Machine Learning Principles

CoE197M/EE298M (Foundations of Machine Learning)

Rowel Atienza, Ph.D.

rowel@eee.upd.edu.ph

Reference: "Mathematics for Machine Learning". Copyright 2020 by Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong. Published by Cambridge University Press.

Empirical Risk Minimization

Learning – estimate the model parameters using training data

Given (Supervised Learning): N examples: $\boldsymbol{x}_n \in \mathbb{R}^D$ with corresponding labels $y_n \in \mathbb{R}$

Goal: Estimate the predictor $f(\cdot, \boldsymbol{\theta}) : \mathbb{R}^D \rightarrow \mathbb{R}$

Problem: Find the optimal parameters $\boldsymbol{\theta}^*$ that fit the data well:

$$f(\boldsymbol{x}_n, \boldsymbol{\theta}^*) \approx y_n \quad n = 1 \ldots N$$

Prediction: $\hat{y}_n = f(\boldsymbol{x}_n, \boldsymbol{\theta}^*)$

Loss Function

Ground truth: y_n

Prediction: $\hat{y}_n$

Loss Function: $L = l(y_n, \hat{y}_n)$

Machine Learning: Use dataset, $\mathcal{D}_{train} = \{(\boldsymbol{x}_n, y_n)\}, \quad n = 1 \dots N$ to estimate the parameters $\boldsymbol{\theta}$ by minimizing $L = l(y_n, \hat{y}_n)$

Assumption: $(\boldsymbol{x}_1, y_1) \dots (\boldsymbol{x}_n, y_n)$ are IID

IID: Independent and identically distributed

Empirical Risk Minimization

$$R_{emp}(f, \boldsymbol{X}, \boldsymbol{y}) = \frac{1}{N}\sum_{i=1}^{n} l(y_n, \hat{y}_n)$$

Where $\boldsymbol{X} := [x_1, \dots, x_n]^T \in \mathbb{R}^{N \times D}$ and $\boldsymbol{y} := [y_1, \dots, y_n]^T \in \mathbb{R}^N$

Least Squares Loss

$$l(y_n, \hat{y}_n) = (y_n - \hat{y}_n)^2$$

$$R_{emp}(f, \boldsymbol{X}, \boldsymbol{y}) = \frac{1}{N}\sum_{i=1}^{n}(y_n - \hat{y}_n)^2 = \frac{1}{N}\sum_{i=1}^{n}\big(y_n - f(\boldsymbol{x}_n, \boldsymbol{\theta}\ )\big)^2$$

Generalization

After training, the model should be validated using data never seen before: $\mathcal{D}_{test} = \{(\boldsymbol{x}_m, y_m)\}, \quad n = 1 \dots M$

The test scores called generalization performance are the ones reported

Issue: Overfitting and Memorization

It is possible that for a given training set, a model with sufficient complexity is able to memorize the input-output data

This leads to a problem called overfitting

Regularization

To prevent overfitting, a regularizer is used

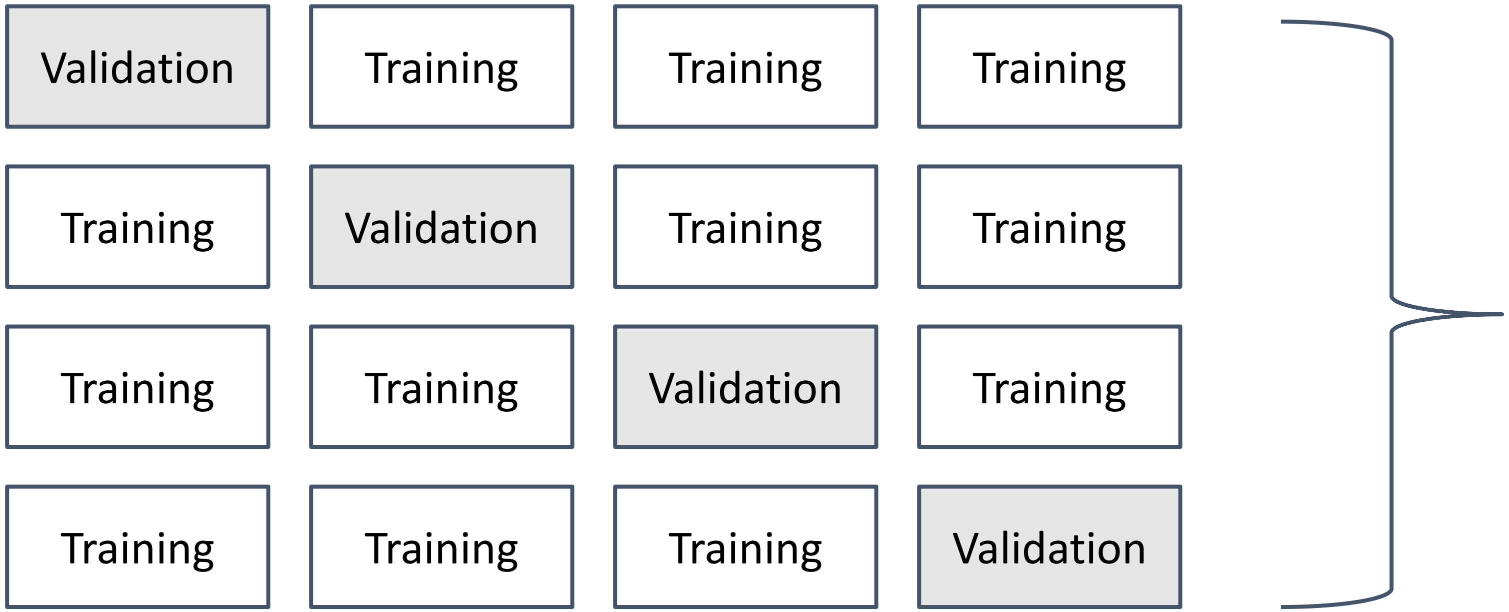
Examples: weight penalty, noise injection, inductive bias on dataset, novel model architecture, dropout

Weight Penalty: Scaled Dot Product of Parameters

$$R_{emp}(f, \boldsymbol{X}, \boldsymbol{y}) = \frac{1}{N}\sum_{i=1}^{n}\left(y_n - f(\boldsymbol{x}_n, \boldsymbol{\theta}\ )\right)^2 + \lambda\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{\theta}$$

Cross-Validation: Use when there is a small dataset (100s to few 1000s)

For Example, 4-fold Validation



Validation	Training	Training	Training
Training	Validation	Training	Training
Training	Training	Validation	Training
Training	Training	Training	Validation

4 folds validation performance is averaged

Probabilistic Parameter Estimation

Maximum Likelihood Estimation (MLE)

Function of parameters that explain the data well:

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = -\log p(\boldsymbol{\mathcal{D}}|\boldsymbol{\theta})$$

The likelihood of parameters $\boldsymbol{\theta}$ given that we observed data $\mathcal{D}$

Maximum because the loss function is minimized when $p(\mathcal{D}|\boldsymbol{\theta}) \to 1.0$

Log does not change the location of minima

Supervised Learning using Gaussian Distribution

$$\mathcal{D}_{train} = \{(\boldsymbol{x}_n, y_n)\}, \quad n = 1 \dots N$$

Target model:$p(\boldsymbol{\mathcal{D}}|\boldsymbol{\theta}) \rightarrow p(\hat{y}_n|\boldsymbol{x}_n, \boldsymbol{\theta})$

Assume a Gaussian distribution with mean as a linear function μ $= \boldsymbol{x}_n^T\boldsymbol{\theta}$ and with the parameter perturbed by a Gaussian noise with zero mean and variance σ^2:

$$p(\hat{y}_n|\boldsymbol{x}_n, \boldsymbol{\theta}) = \mathcal{N}(\hat{y}_n|\boldsymbol{x}_n^T\boldsymbol{\theta}, \sigma^2)$$

Independent Identically Distributed (IID) Assumption

Independent : probabilities can be multiplied $p(a, b) = p(a)p(b)$

Identically Distributed : can share parameters

We can factorize the data distribution:

$$p(\boldsymbol{\mathcal{D}}|\boldsymbol{\theta}) = p(\mathcal{Y}|\mathcal{X}, \boldsymbol{\theta}) = \prod_{n=1}^{N} p(\hat{y}_n|\boldsymbol{x}_n, \boldsymbol{\theta})$$

MLE

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = -\log p(\mathcal{Y}|\mathcal{X}, \boldsymbol{\theta}) = -\sum_{n=1}^{N} \log p(\hat{y}_n|\boldsymbol{x}_n, \boldsymbol{\theta})$$

Since $\log ab = \log a + \log b$

If $p(\hat{y}_n|\boldsymbol{x}_n, \boldsymbol{\theta})$ is a Gaussian

$$p(\boldsymbol{\mathcal{D}}|\boldsymbol{\theta}) = p(\mathcal{Y}|\mathcal{X}, \boldsymbol{\theta}) = \prod_{n=1}^{N} \mathcal{N}(\hat{y}_n|\boldsymbol{x}_n^T\boldsymbol{\theta}, \sigma^2)$$

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = -\log p(\mathcal{Y}|\mathcal{X}, \boldsymbol{\theta}) = -\sum_{n=1}^{N} \log \mathcal{N}(\hat{y}_n|\boldsymbol{x}_n^T\boldsymbol{\theta}, \sigma^2)$$

If $p(\hat{y}_n|\boldsymbol{x}_n, \boldsymbol{\theta})$ is a Gaussian

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = -\sum_{n=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}} \exp - \left(\frac{(\hat{y}_n - \boldsymbol{x}_n^T \boldsymbol{\theta})^2}{2\sigma^2} \right)$$

$$\mathcal{L}_{\mathcal{D}}(\boldsymbol{\theta}) = \sum_{n=1}^{N} \underbrace{\left(\frac{(\hat{y}_n - \boldsymbol{x}_n^T \boldsymbol{\theta})^2}{2\sigma^2} \right)}_{\text{Squared Losses}} - \sum_{n=1}^{N} \underbrace{\log \frac{1}{\sqrt{2\pi\sigma^2}}}_{\text{Constant}}$$

Maximum A Posteriori (MAP) Principle

$$p(\boldsymbol{\theta}|\boldsymbol{\mathcal{D}}) = \frac{p(\boldsymbol{\mathcal{D}}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{\mathcal{D}})} \propto p(\boldsymbol{\mathcal{D}}|\boldsymbol{\theta})p(\boldsymbol{\theta})$$

Since $p(\boldsymbol{\mathcal{D}})$ is not a function of $\boldsymbol{\theta}$, it does not affect the optimization

$$\log p(\boldsymbol{\theta}|\boldsymbol{\mathcal{D}}) = \log p(\boldsymbol{\mathcal{D}}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})$$

$$\underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log p(\boldsymbol{\theta}|\boldsymbol{\mathcal{D}}) = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}(\log p(\boldsymbol{\mathcal{D}}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}))$$

Maximum A Posteriori (MAP) Principle

$$\underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log p(\boldsymbol{\theta}|\boldsymbol{\mathcal{D}}) = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}(\underbrace{-\log p(\boldsymbol{\mathcal{D}}|\boldsymbol{\theta})}_{\text{MLE}} \underbrace{- \log p(\boldsymbol{\theta})}_{\text{Regularizer}})$$

If we assume $p(\boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, then:

$$-\log p(\boldsymbol{\theta}) = \lambda \boldsymbol{\theta}^{\boldsymbol{T}} \boldsymbol{\theta}$$

Capacity

Capacity - ability to fit a wide variety of functions

↓ Capacity → Underfitting: ↑ MSE(train) , ↑ MSE(test)

↑ Capacity → Overfitting: ↓ MSE(train) , ↑ MSE(test)

✓ Capacity → Optimal Fit: ↓ MSE(train) , ↓ MSE(test)

Sample Data

Input: x is a random number from -3 to +3

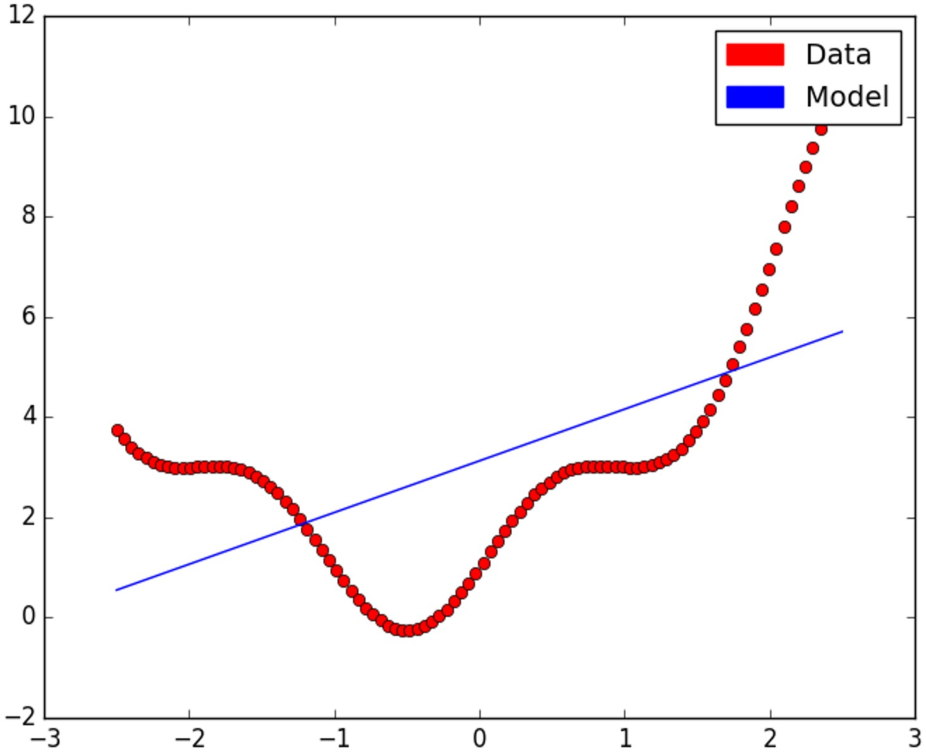
Output: $f(x) = y = x^2 + x + 1 + \underbrace{\lambda \sin \beta x}_{\text{Noise}}$

Model: Our model should be a 2nd degree polynomial

Suppose we can only see the data generated by $f(x)$

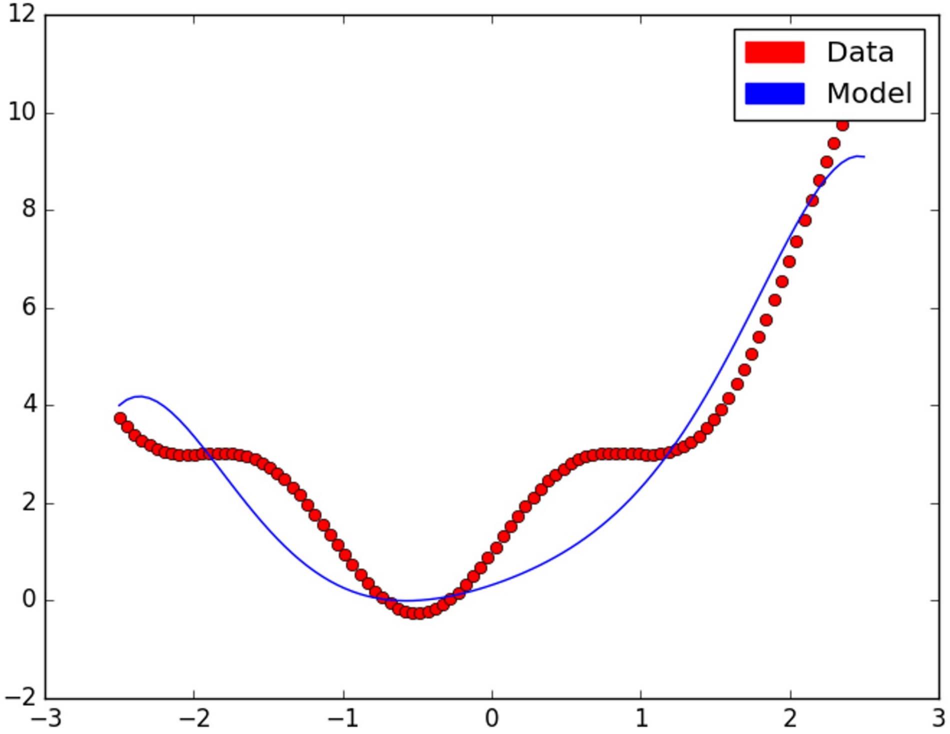
Model: $f(x) = y = \theta_1 x + \theta_0$

Underfitting: the model has both big train and test error



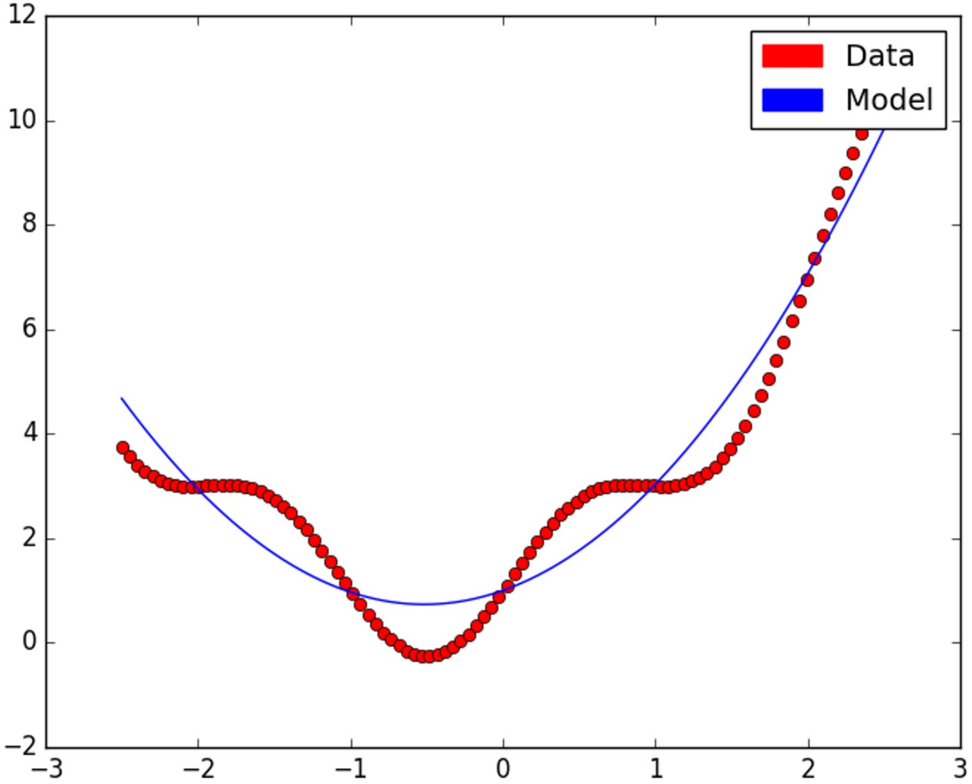
Model: $f(x) = y = \theta_6 x^6 + \cdots + \theta_1 x + \theta_0$

Overfitting: the model has a small train error but has a big test error



Model: $f(x) = y = \theta_2 x^2 + \theta_1 x + \theta_0$

Optimal fit: the model has a small train and test errors



Directed Graphical Models (DGM)

A graphical language for specifying a probabilistic model

DGM

Given a joint distribution $p(x, y, z)$, DGM represents the conditional dependencies of random variables

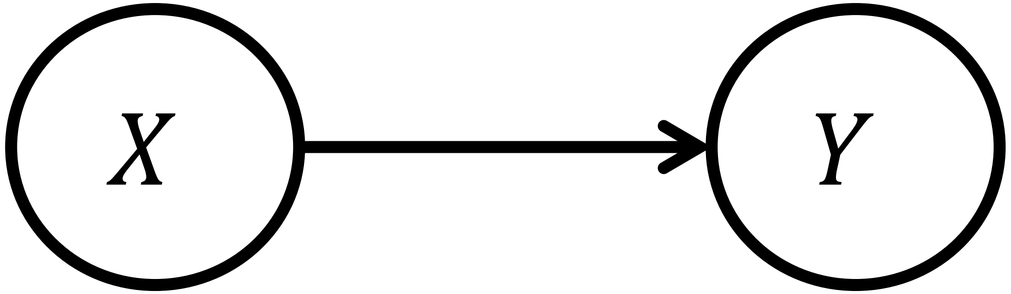
Compact notation to describe factorization of a joint distribution

Graphical Model

Node : a random variable

Edge : conditional probability

$p(Y|X)$

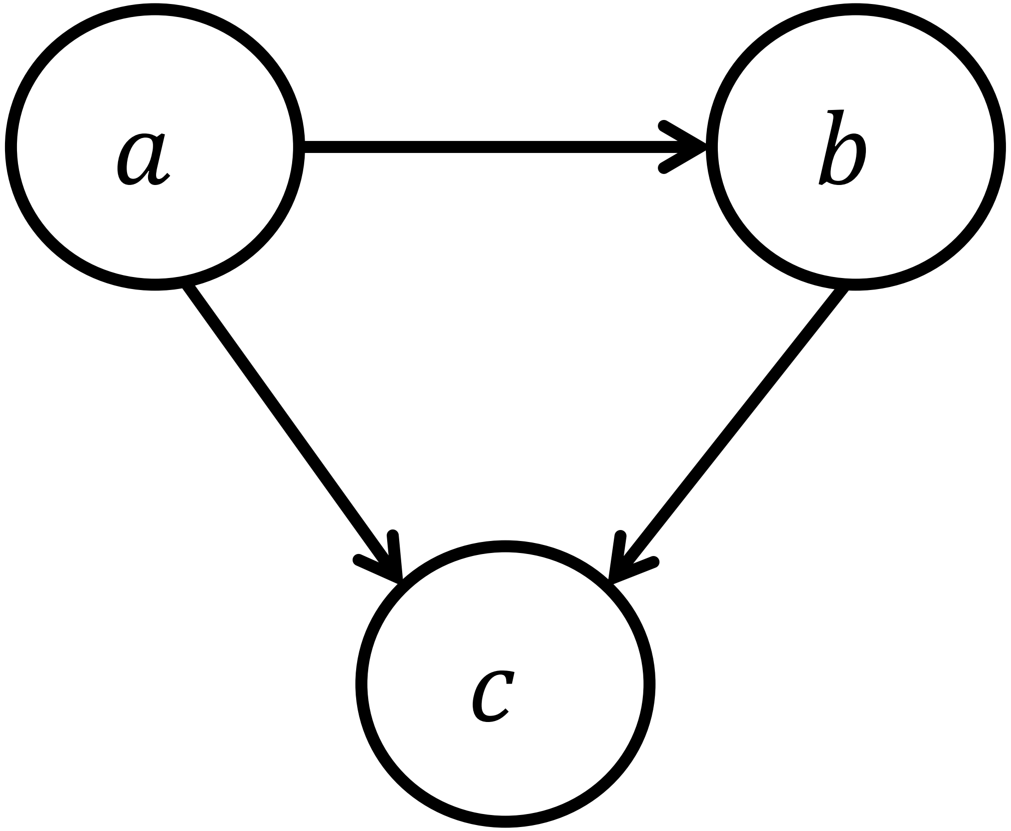


Joint Probability DGM

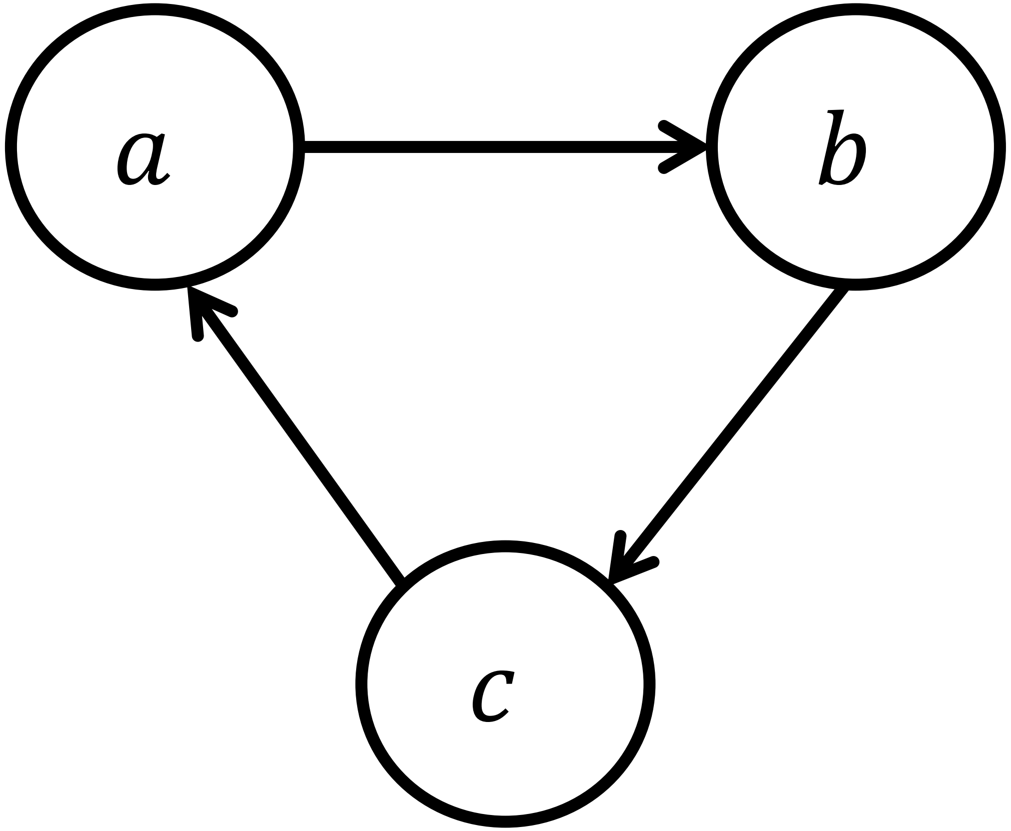
Create a node for all random variables

For each conditional, add a directed link to the graph from the nodes on which the random variable is conditioned

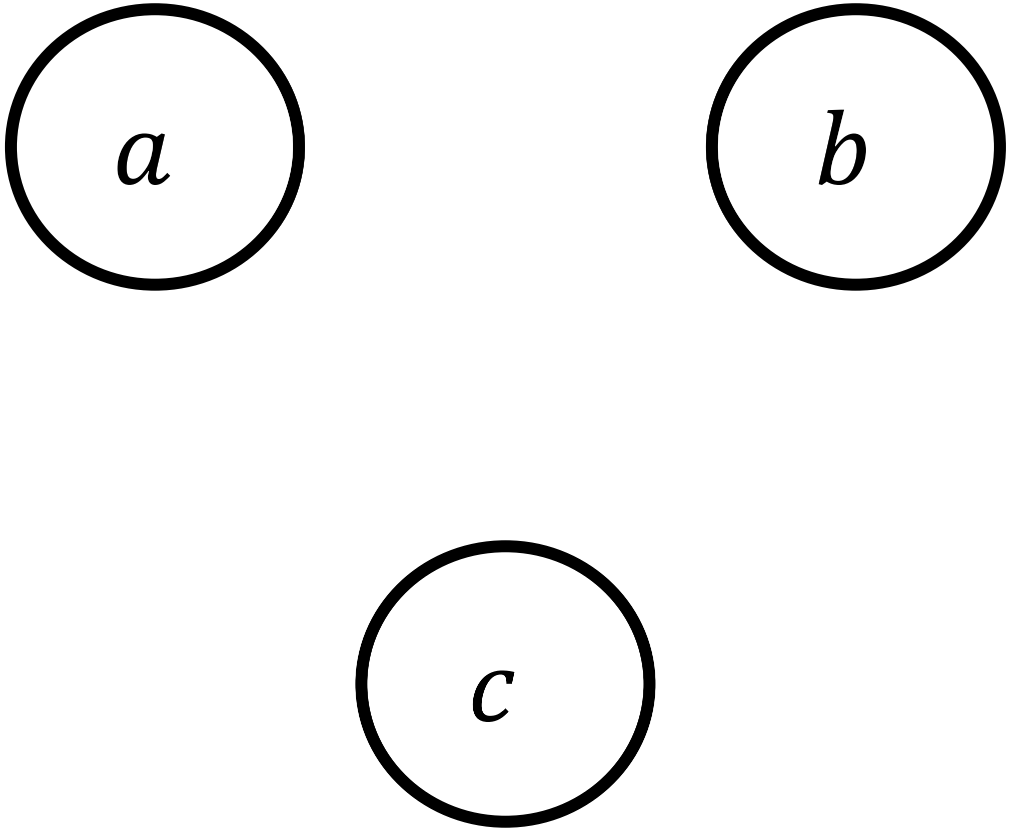
$$p(a, b, c) = p(c|a, b)p(b|a)p(a)$$



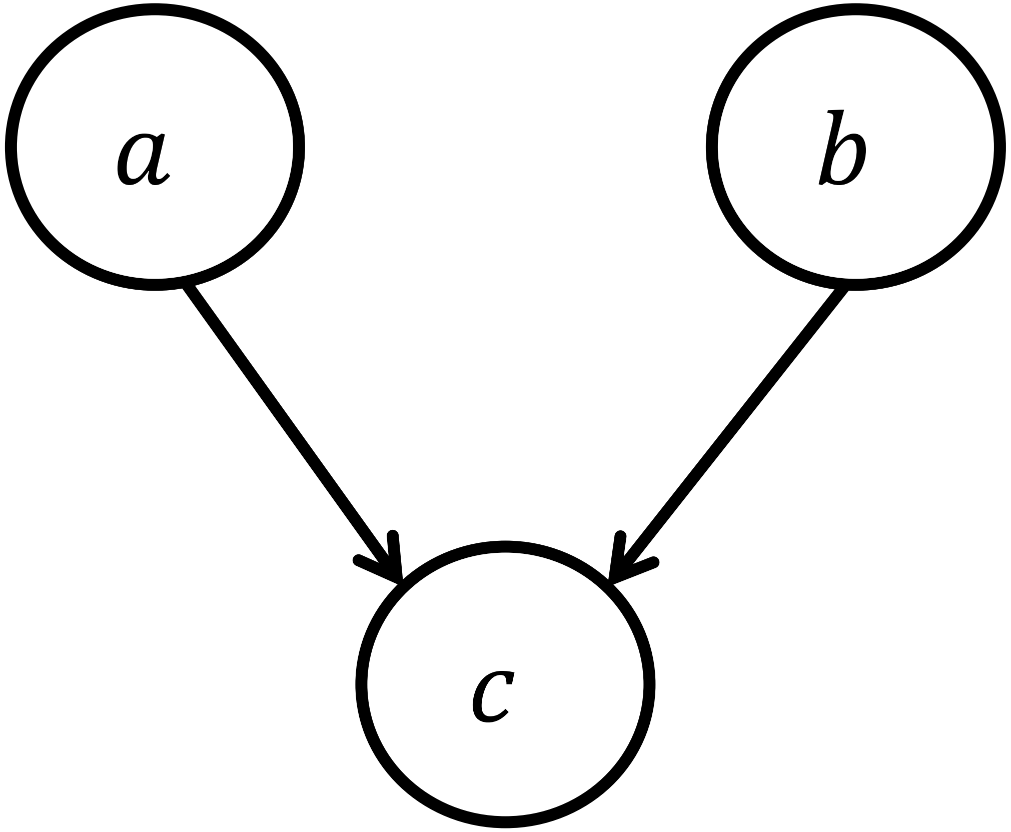
$$p(a,b,c) = p(c|b)p(b|a)p(a|c)$$



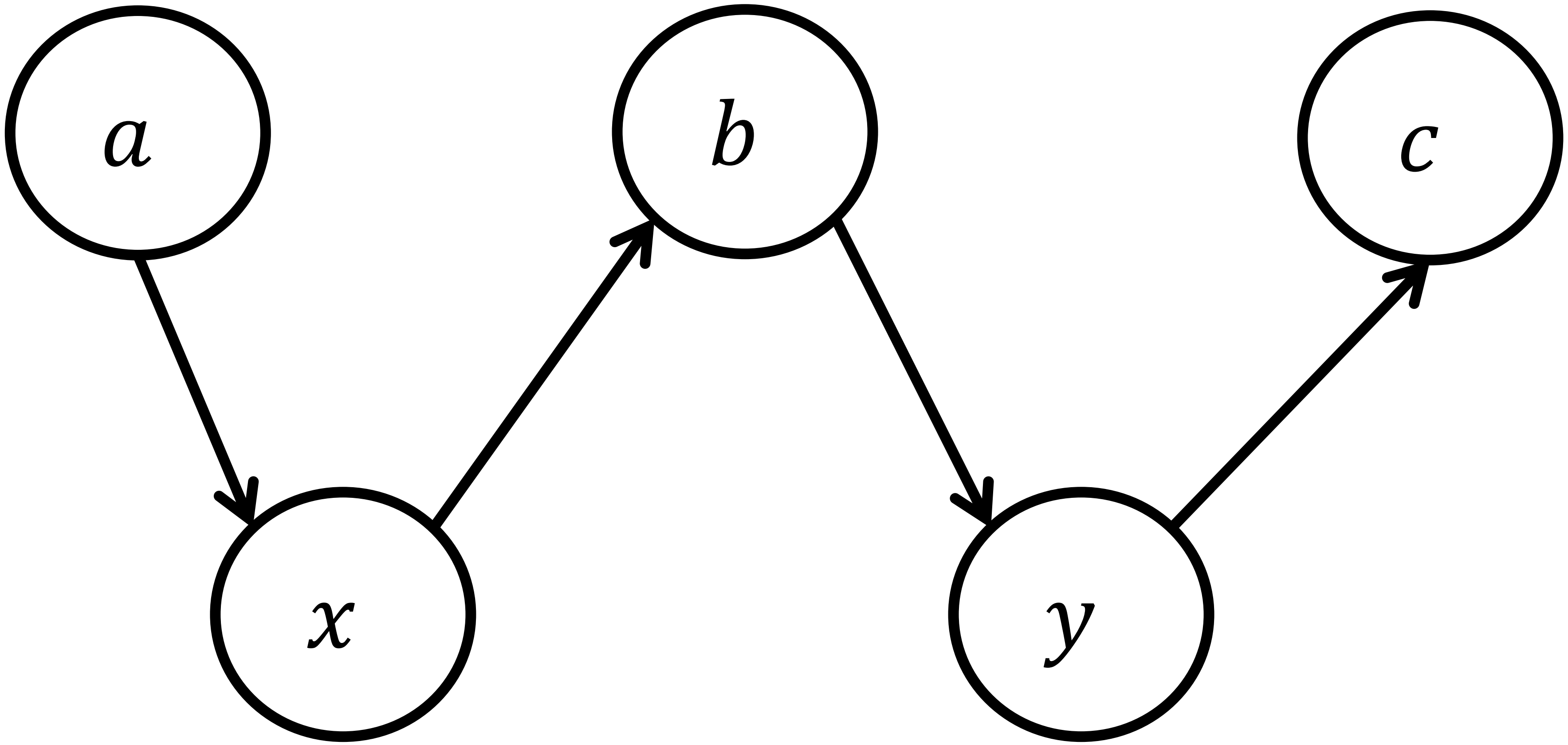
$$p(a, b, c) = p(a)p(b)p(c)$$



$$p(a, b, c) = p(c|a, b)p(a)p(b)$$



$$p(a,b,c,x,y) = p(a)p(x|a)p(b|x)p(y|b)p(c|y)$$

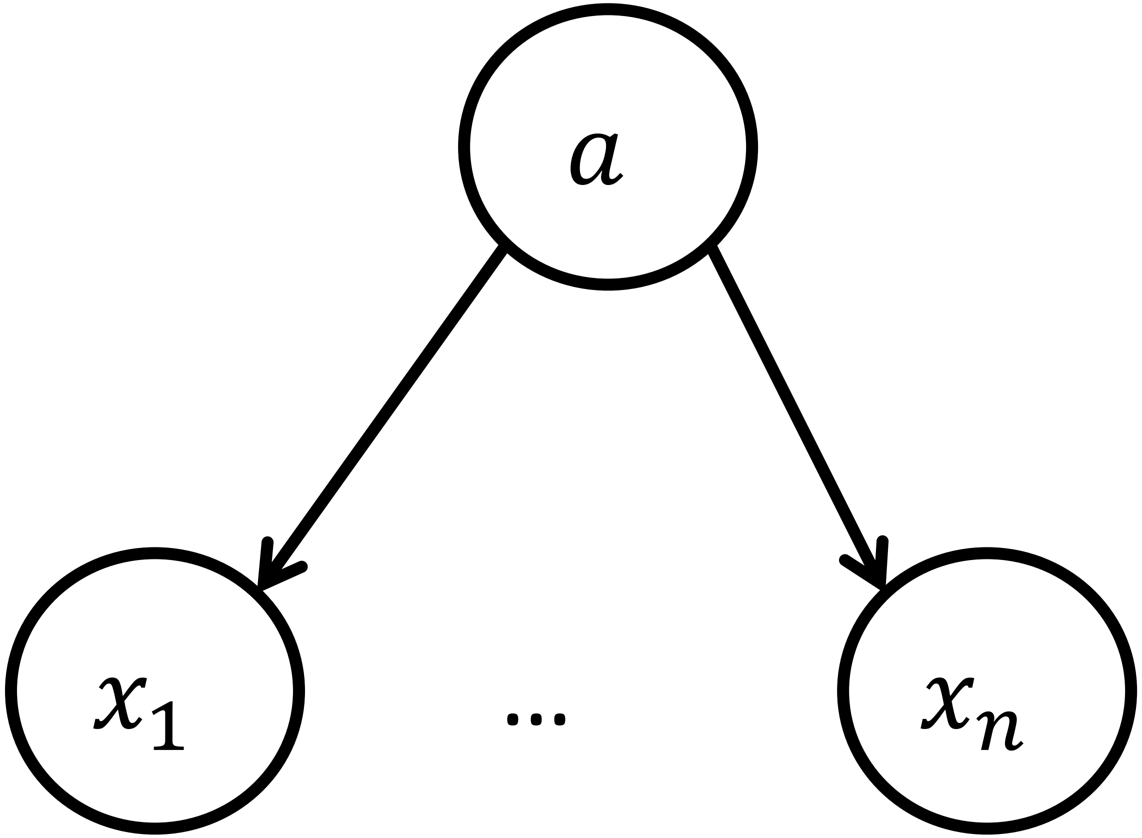


General Form of Joint Probability

$$p(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} p(x_i | Pa(x_i))$$

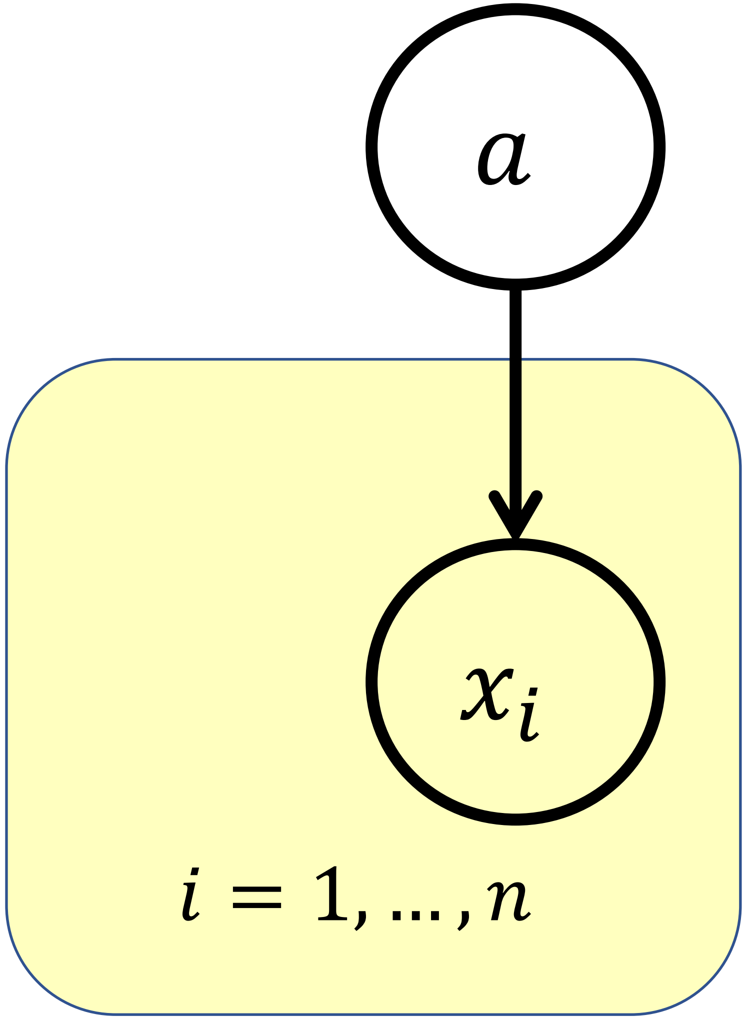
Where $Pa(x_i)$ are the parent random variables of x_i

Joint Probability Conditioned on 1 RV



$$p(x_1, \ldots, x_n | a) = \prod_{i=1}^{n} p(x_i | a)$$

Joint Probability Conditioned on 1 RV



$$p(x_1, \dots, x_n | a) = \prod_{i=1}^{n} p(x_i | a)$$

Conditional Independence and d-Separation

Recall conditional independence, $a \perp b|c : p(a,b|c) = p(a|c)p(b|c)$

Implies directed acyclic graph

General form in terms of sets of RVs: $\mathcal{A} \perp \mathcal{B}|\mathcal{C}$

Blocked Path

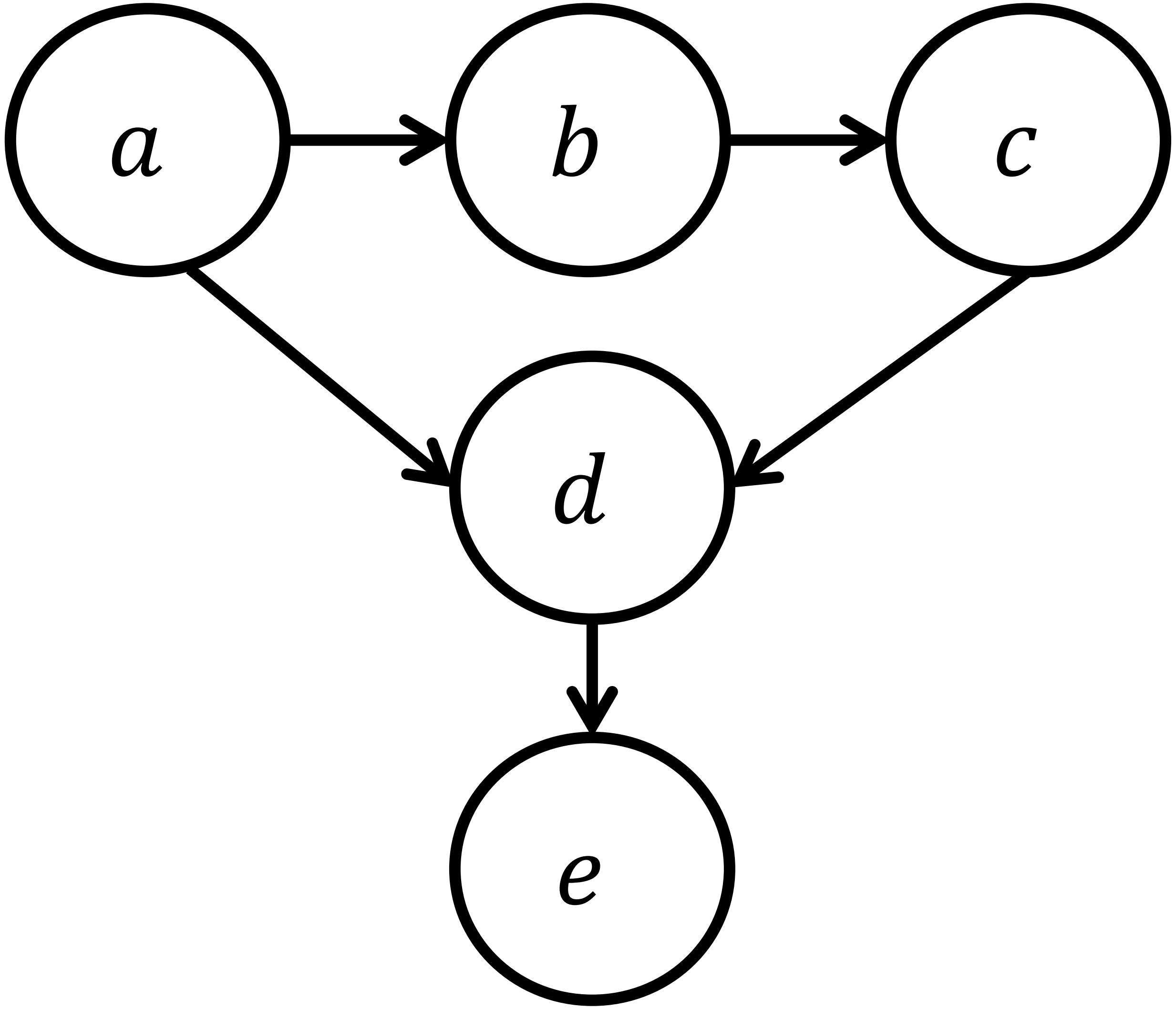
Consider all trails (paths w/o arrows) from $\mathcal{A}$ to $\mathcal{B}$

Blocked if:

1. The arrows on the path meet head to tail, $\mathcal{A} \rightarrow \mathcal{C} \rightarrow \mathcal{B}$, or tail to tail, $\mathcal{A} \leftarrow \mathcal{C} \rightarrow \mathcal{B}$, at the node and the node is in set $\mathcal{C}$
2. The arrows on the path meet head to head , $\mathcal{A} \rightarrow \mathcal{C} \leftarrow \mathcal{B}$, at the node and neither the node nor any of its descendants is in set $\mathcal{C}$

If all paths are blocked, then $\mathcal{A}$ is d-separated from $\mathcal{B}$ by $\mathcal{C}$ and satisfies the conditional independence $\mathcal{A} \perp \mathcal{B}|\mathcal{C}$

Example



$$b \perp d | a, c$$
$$a \perp c | b$$
$$b \not\perp d | c$$
$$a \not\perp c | b, e$$

Introduction to Information Theory

Information Theory (Intuition)

Likely event has no information (e.g. the sun rises in the east)

Less likely event has higher information (e.g. solar eclipse at 12n)

Independent events have additive information (e.g. solar eclipse at 12n + people are coming out from the building to see solar eclipse → Conclusion?)

Information Theory

Self Information of event $x = x$: $I(x) = -logP(x)$

Measure of surprise or uncertainty

Measured in nats (bits and shannon are also used dep on the log's base)

$\log()$ should be interpreted as $\ln()$

Shannon Entropy

$$H(x) = H(P) = \mathbb{E}_{x\sim P}[I(x)] = -\mathbb{E}_{x\sim P}[\log P\,(x)]$$

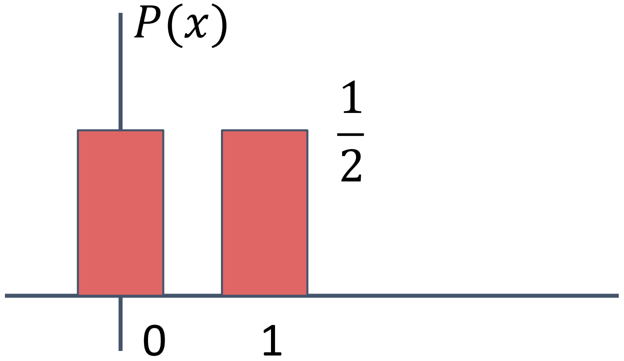
Number of bits needed to encode symbols drawn with PD $P(x)$

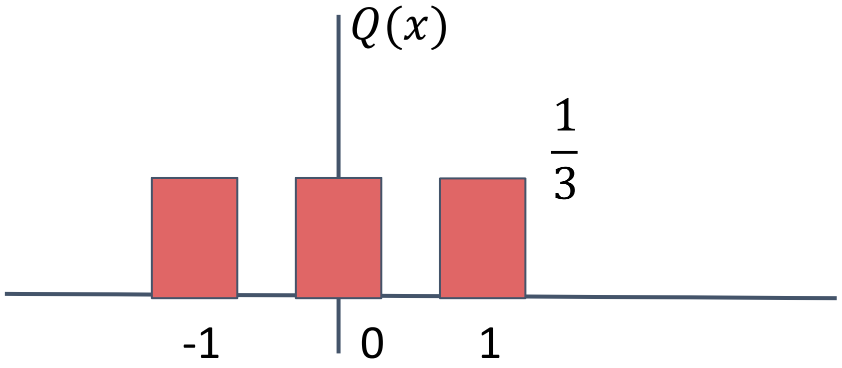
Kullback-Leibler Divergence

$$D_{KL}(P\|Q) = \mathbb{E}_{x\sim P}\left[\log \frac{P(x)}{Q(x)}\right] = \mathbb{E}_{x\sim P}[\log P(x) - \log Q(x)]$$

KL is a measure of distance between two distributions

What is $D_{KL}(P\|Q)$? $D_{KL}(Q\|P)$?



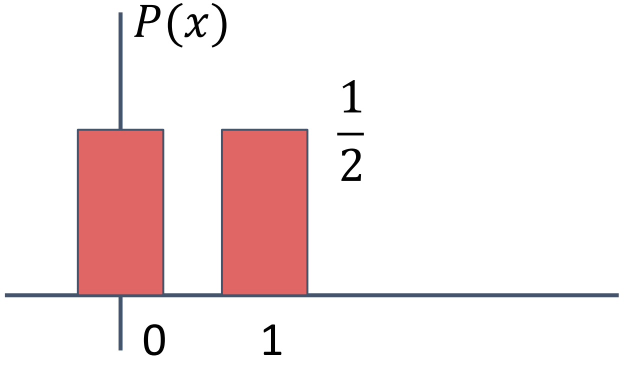


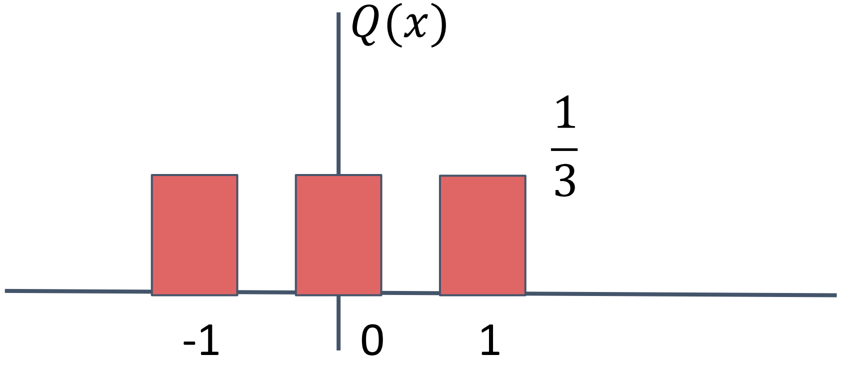
Kullback-Leibler Divergence

$$D_{KL}(P\|Q) = \mathbb{E}_{x\sim P}[\log P(x) - \log Q(x)]$$

$$D_{KL}(P\|Q) = 2\left(\frac{1}{2}\left(\log\frac{1}{2} - \log\frac{1}{3}\right)\right) = \log\frac{3}{2}$$

What is $D_{KL}(Q\|P)$? Is $D_{KL}(P\|Q) = D_{KL}(Q\|P)$?





Cross Entropy

Cross Entropy:

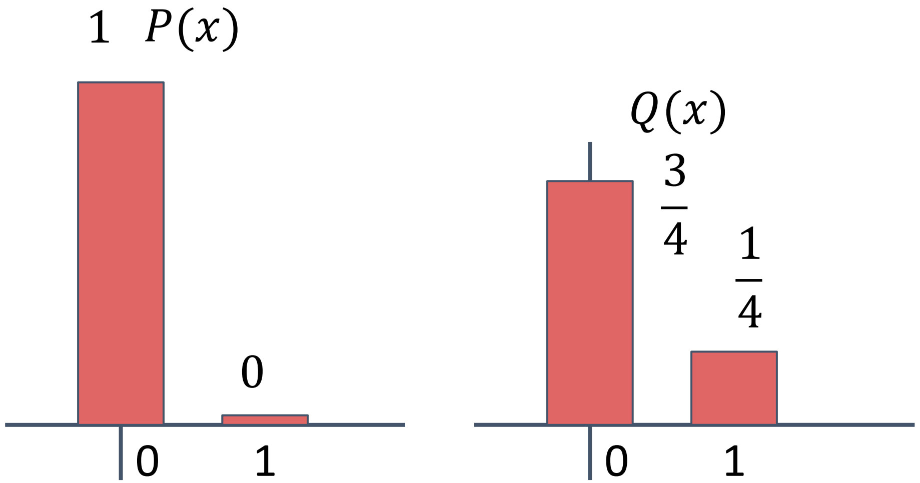
$$H(P,Q) = \underbrace{H(P)}_{\text{Entropy}} + \underbrace{D_{KL}(P\|Q)}_{\text{Distance}} = -\mathbb{E}_{x\sim P}[\log Q(x)]$$

In Machine Learning, minimizing $H(P,Q)$ minimizes the distance of prediction model Q from the empirical model P

Categorical Cross-Entropy

For discrete distribution, Categorical Cross-Entropy is:

$$CE = H(P,Q) = -\underbrace{\textstyle\sum_i P(x_i)}_{\text{Empirical Label}} \underbrace{\log Q(x_i)}_{\text{Predicted Label}}$$

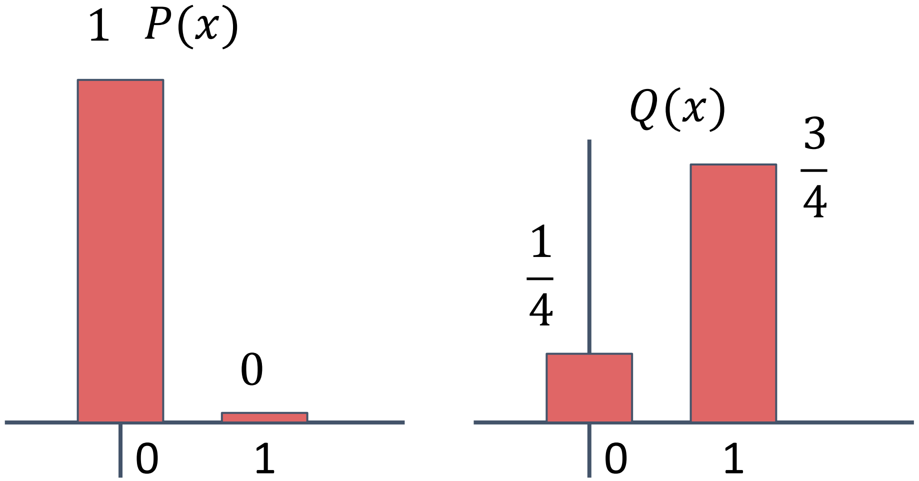


What is CE?

$$CE = -\left(1 \log \frac{3}{4} + 0 \log \frac{1}{4}\right) = 0.29$$

What is CE?

$$CE = -\left(1 \log \frac{1}{4} + 0 \log \frac{3}{4}\right) = 1.39$$



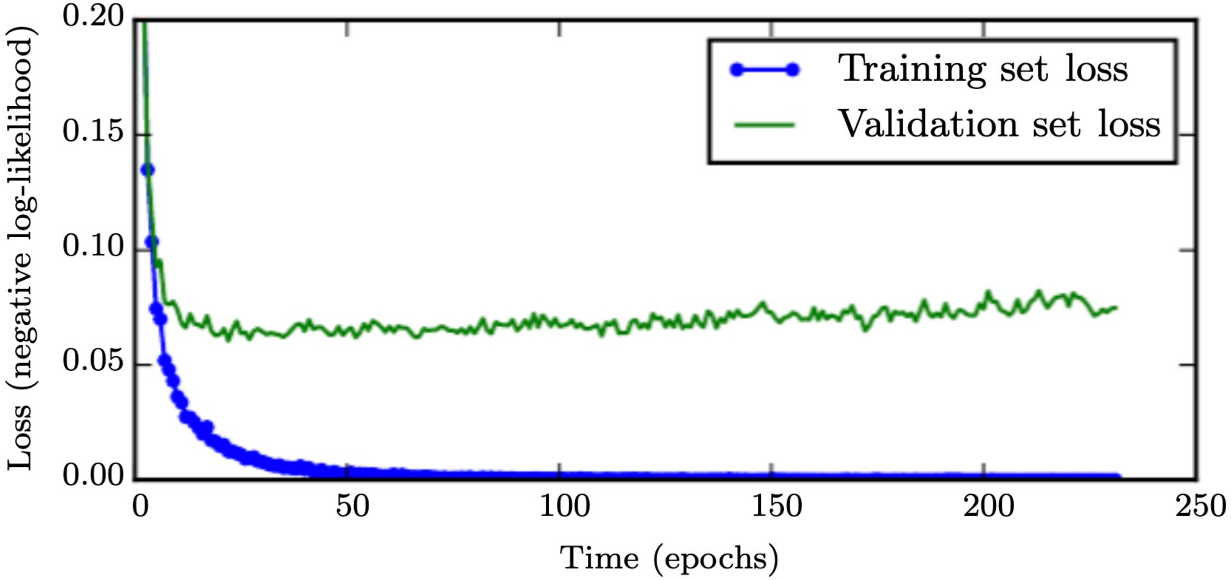
In Summary

A parametric model learns from data using ERM, MLE, or MAP

CE (hence KL) measures how far is the model prediction from empirical label

Regularization is a method to prevent overfitting by improving the generalization error

Overfitting



Deep Learning Book, Goodfellow et al 2016