) = \frac{\theta^{a-1}(1-\theta)^{b-1}}{B(a,b)} \qquad p(\theta) = \frac{\theta(1-\theta)}{B(2,2)}$$

posterior

$$p(\theta|Y = y) = \frac{p(Y=y|\theta)p(\theta)}{p(Y=y)} \propto p(Y = y|\theta)p(\theta)$$

$$\propto \binom{20}{y}\theta^y(1-\theta)^{20-y}\theta^{a-1}(1-\theta)^{b-1}$$

$$\propto \theta^y(1-\theta)^{20-y}\theta(1-\theta)$$

$$\propto \theta^{y+1}(1-\theta)^{21-y} = \theta^{y+2-1}(1-\theta)^{20-y+2-1}$$

"is proportional to"
- we drop terms in this product that don't depend on theta

EX: ESTIMATING BIAS OF A COIN

$$\underbrace{P(\theta|Y)}_{\text{posterior}} = \frac{\overbrace{P(Y|\theta)}^{\text{likelihood}}\overbrace{P(\theta)}^{\text{prior}}}{P(Y)}$$

Y = observed data = # of heads observed in 20 flips
θ = parameter of interest = probability that the coin lands heads

likelihood

$$Y \sim \text{Binomial}(20, \theta)$$
$$p(Y = y|\theta) = \binom{20}{y}\theta^y(1-\theta)^{20-y}$$

prior

$$\theta \sim \text{Beta}(a, b) \qquad \theta \sim \text{Beta}(2, 2)$$
$$p(\theta) = \frac{\theta^{a-1}(1-\theta)^{b-1}}{B(a,b)} \qquad p(\theta) = \frac{\theta(1-\theta)}{B(2,2)}$$

posterior

$$p(\theta|Y = y) = \frac{p(Y=y|\theta)p(\theta)}{p(Y=y)} \propto p(Y = y|\theta)p(\theta)$$
$$\propto \binom{20}{y}\theta^y(1-\theta)^{20-y}\theta^{a-1}(1-\theta)^{b-1}$$
$$\propto \theta^y(1-\theta)^{20-y}\theta(1-\theta)$$
$$\propto \theta^{y+1}(1-\theta)^{21-y} = \theta^{y+2-1}(1-\theta)^{20-y+2-1}$$

"is proportional to"
- we drop terms in this product that don't depend on theta

$$\theta|Y = y \sim \text{Beta}(y + 2, (20 - y) + 2)$$
$$\theta|Y = y \sim \text{Beta}(y + a, (20 - y) + b)$$

EX: ESTIMATING BIAS OF A COIN

$$\underbrace{P(\theta|Y)}_{\text{posterior}} = \frac{\overbrace{P(Y|\theta)}^{\text{likelihood}}\overbrace{P(\theta)}^{\text{prior}}}{P(Y)}$$

Y = observed data = # of heads observed in 20 flips
θ = parameter of interest = probability that the coin lands heads

likelihood

$$Y \sim \text{Binomial}(20, \theta)$$

$$p(Y = y|\theta) = \binom{20}{y}\theta^y(1-\theta)^{20-y}$$

prior

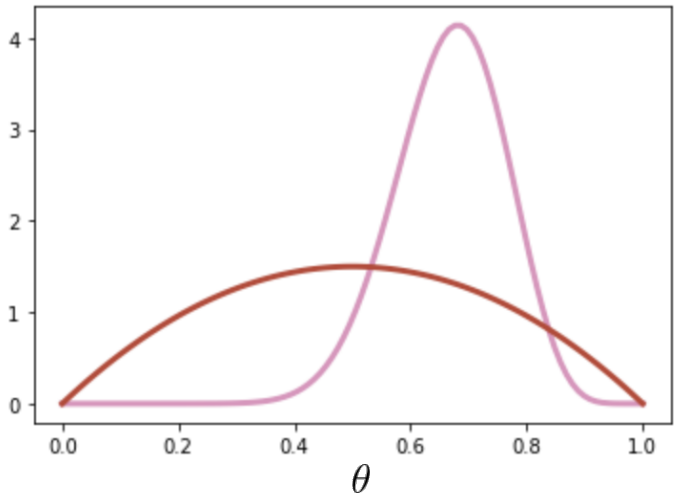
$$\theta \sim \text{Beta}(a, b) \qquad \theta \sim \text{Beta}(2, 2)$$

$$p(\theta) = \frac{\theta^{a-1}(1-\theta)^{b-1}}{B(a,b)} \qquad p(\theta) = \frac{\theta(1-\theta)}{B(2,2)}$$

posterior

data: y = 14

$$\theta|Y = 14 \sim \text{Beta}(16, 8)$$



EX: ESTIMATING BIAS OF A COIN: REFLECTION

- Wrote down our assumptions about the data generating process
- Used existing knowledge of coins to set a prior
- Computed the posterior distribution using the prior and the posterior, updating our beliefs about the parameter of interest
- Posterior distribution for theta had smaller variance than the prior distribution for theta
- Were “lucky” in that the posterior distribution turned out to have a convenient, recognizable form - this won’t always be the case. (When the prior and posterior have the same form, we have what we call conjugacy.)

REFLECTION

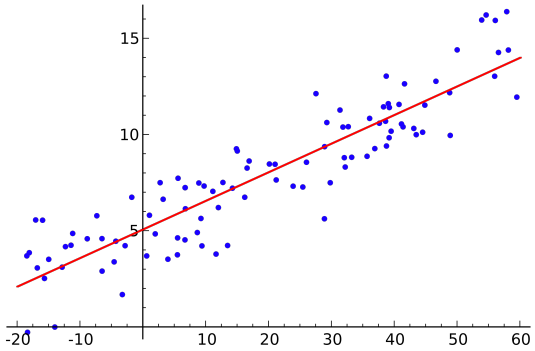
What did you learn?

What's something that made sense to you?

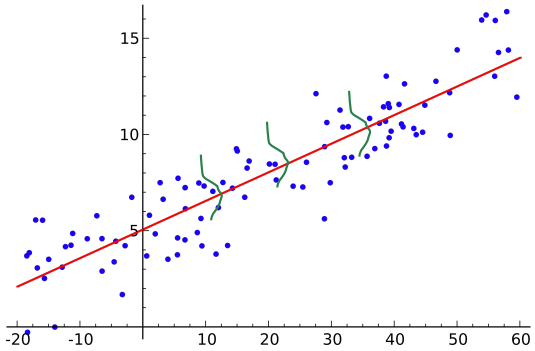
What was your "muddiest point" from today?

LINEAR REGRESSION

$$y = w_0 + w_1 X_1$$



LINEAR REGRESSION

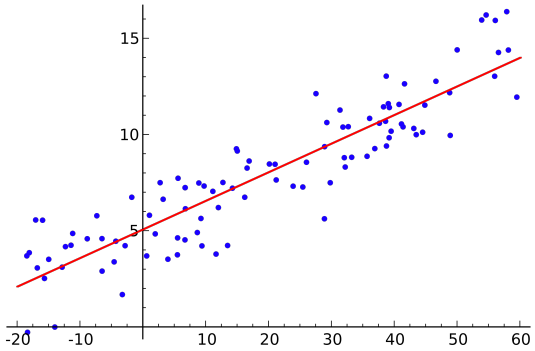


$$y = w_0 + w_1 X_1$$

$$y_i | X_i, w_0, w_1 \sim Normal(w_0 + w_1 X_1, \sigma^2)$$

$$p(y_i) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(y_i - (w_0 + w_1 X_i))^2}{2\sigma^2} \right)$$

LINEAR REGRESSION



$$y = w_0 + w_1 X_1$$

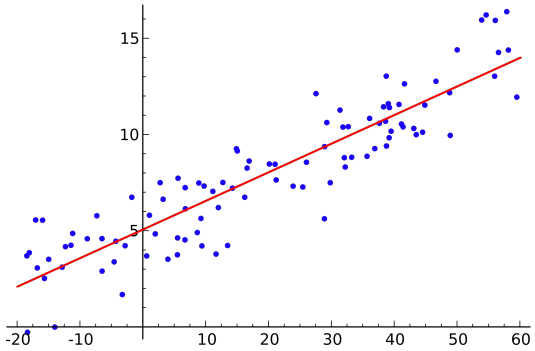
$$y_i | X_i, w_0, w_1 \sim Normal(w_0 + w_1 X_1, \sigma^2)$$

$$p(y_i) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(y_i - (w_0 + w_1 X_i))^2}{2\sigma^2} \right)$$

Likelihood

$$p(\mathbf{y}) = \prod_{i=1}^{N} \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(y_i - (w_0 + w_1 X_i))^2}{2\sigma^2} \right)$$

LINEAR REGRESSION



$$y = w_0 + w_1 X_1$$

$$y_i | X_i, w_0, w_1 \sim Normal(w_0 + w_1 X_1, \sigma^2)$$

$$p(y_i) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(y_i - (w_0 + w_1 X_i))^2}{2\sigma^2} \right)$$

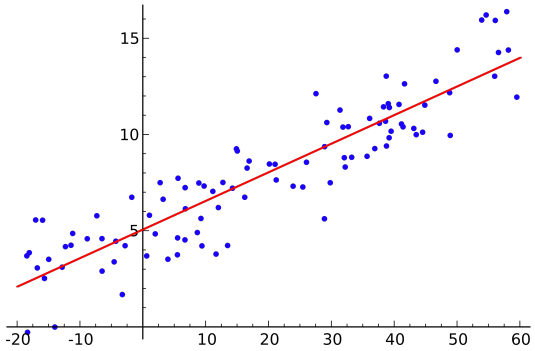
Likelihood

$$p(\mathbf{y}) = \prod_{i=1}^{N} \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(y_i - (w_0 + w_1 X_i))^2}{2\sigma^2} \right)$$

$$\log p(\mathbf{y} | X, w) = -\sum_{i=1}^{N} \left(\frac{(y_i - (w_0 + w_1 X_i))^2}{2\sigma^2} \right) + \sum_{i=1}^{N} \log(1/(2\pi\sigma^2))$$

Log-likelihood (recall log(ab) = log(a) + log(b))

LINEAR REGRESSION



$$y = w_0 + w_1 X_1$$

$$y_i | X_i, w_0, w_1 \sim Normal(w_0 + w_1 X_1, \sigma^2)$$

$$p(y_i) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(y_i - (w_0 + w_1 X_i))^2}{2\sigma^2} \right)$$

Likelihood

$$p(\mathbf{y}) = \prod_{i=1}^{N} \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(y_i - (w_0 + w_1 X_i))^2}{2\sigma^2} \right)$$

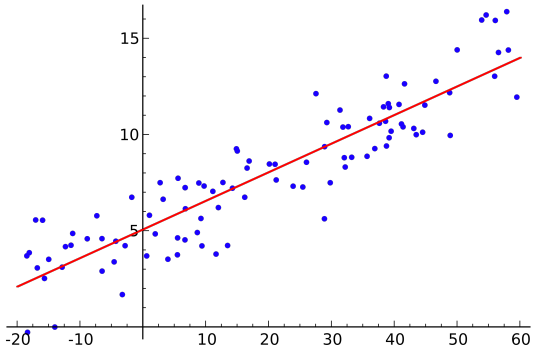
To maximize this with respect to w, it suffices to minimize:

$$\sum_{i=1}^{N} (y_i - (w_0 + w_1 X_i))^2$$

$$\log p(\mathbf{y} | X, w) = -\sum_{i=1}^{N} \left(\frac{(y_i - (w_0 + w_1 X_i))^2}{2\sigma^2} \right) + \sum_{i=1}^{N} \log(1/(2\pi\sigma^2))$$

Log-likelihood (recall log(ab) = log(a) + log(b))

LINEAR REGRESSION



- Log-likelihood easier to manage than the likelihood
- Maximize the likelihood by minimizing the **negative log-likelihood**
- Negative log-likelihood as the **loss function** that we seek to minimize
- Derived ordinary least squares setup from probabilistic assumptions about the data

LINEAR REGRESSION: INCLUDING BASIS FUNCTIONS

$$y = w_0 + w_1 x_1 + ... + w_D x_D$$

LINEAR REGRESSION: INCLUDING BASIS FUNCTIONS

$$y = w_0 + w_1 x_1 + \ldots + w_D x_D$$

Basis functions: give lots more flexibility!

$$y = w_0 + w_1 \phi_1(\mathbf{x}) + \ldots + w_{M-1} \phi_{M-1}(\mathbf{x})$$

$$y = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) \quad \xrightarrow{\text{Rewriting with vectors}} \quad y = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$$

$$\phi_0(\mathbf{x}) = 1$$

LINEAR REGRESSION: MAXIMUM LIKELIHOOD (AGAIN)

$$y_i|\mathbf{x}, \mathbf{w}, \sigma^2 \sim N(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i), \sigma^2)$$

LINEAR REGRESSION: MAXIMUM LIKELIHOOD (AGAIN)

$$y_i|\mathbf{x}, \mathbf{w}, \sigma^2 \sim N(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i), \sigma^2)$$

likelihood

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma^2) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2/2\sigma^2\right)$$

$$\log p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma^2) = \sum_{i=1}^{N} \left(-(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2/2\sigma^2\right) + \sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}}$$

$$= -\frac{1}{2\sigma^2} \sum_{i=1}^{N} (y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2 + N \log \frac{1}{\sqrt{2\pi\sigma^2}}$$

LINEAR REGRESSION: MAXIMUM LIKELIHOOD (AGAIN)

$$y_i|\mathbf{x}, \mathbf{w}, \sigma^2 \sim N(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i), \sigma^2)$$

likelihood

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma^2) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2/2\sigma^2\right)$$

$$\log p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma^2) = \sum_{i=1}^{N} \left(-(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2/2\sigma^2\right) + \sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}}$$

$$= -\frac{1}{2\sigma^2} \sum_{i=1}^{N} (y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2 + N \log \frac{1}{\sqrt{2\pi\sigma^2}}$$

Maximize likelihood by minimizing: $\sum_{i=1}^{N} (y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2$

LINEAR REGRESSION: MAXIMUM LIKELIHOOD (AGAIN)

$$y_i|\mathbf{x}, \mathbf{w}, \sigma^2 \sim N(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i), \sigma^2)$$

likelihood

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma^2) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2/2\sigma^2\right)$$

$$\log p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \sigma^2) = \sum_{i=1}^{N} \left(-(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2/2\sigma^2\right) + \sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}}$$

$$= -\frac{1}{2\sigma^2} \sum_{i=1}^{N} (y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2 + N \log \frac{1}{\sqrt{2\pi\sigma^2}}$$

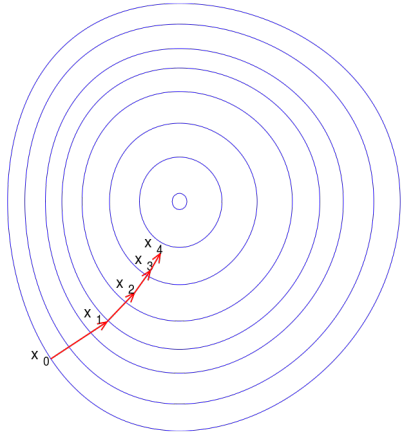
Moore-Penrose pseudo-inverse

Maximize likelihood by minimizing: $\sum_{i=1}^{N} (y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2$

$$\mathbf{w}_{ML} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{y} = \Phi^\dagger\mathbf{y}$$

where $\Phi_{ij} = \phi_j(\mathbf{x}_i)$

LINEAR REGRESSION AND GRADIENT DESCENT



Error function/loss function E(**w**):

$$\sum_{i=1}^{N}(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2$$

Gradient of E(**w**)

$$\nabla E(\mathbf{w}) = \sum_{i=1}^{N}(t_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))\boldsymbol{\phi}(\mathbf{x}_i)^T$$

Here, we can set to zero and solve, but we could also imagine moving iteratively:

$$\mathbf{w}^{\tau+1} = \mathbf{w}^{\tau} - \eta\nabla E(w)$$

We could also use **stochastic gradient descent**, where each step depends on the gradient of the term of the error function that depends only on *one data point:*

$$\mathbf{w}^{\tau+1} = \mathbf{w}^{\tau} - \eta\nabla E_n(w)$$

$$\text{in this example, } E_n = (y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))\boldsymbol{\phi}(\mathbf{x}_i)^T$$

LINEAR REGRESSION AND REGULARIZATION: RIDGE REGRESSION

Same likelihood: $y_i|\mathbf{x}, \mathbf{w}, \sigma^2 \sim N(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i), \sigma^2)$

Can we put a prior over **w** (params of interest)?

LINEAR REGRESSION AND REGULARIZATION: RIDGE REGRESSION

Same likelihood: $y_i|\mathbf{x}, \mathbf{w}, \sigma^2 \sim N(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i), \sigma^2)$

Can we put a prior over **w** (params of interest)?

hyperparameter!

$w|\alpha \sim N(0, \alpha^2)$

LINEAR REGRESSION AND REGULARIZATION: RIDGE REGRESSION

likelihood $y_i|\mathbf{x}, \mathbf{w}, \sigma^2 \sim N(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i), \sigma^2)$

prior $w|\alpha \sim N(0, \alpha^2)$

LINEAR REGRESSION AND REGULARIZATION : RIDGE REGRESSION

likelihood $y_i|\mathbf{x}, \mathbf{w}, \sigma^2 \sim N(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i), \sigma^2)$

prior $w|\alpha \sim N(0, \alpha^2)$

posterior: $p(\mathbf{w}|\mathbf{x}, \mathbf{y}, \sigma, \alpha) \propto \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-(y_i - \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i))^2/2\sigma^2\right) \prod_{j=0}^{M-1} \frac{1}{\sqrt{2\pi\alpha^2}} \exp(-w_j^2/2\alpha^2)$

LINEAR REGRESSION AND REGULARIZATION : RIDGE REGRESSION

likelihood $y_i|\mathbf{x}, \mathbf{w}, \sigma^2 \sim N(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i), \sigma^2)$

prior $w|\alpha \sim N(0, \alpha^2)$

posterior: $$p(\mathbf{w}|\mathbf{x}, \mathbf{y}, \sigma, \alpha) \propto \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-(y_i - \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i))^2/2\sigma^2\right) \prod_{j=0}^{M-1} \frac{1}{\sqrt{2\pi\alpha^2}} \exp(-w_j^2/2\alpha^2)$$

$$\log p(\mathbf{w}|\mathbf{x}, \mathbf{y}, \sigma, \alpha) = -\frac{1}{2\sigma^2} \sum_{n=1}^{N} (y_i - \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_i)) - \frac{1}{\alpha^2} \sum_{j=0}^{M-1} w_j^2 + C$$

LINEAR REGRESSION AND REGULARIZATION : RIDGE REGRESSION

likelihood $y_i|\mathbf{x}, \mathbf{w}, \sigma^2 \sim N(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i), \sigma^2)$

prior $w|\alpha \sim N(0, \alpha^2)$

posterior: $p(\mathbf{w}|\mathbf{x}, \mathbf{y}, \sigma, \alpha) \propto \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i))^2/2\sigma^2\right) \prod_{j=0}^{M-1} \frac{1}{\sqrt{2\pi\alpha^2}} \exp(-w_j^2/2\alpha^2)$

$$\log p(\mathbf{w}|\mathbf{x}, \mathbf{y}, \sigma, \alpha) = -\frac{1}{2\sigma^2}\sum_{n=1}^{N}(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i)) - \frac{1}{\alpha^2}\sum_{j=0}^{M-1} w_j^2 + C$$

Find MAP (=maximum a posteriori) estimate for **w** by minimizing:

$$\sum_{n=1}^{N}(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i)) + \frac{\sigma^2}{\alpha^2}\mathbf{w}^T\mathbf{w}$$

Minimize:

$$\sum_{n=1}^{N}(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i)) + \lambda||\mathbf{w}||^2$$

LINEAR REGRESSION AND REGULARIZATION : RIDGE REGRESSION

Minimize:

$$\sum_{n=1}^{N}(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i)) + \lambda||\mathbf{w}||^2$$

Regularization involves adding additional information to prevent overfitting.

A common form it can take is adding a penalty term to the loss function.

LINEAR REGRESSION AND REGULARIZATION: LASSO

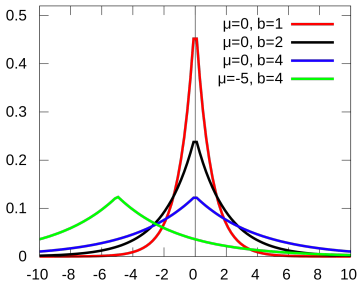
$$\sum_{n=1}^{N}(y_i - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_i)) + \lambda||\mathbf{w}||_1^2$$

$$||\mathbf{w}||_1 := \sum |w_j|$$

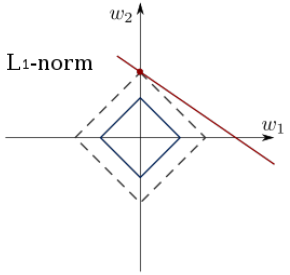
Can derive from putting a Laplacian prior over **w**

Can derive from putting a Laplacian prior over **w**
Leads to **sparse** solutions for **w** (i.e. many components of w are zero).

Laplace distribution

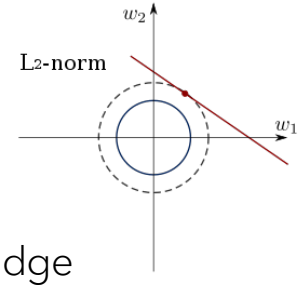


$|w_1| + |w_2| = const$



lasso

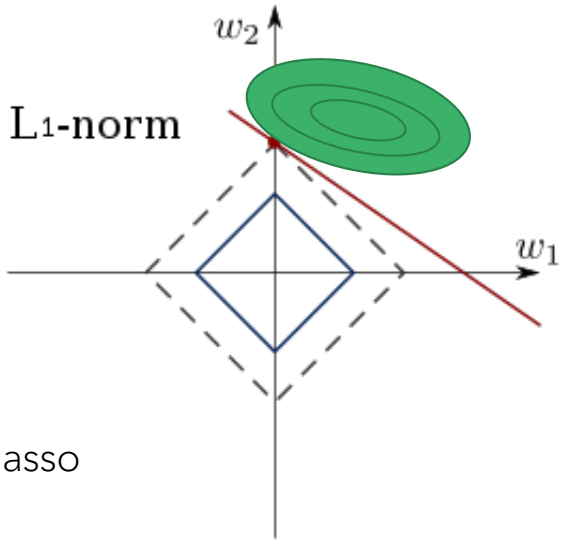
$w_1^2 + w_2^2 = const$



ridge

diamond

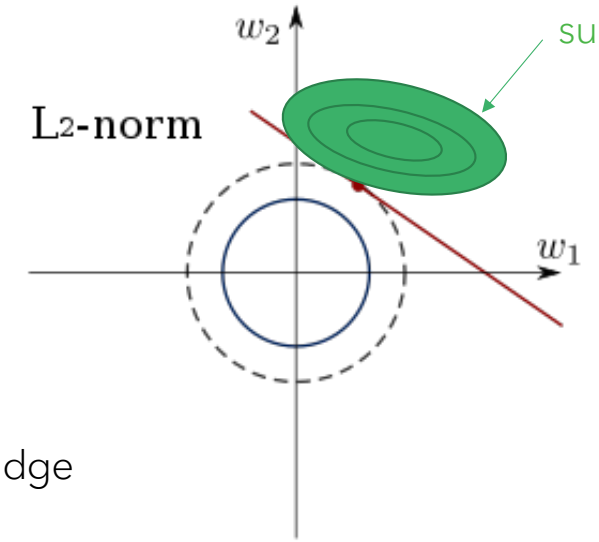
$$|w_1| + |w_2| = const$$



lasso

circle

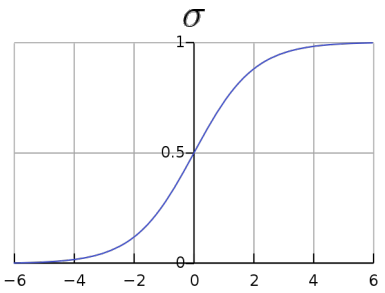
$$w_1^2 + w_2^2 = const$$



ridge

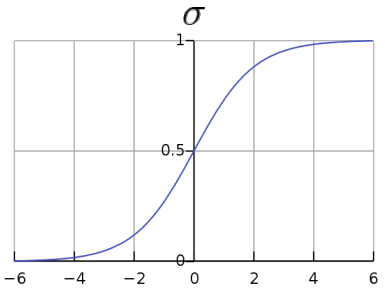
LOGISTIC REGRESSION FOR CLASSIFICATION

$$p(C_0|\phi) = y(\phi) = \sigma(\mathbf{w}^T\phi) = \frac{1}{1+e^{-\mathbf{w}^T\phi}}$$



LOGISTIC REGRESSION FOR CLASSIFICATION

$$p(C_0|\phi) = y(\phi) = \sigma(\mathbf{w}^T\phi) = \frac{1}{1+e^{-\mathbf{w}^T\phi}}$$



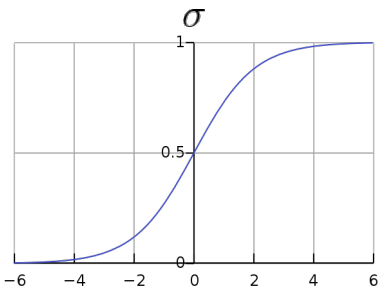
Data: t =0 or 1, class 0 or class 1

$$p(t_i) = y_i^{t_i}(1-y_i)^{1-t_i}$$

$$p(\mathbf{t}|\mathbf{w}) = \prod_{i=1}^{N} y_i^{t_i}(1-y_i)^{1-t_i}$$

LOGISTIC REGRESSION FOR CLASSIFICATION

$$p(C_0|\phi) = y(\phi) = \sigma(\mathbf{w}^T\phi) = \frac{1}{1+e^{-\mathbf{w}^T\phi}}$$



Data: t =0 or 1, class 0 or class 1

$$p(t_i) = y_i^{t_i}(1-y_i)^{1-t_i}$$

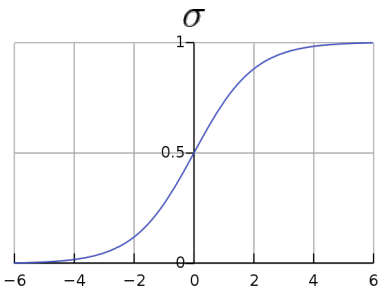
$$p(\mathbf{t}|\mathbf{w}) = \prod_{i=1}^{N} y_i^{t_i}(1-y_i)^{1-t_i}$$

error function, which has a nice gradient: $\nabla E(\mathbf{w}) = \sum_{i=1}^{N}(y_n - t_n)\phi_n$

$$-\log p(\mathbf{t}|\mathbf{w}) = -\sum_{i=1}^{N} t_i \log y_i + (1-t_i)\log(1-y_i)$$

LOGISTIC REGRESSION FOR CLASSIFICATION

$$p(C_0|\phi) = y(\phi) = \sigma(\mathbf{w}^T\phi) = \frac{1}{1+e^{-\mathbf{w}^T\phi}}$$



Data: t =0 or 1, class 0 or class 1

$$p(t_i) = y_i^{t_i}(1-y_i)^{1-t_i}$$

$$p(\mathbf{t}|\mathbf{w}) = \prod_{i=1}^{N} y_i^{t_i}(1-y_i)^{1-t_i}$$

error function, which has a nice gradient: $\nabla E(\mathbf{w}) = \sum_{i=1}^{N}(y_n - t_n)\phi_n$

$$-\log p(\mathbf{t}|\mathbf{w}) = -\sum_{i=1}^{N} t_i \log y_i + (1-t_i)\log(1-y_i)$$

cross entropy

LOGISTIC REGRESSION FOR CLASSIFICATION

$$p(C_0|\phi) = y(\phi) = \sigma(\mathbf{w}^T\phi) = \frac{1}{1+e^{-\mathbf{w}^T\phi}}$$

Data: t =0 or 1, class 0 or class 1

$$p(t_i) = y_i^{t_i}(1-y_i)^{1-t_i}$$

$$p(\mathbf{t}|\mathbf{w}) = \prod_{i=1}^{N} y_i^{t_i}(1-y_i)^{1-t_i}$$

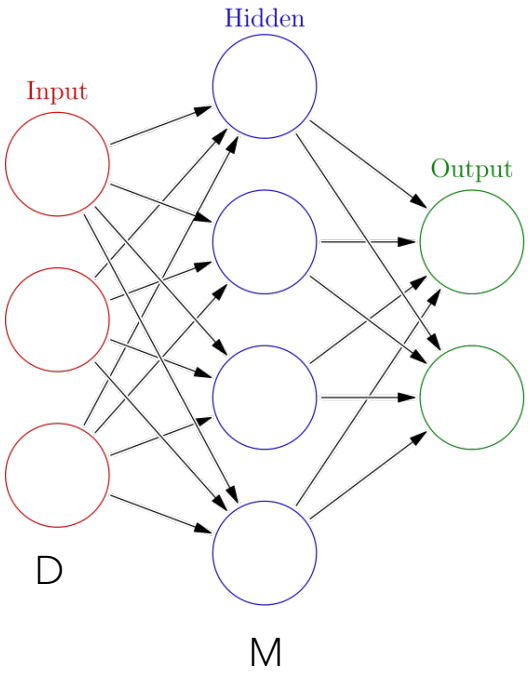
$$-\log p(\mathbf{t}|\mathbf{w}) = -\sum_{i=1}^{N} t_i \log y_i + (1-t_i)\log(1-y_i)$$

cross entropy

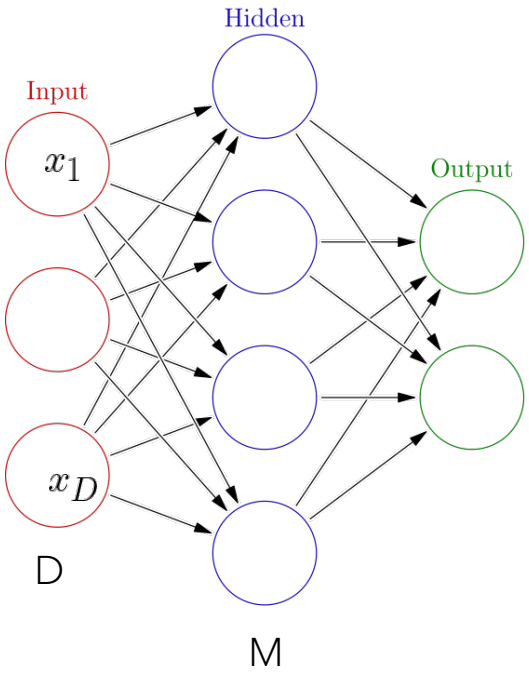
error function, which has a nice gradient: $\nabla E(\mathbf{w}) = \sum_{i=1}^{N}(y_n - t_n)\phi_n$

Note: we can regularize here too!
scikit-learn's logistic regression is regularized by default!

NEURAL NETWORKS



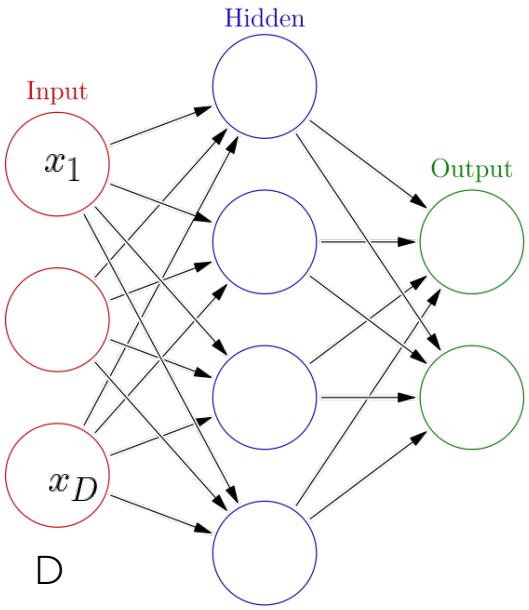
NEURAL NETWORKS



$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^2 h\left(\sum_{i=1}^{D} w_{ji}^1 x_i + w_{j0}^1\right) + w_{k0}^2\right)$$

hidden unit

NEURAL NETWORKS



$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^2 h\left(\sum_{i=1}^{D} w_{ji}^1 x_i + w_{j0}^1\right) + w_{k0}^2\right)$$

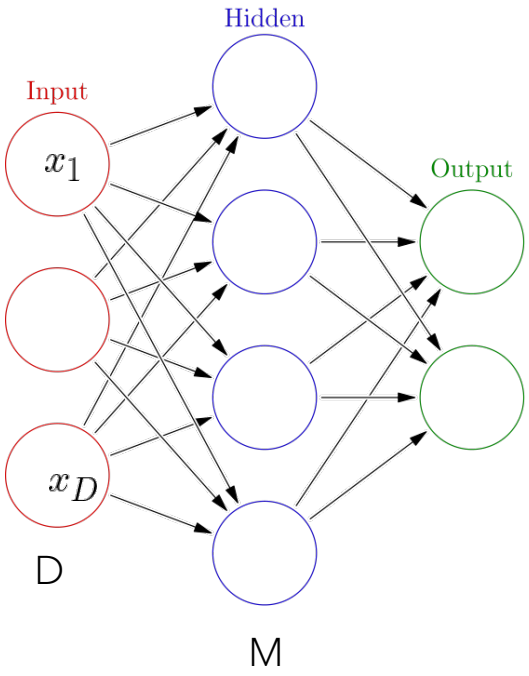
hidden unit

h some nonlinear function (e.g. logistic sigmoid, tanh, ReLU)
x values are inputs

For binary classification: can consider error function as cross-entropy

$$-\log p(\mathbf{t}|\mathbf{w}) = -\sum_{i=1}^{N} t_i \log y_i + (1 - t_i) \log(1 - y_i)$$

NEURAL NETWORKS



$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^2 h\left(\sum_{i=1}^{D} w_{ji}^1 x_i + w_{j0}^1\right) + w_{k0}^2\right)$$

hidden unit

h some nonlinear function (e.g. logistic sigmoid, tanh, ReLU)
x values are inputs

For binary classification: can consider error function as cross-entropy

$$-\log p(\mathbf{t}|\mathbf{w}) = -\sum_{i=1}^{N} t_i \log y_i + (1 - t_i) \log(1 - y_i)$$

Many weights w, error function with many local minima -> not as simple to fit!

DEEP LEARNING LOSS FUNCTIONS

A fascinating visualization project: https://losslandscape.com/

REFERENCE

Pattern Recognition and Machine Learning by Christopher Bishop, esp. Ch4-5:

https://www.microsoft.com/en-us/research/uploads/prod/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf

REFLECTION

What did you learn?

What's something that made sense to you?

What was your "muddiest point" from today?

*If you are comfortable, please share with me using Zoom's private chat feature

WHAT WE'VE COVERED SO FAR

- Interpretations of probability
- Random variables (discrete and continuous)
- Probability mass functions, probability density functions
- Expectation, variance, covariance
- Joint, conditional, marginal probabilities
- Independence
- Bayes theorem
- Applications of Bayes theorem for statistical learning – beta/binomial coin-flipping example, ridge reg. and lasso
- Priors, likelihoods, posteriors
- Maximum likelihood, maximum a posteriori (MAP estimates)

WHAT WE'VE COVERED SO FAR

- Linear regression: maximum likelihood approach
- Linear regression with regularization: ridge regression and lasso
- Loss/error functions
- Loss/error functions as derived from likelihood or posterior distributions
- Logistic regression for classification (model, error function, options for regularization)
- Neural network: model, error function
- Gradient descent and stochastic gradient descent

POSTERIOR DISTRIBUTIONS REVISTED: BEYOND THE MAXIMUM

- In our coin flipping example, we mostly looked at the maximum a posteriori (MAP) estimate of the bias of the coin
- BUT, we actually knew the full posterior distribution of the bias of the coin.

POSTERIOR DISTRIBUTIONS REVISTED: BEYOND THE MAXIMUM

- In our coin flipping example, we mostly looked at the maximum a posteriori (MAP) estimate of the bias of the coin
- BUT, we actually knew the full posterior distribution of the bias of the coin.
- What else might we be interested in here?

POSTERIOR DISTRIBUTIONS REVISTED: BEYOND THE MAXIMUM

- In our coin flipping example, we mostly looked at the maximum a posteriori (MAP) estimate of the bias of the coin
- BUT, we actually knew the full posterior distribution of the bias of the coin.
- What else might we be interested in here?
  - Mean?
  - Variance?
  - Some function of the coin-bias?

POSTERIOR DISTRIBUTIONS REVISTED: BEYOND THE MAXIMUM

The posterior distribution gives us lots of information beyond the maximum that we can use – e.g. posterior means, quantifications of uncertainty

If we can sample from the posterior distribution, we can compute lots of quantities of interest

Sometimes we have a posterior distribution that might not be nice at all, but from which we can still sample (e.g. using MCMC) – that's not a bad place to be!

REGRESSION, LASSO AND RIDGE, REGRESSION: SCALING?

What happens if I multiply a feature by a constant?

REGRESSION, LASSO AND RIDGE, REGRESSION: SCALING?

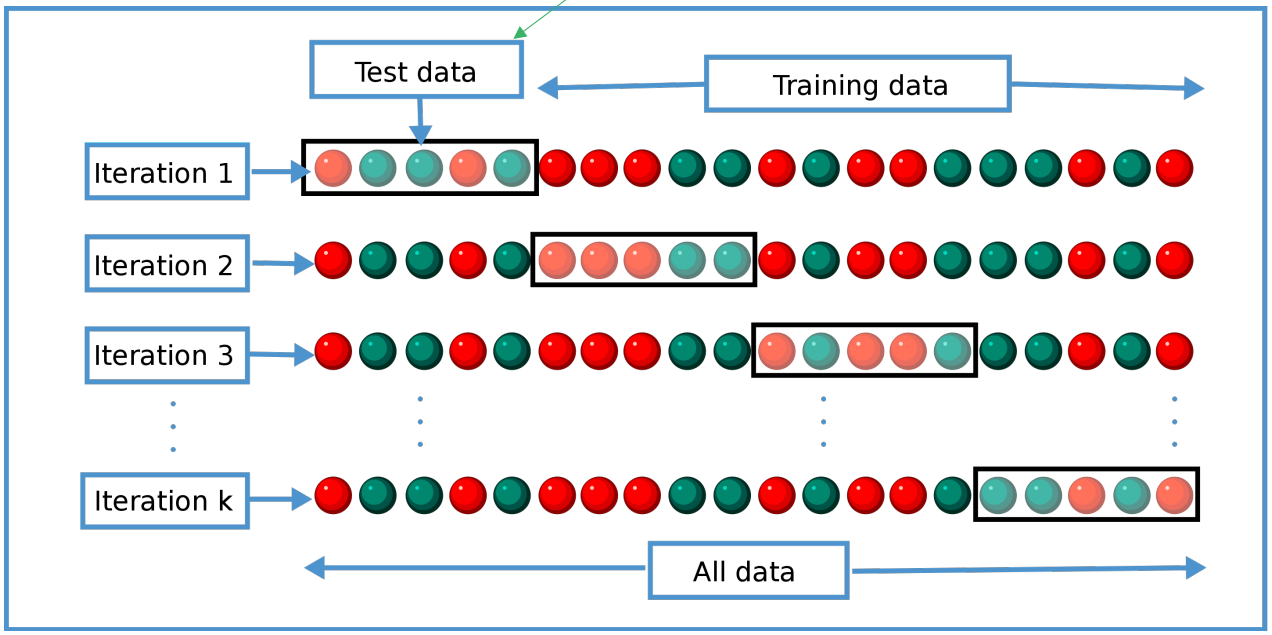
What happens if I multiply a feature by a constant?

Answer: model adjusts predictably for regression, but not so much for lasso and ridge!

Scale by subtract meaning and dividing by standard deviation before you apply!

Or, validation set

CROSS VALIDATION



Can use to tune model parameters

Want to hold out a separate set of test data to evaluate final model

BIAS-VARIANCE DECOMPOSITION

$$\text{expected loss} = \text{bias}^2 + \text{variance} + \text{noise}$$

bias – how does our prediction differ from an optimal prediction?
(high bias might mean a poor choice of model)

variance – how sensitive is our choice of y to a specific dataset?
(high variance means our model might be overly flexible, sensitive to particular dataset)

bias – how does our prediction differ from an optimal prediction?
variance – how sensitive is our choice of y to a specific dataset?

$$h(\mathbf{x}) = \mathbb{E}[t|\mathbf{x}] \text{ optimal prediction (for squared loss function)}$$

$$y(\mathbf{x}) \text{ model prediction}$$

$$\mathbb{E}_{data}[(y(\mathbf{x})-h(\mathbf{x}))^2] = [\mathbb{E}_{data}(y(\mathbf{x})-h(\mathbf{x}))]^2+\mathbb{E}_{data}[(y(\mathbf{x})-\mathbb{E}_{data}(y(\mathbf{x}))^2]$$

$$\begin{aligned}
\text{expected loss} &= \iint (y(\mathbf{x}) - t)^2 p(\mathbf{x}, t)\, d\mathbf{x} dt \\
&= \int (y(\mathbf{x}) - h(\mathbf{x}))^2\, p(\mathbf{x})\, d\mathbf{x} + \\
&\iint (h(\mathbf{x}) - t)^2 p(\mathbf{x}, t) d\mathbf{x} dt \\
&= \int [\mathbb{E}_{data}(y(\mathbf{x}) - h(\mathbf{x}))]^2 p(\mathbf{x})\, d\mathbf{x} + \int \mathbb{E}_{data}[(y(\mathbf{x}) - \mathbb{E}_{data}(y(\mathbf{x}))^2] p(\mathbf{x})\, d\mathbf{x} + \\
&\iint (h(\mathbf{x}) - t)^2 p(\mathbf{x}, \mathbf{t}) d\mathbf{x} dt
\end{aligned}$$

bias^2

variance

noise

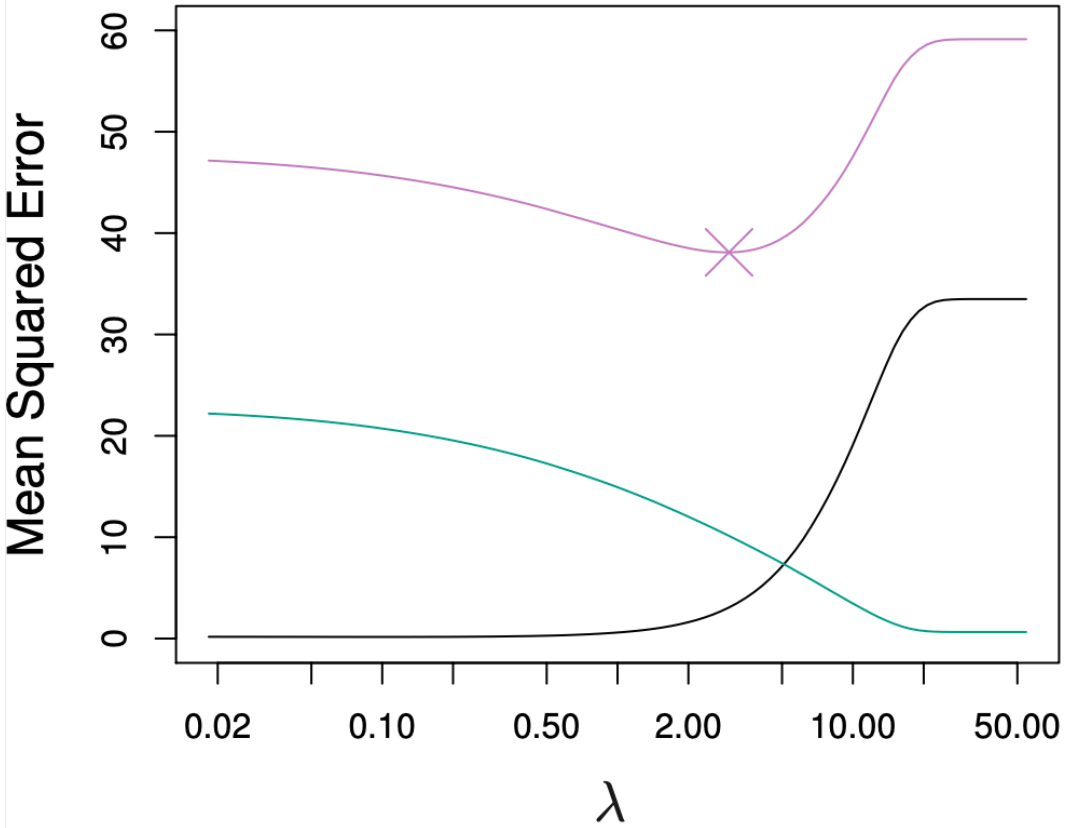


Figure from An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani "

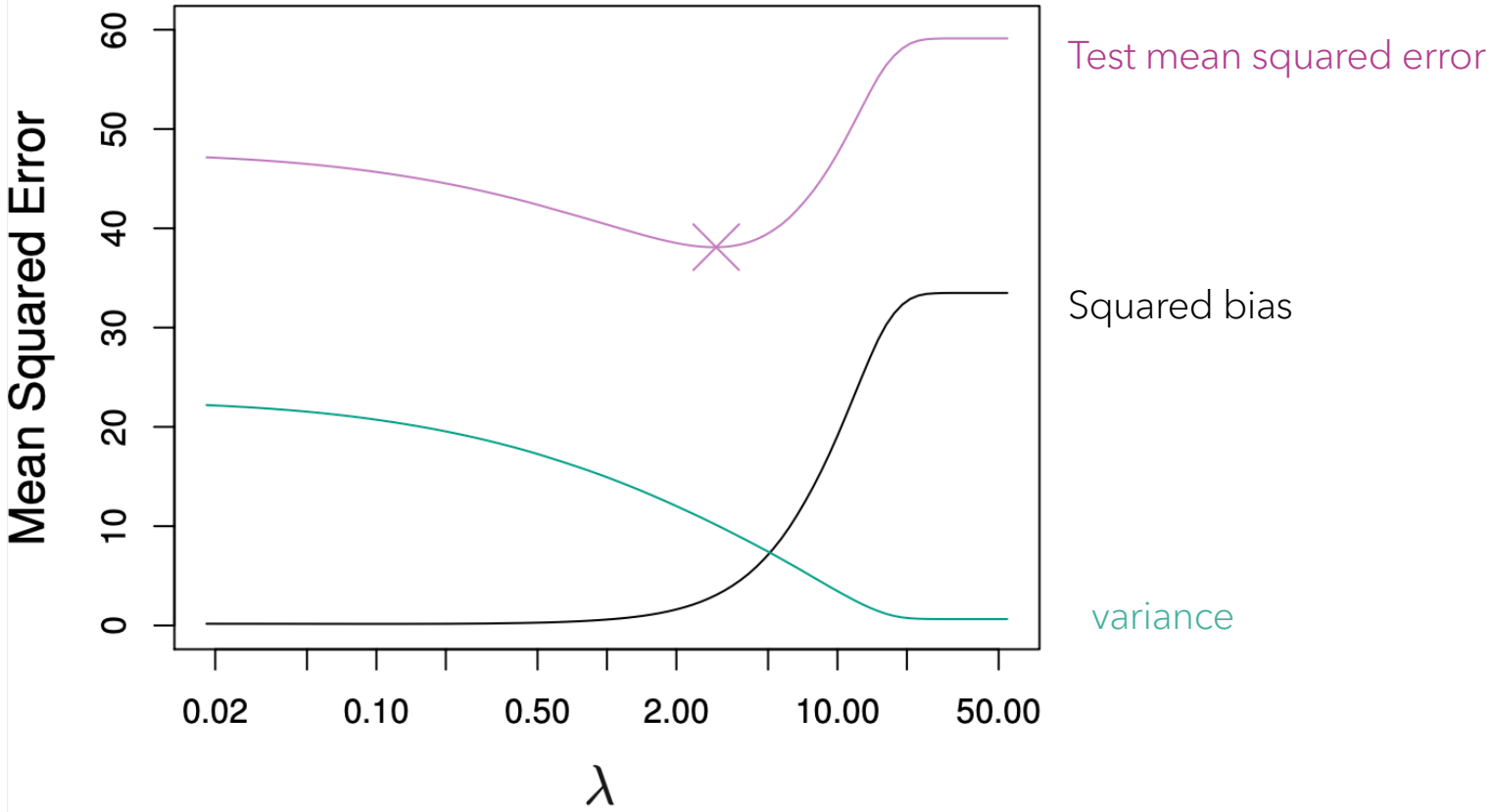
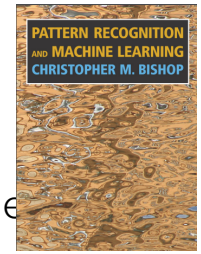


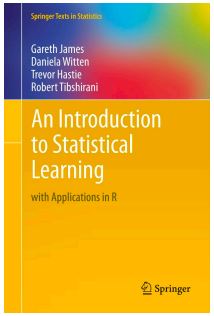
Figure from An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani "

REFERENCE

Pattern Recognition and Machine Learning by Christopher Bishop, esp. Ch3-5:



An Introduction to Statistical Learning with Applications in R by Gareth James, Danie
Witten, Trevor Hastie, Robert Tibshirani



SOME TYPES OF MACHINE LEARNING

Supervised learning – have inputs and corresponding targets

(May want to make predictions with new input data, or just understand relationship!)

- Regression: continuous target variables
- Classification: target variable is discrete category

SOME TYPES OF MACHINE LEARNING

Supervised learning – have inputs and corresponding targets

(May want to make predictions with new input data, or just understand relationship!)

- Regression: continuous target variables
- Classification: target variable is discrete category

Unsupervised learning – do not have target values

Typically want to discover structure in the data. Examples goals:

- Clustering
- Dimensionality reduction
- Density estimation

Reinforcement learning

PCA (PRINCIPAL COMPONENT ANALYSIS)

Data is high dimensional

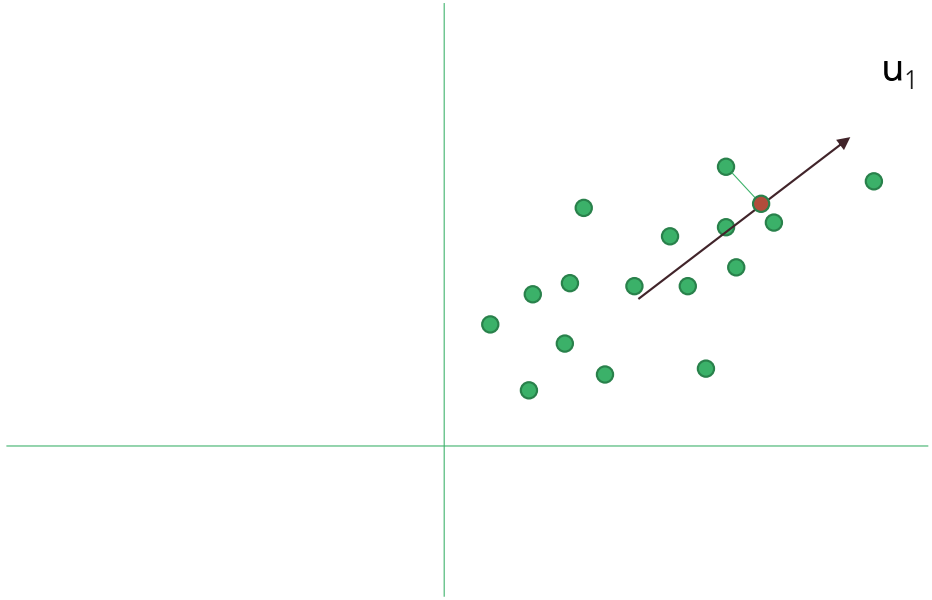
Want to project to a lower dimensional space but still keep relevant information

$$\{\mathbf{x}_n\} \in \mathbb{R}^D \rightarrow \{\tilde{\mathbf{x}}_n\} \in \mathbb{R}^M$$

M<D

PCA (PRINCIPAL COMPONENT ANALYSIS)

First, find u_1 (vector of length1) to maximize variance of the projection onto u_1

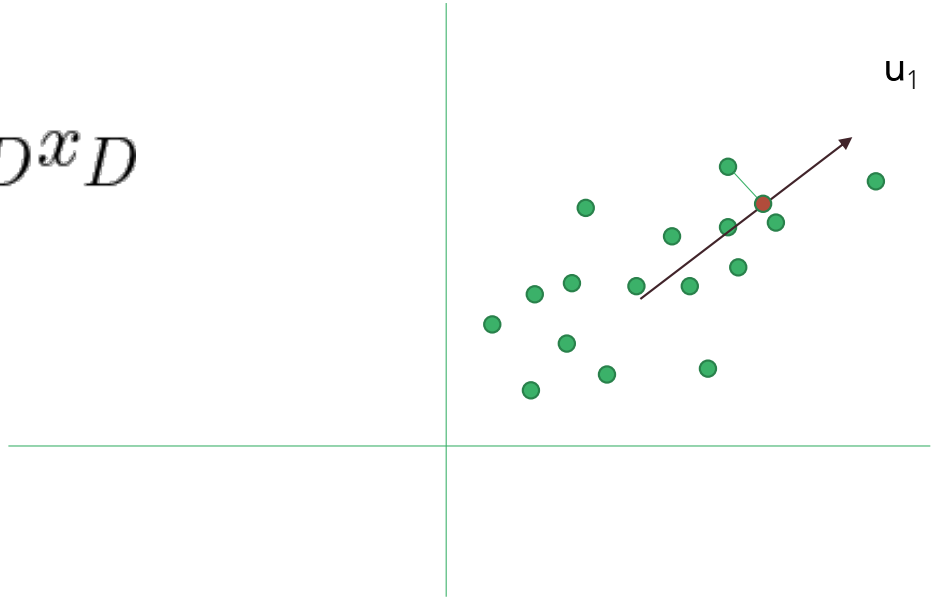


PCA (PRINCIPAL COMPONENT ANALYSIS)

First, find $\mathbf{u}_1$ (vector of length1) to minimize variance of the projection onto $\mathbf{u}_1$

Projection of $\mathbf{x}_n$ onto $\mathbf{u}_1$:

$$\mathbf{u}_1^T \mathbf{x}_n = u_1 x_1 + ... u_D x_D$$

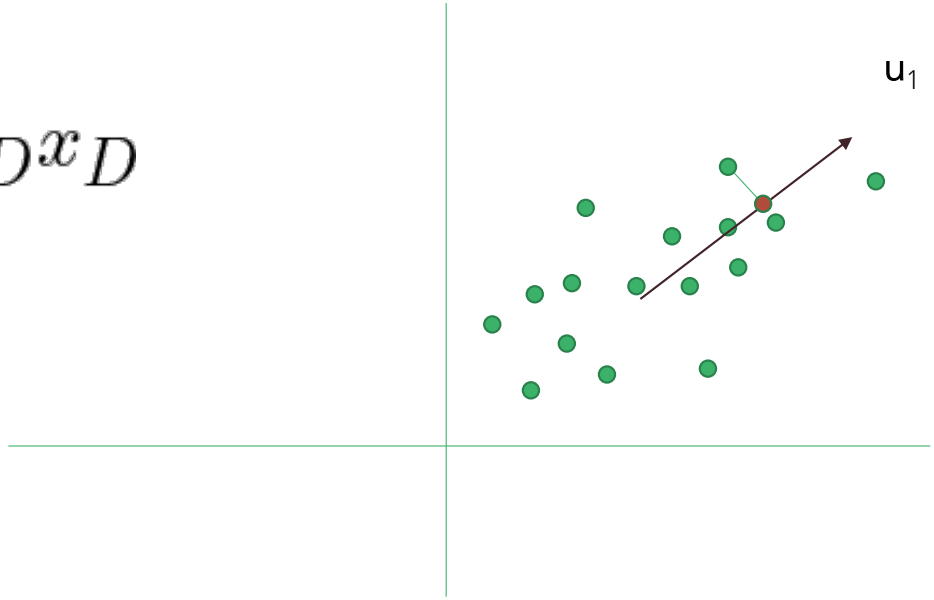


PCA (PRINCIPAL COMPONENT ANALYSIS)

First, find $\mathbf{u}_1$ (vector of length1) to minimize variance of the projection onto $\mathbf{u}_1$

Projection of $\mathbf{x}_n$ onto $\mathbf{u}_1$:

$$\mathbf{u}_1^T \mathbf{x}_n = u_1 x_1 + ... u_D x_D$$



PCA (PRINCIPAL COMPONENT ANALYSIS)

First, find u_1 (vector of length1) to minimize variance of the projection onto u_1

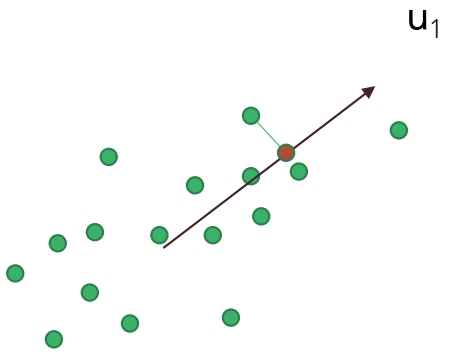
Projection of x_n onto u_1:

$$\mathbf{u}_1^T \mathbf{x}_n = u_1 x_1 + \ldots u_D x_D$$

variance of projection: $\frac{1}{N}\sum_{n=1}^{N}(\mathbf{u}_1^T\mathbf{x}_n - \mathbf{u}_1^T\bar{\mathbf{x}})^2 = \mathbf{u}_1^T\mathbf{S}\mathbf{u}_1$

where $\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T$, $\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$

data covariance



PCA (PRINCIPAL COMPONENT ANALYSIS)

First, find u_1 to maximize variance of the projection onto u_1

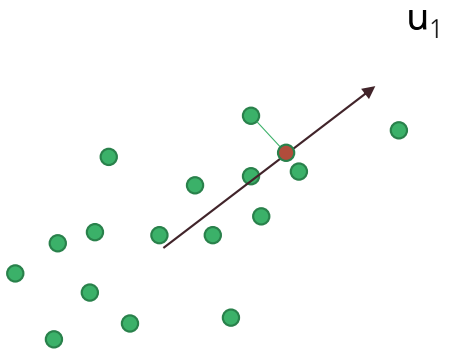
variance of projection: $\frac{1}{N}\sum_{n=1}^{N}(\mathbf{u}_1^T\mathbf{x}_n - \mathbf{u}_1^T\bar{\mathbf{x}})^2 = \mathbf{u}_1^T\mathbf{S}\mathbf{u}_1$

where $\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T$, $\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$

maximizing subject to $||\mathbf{u}_1|| = 1$, we get:

$\mathbf{S}\mathbf{u}_1 = \lambda_1\mathbf{u}_1$

$\mathbf{u}_1^T\mathbf{S}\mathbf{u}_1 = \lambda_1$



PCA (PRINCIPAL COMPONENT ANALYSIS)

First, find u_1 to maximize variance of the projection onto u_1

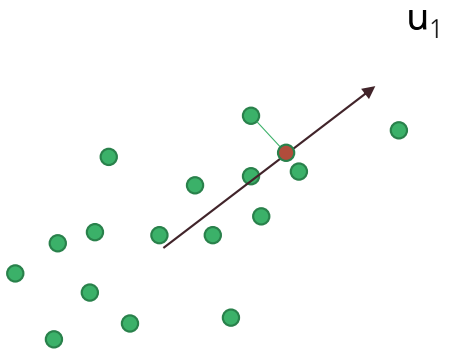
$$\text{variance of projection: } \frac{1}{N}\sum_{n=1}^{N}(\mathbf{u}_1^T\mathbf{x}_n - \mathbf{u}_1^T\bar{\mathbf{x}})^2 = \mathbf{u}_1^T\mathbf{S}\mathbf{u}_1$$

$$\text{where } \mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T,\ \bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

$$\text{maximizing subject to } ||\mathbf{u}_1|| = 1, \text{ we get:}$$

$$\mathbf{S}\mathbf{u}_1 = \lambda_1\mathbf{u}_1$$

$$\mathbf{u}_1^T\mathbf{S}\mathbf{u}_1 = \lambda_1$$



So choose u_1 to be unit-length eigenvector corresponding to largest eigenvalue of S

PCA (PRINCIPAL COMPONENT ANALYSIS)

Can imagine continuing iteratively, e.g. next choosing $\mathbf{u}_2$ to maximize the covariance of the projections of the $\mathbf{x}_n - \mathbf{u}_1^T\mathbf{x}_n$ onto $\mathbf{u}_2$

Stop at desired number of components M – can use variance explained to decide

PROBABALISTIC PCA

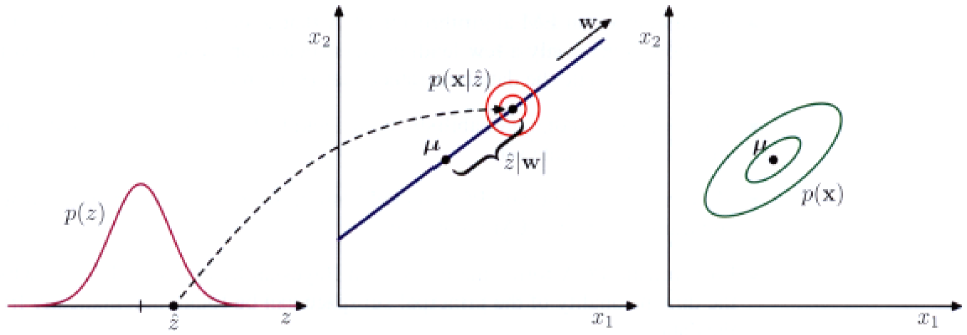


Figure 12.9 from Pattern Recognition and Machine Learning, by Christopher Bishop

PROBABALISTIC PCA

$$p(\mathbf{z}) \sim N(0, \mathbf{I})$$
$$p(\mathbf{x}|\mathbf{z}) = N(\mathbf{Wz} + \mathbf{mu}, \sigma^2\mathbf{I})$$
$$\implies p(\mathbf{x}) = N(\mathbf{x}, \mu, \mathbf{WW^T} + \sigma^2\mathbf{I}),$$
$$\text{can compute } p(\mathbf{z}|\mathbf{x})$$

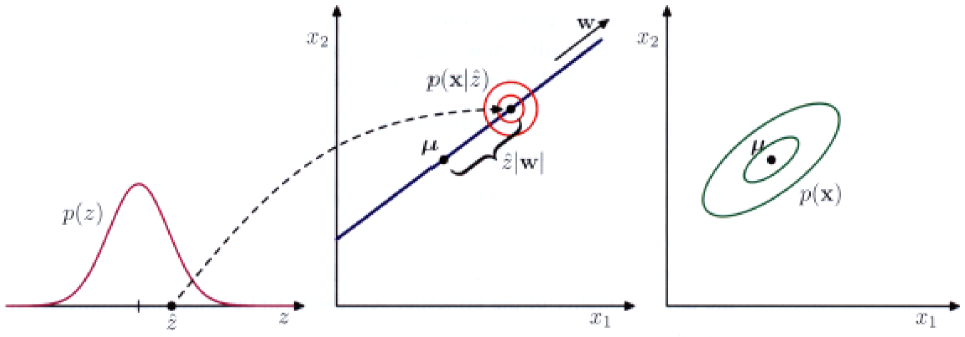


Figure 12.9 from Pattern Recognition and Machine Learning, by Christopher Bishop

z are latent variables

PROBABALISTIC PCA

Can also put a prior over W!

$$p(\mathbf{z}) \sim N(0, \mathbf{I})$$
$$p(\mathbf{x}|\mathbf{z}) = N(\mathbf{Wz} + \mathbf{mu}, \sigma^2\mathbf{I})$$
$$\implies p(\mathbf{x}) = N(\mathbf{x}, \mu, \mathbf{WW^T} + \sigma^2\mathbf{I}),$$
$$\text{can compute } p(\mathbf{z}|\mathbf{x})$$

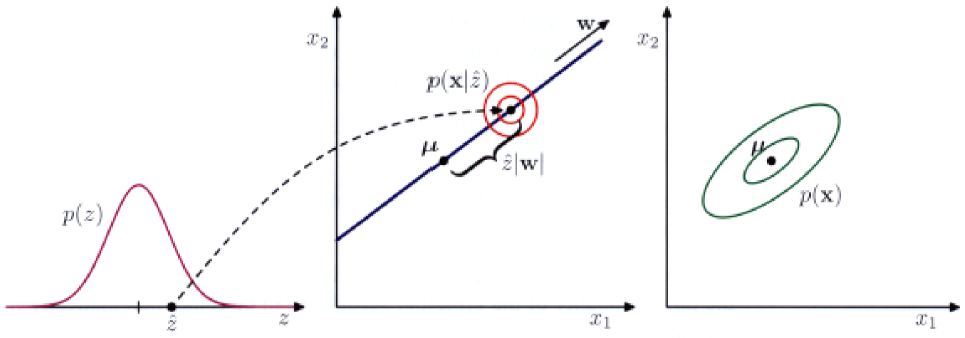


Figure 12.9 from Pattern Recognition and Machine Learning, by Christopher Bishop

z are latent variables

T-DISTRIBUTED STOCHASTIC NEIGHBOR EMBEDDING (T-SNE)

Again want to represent high dimensional vectors x with low dimensional counterparts y

Aim to transform close points into points that are still close

T-DISTRIBUTED STOCHASTIC NEIGHBOR EMBEDDING (T-SNE)

Again want to represent high dimensional vectors x with low dimensional counterparts y

Aim to transform close points into points that are still close

$$p_{j|i} = \frac{\exp(-||\mathbf{x}_i - \mathbf{x}_j||/\sigma_i^2)}{\sum_{k \neq i} \exp(-||\mathbf{x}_i - \mathbf{x}_k||/\sigma_i^2)}$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

T-DISTRIBUTED STOCHASTIC NEIGHBOR EMBEDDING (T-SNE)

Again want to represent high dimensional vectors x with low dimensional counterparts y

Aim to transform close points into points that are still close

$$p_{j|i} = \frac{\exp(-||\mathbf{x}_i - \mathbf{x}_j||/\sigma_i^2)}{\sum_{k \neq i} \exp(-||\mathbf{x}_i - \mathbf{x}_k||/\sigma_i^2)}$$

$$q_{ij} = \frac{(1 + ||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}}{\sum_{k \neq i} (1 + ||\mathbf{y}_i - \mathbf{y}_k||^2)^{-1}}$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

T-DISTRIBUTED STOCHASTIC NEIGHBOR EMBEDDING (T-SNE)

Again want to represent high dimensional vectors x with low dimensional counterparts y

Aim to transform close points into points that are still close

$$p_{j|i} = \frac{\exp(-||\mathbf{x}_i - \mathbf{x}_j||/\sigma_i^2)}{\sum_{k \neq i} \exp(-||\mathbf{x}_i - \mathbf{x}_k||/\sigma_i^2)} \qquad q_{ij} = \frac{(1 + ||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}}{\sum_{k \neq i}(1 + ||\mathbf{y}_i - \mathbf{y}_k||^2)^{-1}}$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

$$\text{minimize } KL(p,q) = \int p_{ij} \log(\frac{p_{ij}}{q_{ij}}) = \int p_{ij}(\log(p_{ij}) - \log(q_{ij}))$$

KL DIVERGENCE

$$\text{minimize } KL(p,q) = \int p_{ij} \log(\frac{p_{ij}}{q_{ij}}) = \int p_{ij}(\log(p_{ij}) - \log(q_{ij}))$$

A way to measure the "distance" two probability distributions, but caution….

Not symmetric! KL(p, q) and KL(q, p) are different in general

An intro blog post on KL divergence: https://karinknudson.com/kl_divergence.html

Useful in variational inference, among other settings

For an excellent discussion of t-SNE, see:

https://distill.pub/2016/misread-tsne/

Check out alternative **manifold learning** techniques – e.g. UMAP! For a brief introduction to those that are implemented in sci-kit learn, see: https://scikit-learn.org/stable/modules/manifold.html#manifol

For a discussion and comparison of t-SNE and PCA (with examples in Python), see e.g.:

https://towardsdatascience.com/an-introduction-to-t-sne-with-python-example-5a3a293108d1

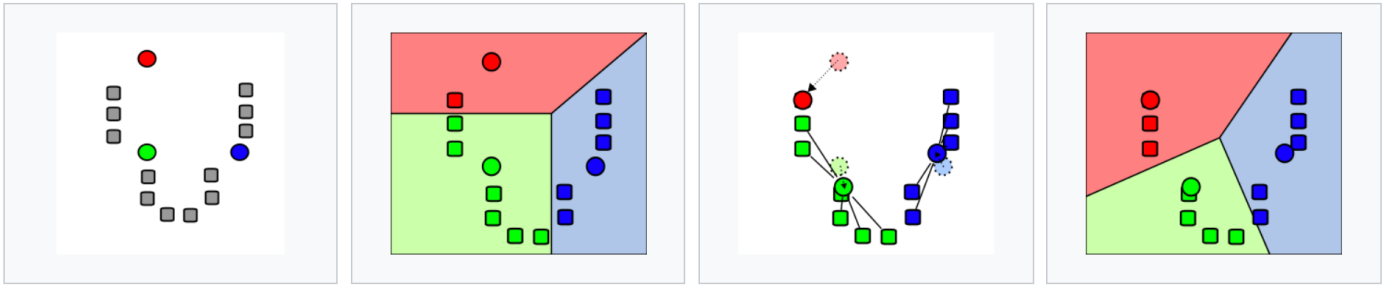
K-MEANS CLUSTERING

Choose the number of clusters you expect: K

Choose a set of K points ("means") then alternately

1. "Assign" each data point to the cluster corresponding to the nearest "mean"

2. Recalculate the means by taking the average of the points assigned to each cluster

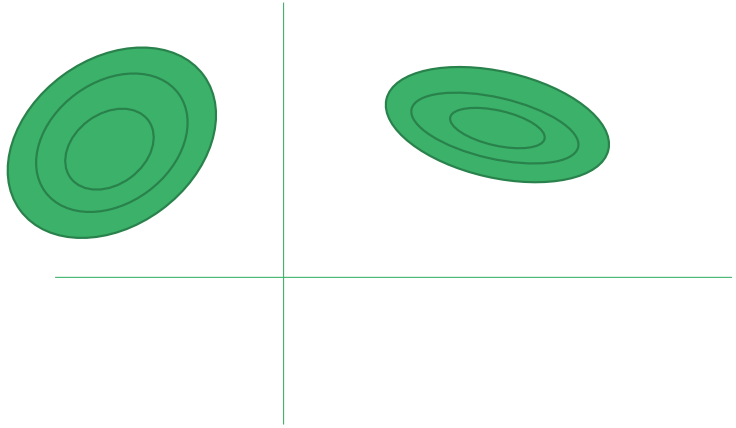


MIXTURE OF GAUSSIANS

Assume data points are generated from a mixture of K Gaussian distributions

Find the mean and variance of those Gaussians that give the highest likelihood

(Can use EM algorithm to optimize the likelihood)



$\pi_k = \text{prob a point is in class k}$

$\mathbf{x}_i | \pi_k, \mu_k, \Sigma_k \sim N(\mu_k, \Sigma_k)$

TODAY: SEQUENCES

MARKOV CHAINS (DISCRETE-TIME)

Sequence of random variables satisfying a “memorylessness” property: next value depends only on current value

MARKOV CHAINS (DISCRETE-TIME)

Sequence of random variables satisfying a "memorylessness" property: next value depends only on current value

$X_1, X_2, X_3, ...$ $\qquad P(X_n|X_1, X_2, ..X_{n-1}) = P(X_n|X_{n-1})$

MARKOV CHAINS (DISCRETE-TIME)

Sequence of random variables satisfying a "memorylessness" property: next value depends only on current value

$$X_1, X_2, X_3, \ldots \qquad P(X_n|X_1, X_2, ..X_{n-1}) = P(X_n|X_{n-1})$$

If the random variables have finite support (i.e. the **state space** is finite), and the probabilities don't depend on n (time-homogeneous Markov chain), then can specify with:

$$p_{ij} = P(X_n = j|X_{n-1} = i)$$

MARKOV CHAINS (DISCRETE-TIME)

Sequence of random variables satisfying a "memorylessness" property: next value depends only on current value

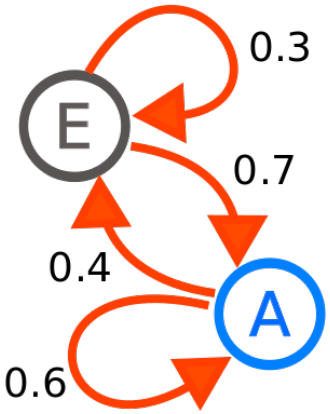
$$X_1, X_2, X_3, \ldots \qquad P(X_n|X_1, X_2, ..X_{n-1}) = P(X_n|X_{n-1})$$

If the random variables have finite support (i.e. the **state space** is finite), and the probabilities don't depend on n (time-homogeneous Markov chain), then can specify with:

gives **transition matrix** → $p_{ij} = P(X_n = j|X_{n-1} = i)$

Can extend to a Markov chain of order k, where state depends on previous k states.

MARKOV CHAINS (DISCRETE-TIME)

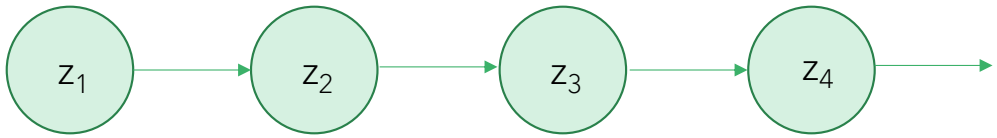


Transition matrix P

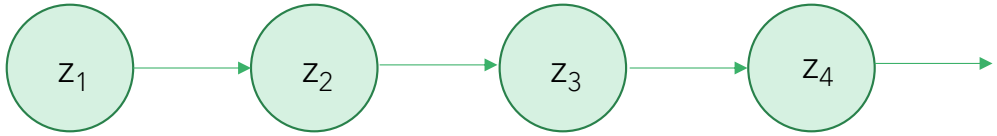
	E	A
E	.3	.7
A	.4	.6

A stationary distribution x satisfies xP = x

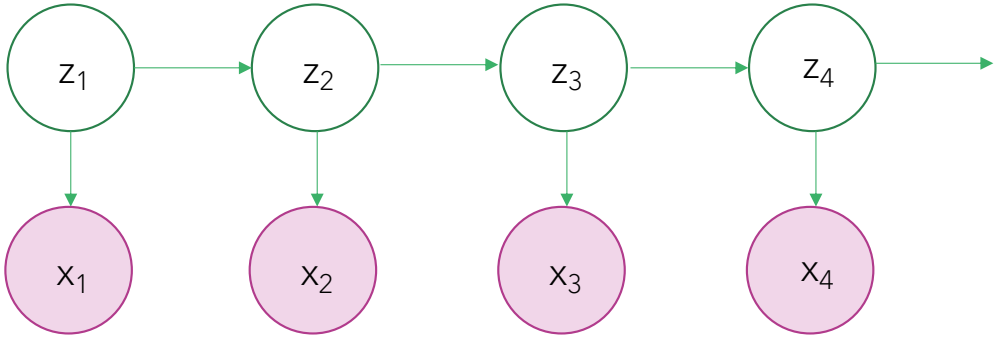
MARKOV CHAIN



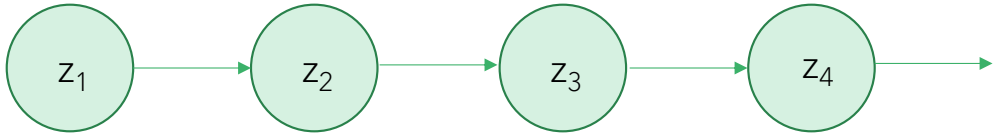
MARKOV CHAIN



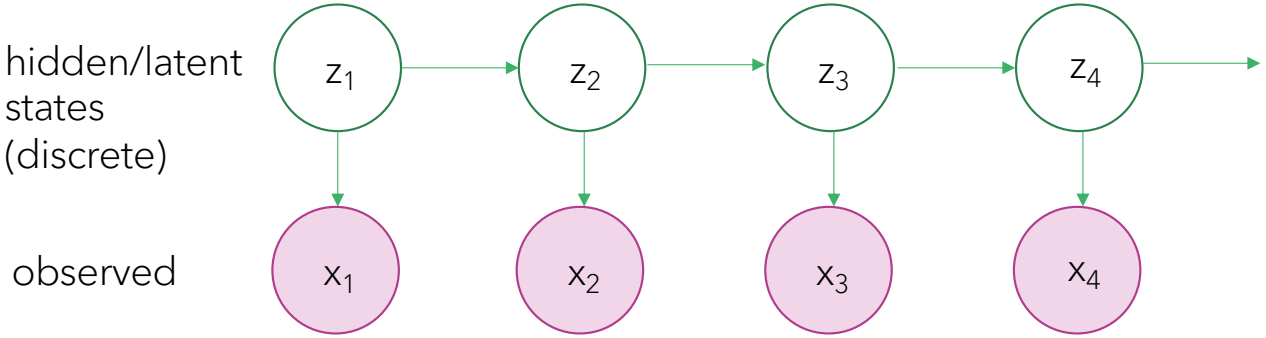
HIDDEN MARKOV MODELS (HMM)



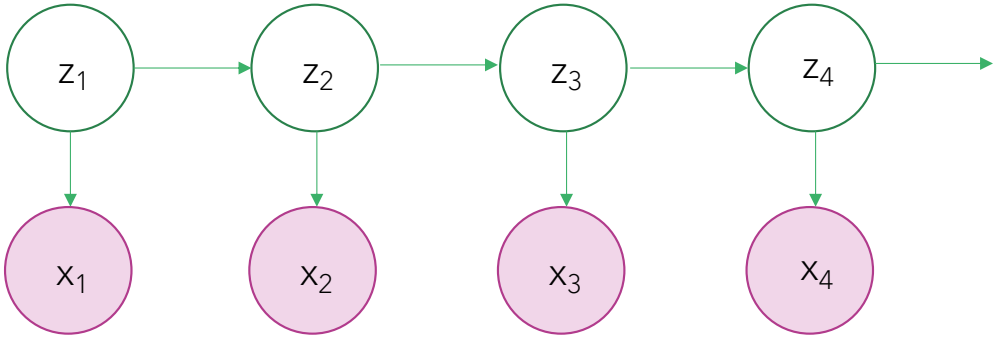
MARKOV CHAIN



HIDDEN MARKOV MODELS (HMM)



HIDDEN MARKOV MODELS (HMM)

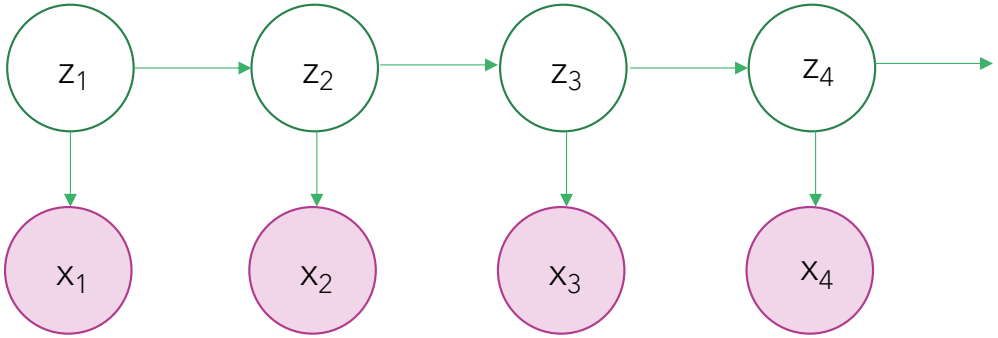


Output probabilities/emission probabilities: $p(\mathbf{x_n}|\mathbf{z}_n, \phi)$

Transition probabilities given by matrix: $\mathbf{A}$

Distribution of first hidden state: $\boldsymbol{\pi}$

HIDDEN MARKOV MODELS (HMM)



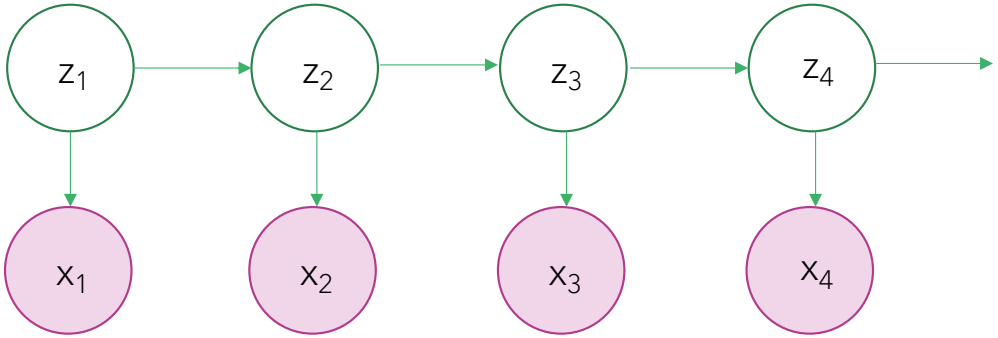
Output probabilities/emission probabilities: $p(\mathbf{x_n}|\mathbf{z}_n, \phi)$

$$p(\mathbf{x}_n|\mathbf{z}_n, \phi) = \prod_{k=1}^{K} p(\mathbf{x}_n|\phi_k)^{z_{nk}} \text{ (z gives one-hot encoding of state)}$$

Transition probabilities given by matrix: $\mathbf{A}$

Distribution of first hidden state: $\boldsymbol{\pi}$

HIDDEN MARKOV MODELS (HMM)



Output probabilities/emission probabilities: $p(\mathbf{x_n}|\mathbf{z}_n, \phi)$

$$p(\mathbf{x}_n|\mathbf{z}_n, \phi) = \prod_{k=1}^{K} p(\mathbf{x}_n|\phi_k)^{z_{nk}} \text{ (z gives one-hot encoding of state)}$$

Transition probabilities given by matrix: $\mathbf{A}$

Distribution of first hidden state: $\boldsymbol{\pi}$

HIDDEN MARKOV MODELS (HMM)

Output probabilities/emission probabilities: $p(\mathbf{x_n}|\mathbf{z}_n, \phi)$

$$p(\mathbf{x}_n|\mathbf{z}_n, \phi) = \prod_{k=1}^{K} p(\mathbf{x}_n|\phi_k)^{z_{nk}} \text{ (z gives one-hot encoding of state)}$$

Transition probabilities given by matrix: $\mathbf{A}$

Distribution of first hidden state: $\boldsymbol{\pi}$

HIDDEN MARKOV MODELS (HMM)

Output probabilities/emission probabilities: $p(\mathbf{x_n}|\mathbf{z}_n, \phi)$

$$p(\mathbf{x}_n|\mathbf{z}_n, \phi) = \prod_{k=1}^{K} p(\mathbf{x}_n|\phi_k)^{z_{nk}} \text{ (z gives one-hot encoding of state)}$$

Transition probabilities given by matrix: $\mathbf{A}$

Distribution of first hidden state: $\boldsymbol{\pi}$

Learning parameters: can use expectation maximization (EM) algorithm – useful for optimizing a posterior or likelihood when we have latent variables. Iteratively:

Find $P(Z|X, \theta^{old})$

Compute a new θ to maximize the expectation under $P(Z|X, \theta^{old})$ of the log likelihood of $P(Z, X|\theta)$

HIDDEN MARKOV MODELS (HMM)

Output probabilities/emission probabilities: $p(\mathbf{x_n}|\mathbf{z}_n, \phi)$

$$p(\mathbf{x}_n|\mathbf{z}_n, \phi) = \prod_{k=1}^{K} p(\mathbf{x}_n|\phi_k)^{z_{nk}} \text{ (z gives one-hot encoding of state)}$$

Transition probabilities given by matrix: $\mathbf{A}$

Distribution of first hidden state: $\boldsymbol{\pi}$

Learning parameters: expectation maximization (EM) algorithm – useful for optimizing a posterior or likelihood when we have latent variables. Iteratively:

Find $P(Z|X, \theta^{old})$

Compute a new θ to maximize the expectation under $P(Z|X, \theta^{old})$ of the log likelihood of $P(Z, X|\theta)$

Viterbi algorithm for finding most likely sequence of latent states

HIDDEN MARKOV MODELS (HMM)

Can put priors over quantities of interest

If we had a two state system, what form might the prior of a set of transition probabilities take?

For prior over transition probabilities with more than two states, could use Dirichlet prior (generalizes the beta distribution)

If we don't know the number of states, could use a Dirichlet process

Final point: note connection to mixture models!

HIDDEN MARKOV MODELS (HMM)

Can put priors over quantities of interest

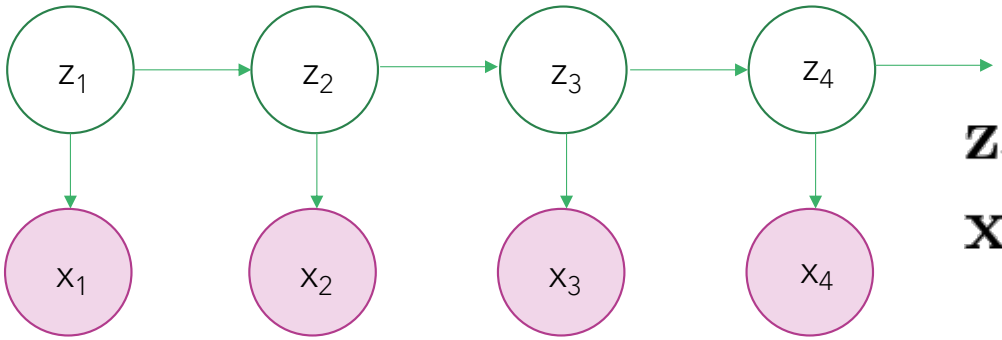
If we had a two state system, what form might the prior of a set of transition probabilities take?

For prior over transition probabilities with more than two states, could use Dirichlet prior (generalizes the beta distribution)

If we don't know the number of states, could use a Dirichlet process

Final point: note connection to mixture models!

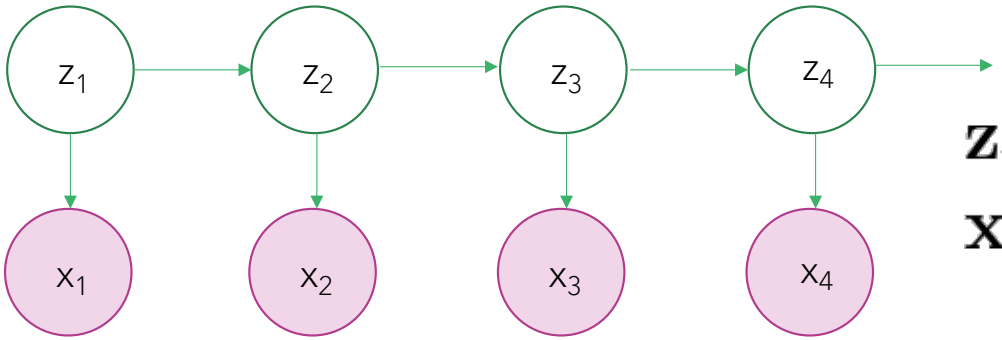
RELATED: LINEAR DYNAMICAL SYSTEMS



$$\mathbf{z}_n|\mathbf{z}_{n-1} \sim N(A\mathbf{z}_{n-1}, \Lambda)$$
$$\mathbf{x}_n|\mathbf{z}_n \sim N(C\mathbf{z}_n, \Sigma)$$

Similar structure with latent and observed variables, but latent and observed variables are Gaussian and have linear relationship

RELATED: LINEAR DYNAMICAL SYSTEMS



$$\mathbf{z}_n|\mathbf{z}_{n-1} \sim N(A\mathbf{z}_{n-1}, \Lambda)$$
$$\mathbf{x}_n|\mathbf{z}_n \sim N(C\mathbf{z}_n, \Sigma)$$

Similar structure with latent and observed variables, but latent and observed variables are Gaussian and have linear relationship

If parameters are known, Kalman filtering. If unknown, can do inference using, e.g. EM

MARKOV CHAIN MONTE CARLO (MCMC)

Recall that even if we can't compute a distribution, having a representative set of samples gives us a great deal of information!

MCMC gives a framework for sampling from many distributions

MCMC

Want to sample from a distribution of interest p*

Will do so by trying to find a Markov chain that is invariant to that distribution, meaning a step in the chain does not change the distribution:

$$p^*(\mathbf{x}) = \sum_{\mathbf{x}'} T(\mathbf{x}', \mathbf{x}) p^*(\mathbf{x}')$$

MCMC

Want to sample from a distribution of interest p*

Will do so by trying to find a Markov chain that is **invariant to that distribution**, meaning a step in the chain does not change the distribution:

$$p^*(\mathbf{x}) = \sum_{\mathbf{x}'} T(\mathbf{x}', \mathbf{x}) p^*(\mathbf{x}')$$

If transition probabilities satisfy **detailed balance**:

$$p^*(\mathbf{x}) T(\mathbf{x}, \mathbf{x}') = p^*(\mathbf{x}') T(\mathbf{x}', \mathbf{x})$$

then p* will be invariant

MCMC

So, if we to define a Markov chain that generates samples from the distribution of interest – e.g. if we can show that detailed balance is satisfied

MCMC

So, if we to define a Markov chain that generates samples from the distribution of interest – e.g. if we can show that detailed balance is satisfied

Metropolis-Hastings algorithm:

$$\text{Propose } \mathbf{x}' \text{ from } q_k(\mathbf{x}|\mathbf{x}^{prev})$$

$$\text{Accept with probability } \min\left(1, \frac{\tilde{p}(\mathbf{x}')q_k(\mathbf{x}^{prev}|\mathbf{x}')}{\tilde{p}(\mathbf{x}^{prev})q_k(\mathbf{x}'|\mathbf{x}^{prev})}\right)$$

$$\tilde{p}(\mathbf{x}) = p^*(\mathbf{x})/Z$$

MCMC

Special case of Metropolis-Hastings: Gibbs sampling

(iteratively sample each component of the vector whose distribution we'd like to know, conditioned on all the other components)

MCMC algorithms that make use of gradient information:

Hamiltonian Monte Carlo, No U-Turn Sampler

RESOURCES

Pattern Recognition and Machine Learning by Christopher Bishop

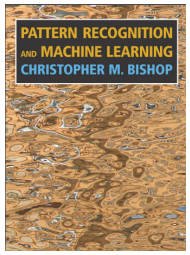
An Introduction to Statistical Learning with Applications in R by Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani

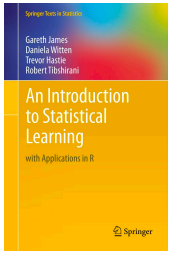
The Elements of Statistical Learning, by Trevor Hastie, Robert Tibshirani, Jerome Friedman

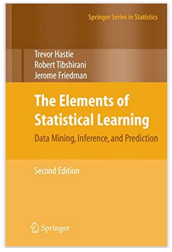
Statistical Rethinking Richard McElreath

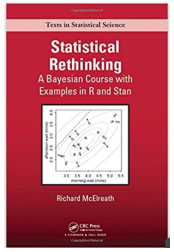
Machine Learning: A Probabilistic Perspective, by Kevin Murphy

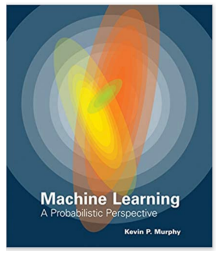
Bayesian Data Analysis, by Gelman et al.











THANK YOU!