"A breakthrough in Machine Learning would be worth ten Microsofts."
– Bill Gates (2004)

Lectures 1&2: Introduction, Background on Probability and Information Theory & Course Overview

Prof. Krishna R. Pattipati
BoT Distinguished Professor &
UTC Chair Professor in Systems Engineering
Dept. of Electrical and Computer Engineering
University of Connecticut
Contact: krishna.pattipati@uconn.edu; (860) 486-2890

Spring 2021
January 19 & January 26, 2021

Reading List

- Chapters 1 & 2 of Bishop
- Chapter 1 of *Model-based Machine Learning* by Winn and Bishop (Online)
- Chapters 1 & 2 of Murphy
- Chapter 2 of Theodoridis
- Class Notes
- Science:
  - Special Issue on AI, July 17, 2015, Pages 248-278
  - AI Transforms Science, July 7, 2017, Pages 16-30
- Web
  - https://waitbutwhy.com/2015/01/artificial-intelligence-revolution-1.html
  - https://waitbutwhy.com/2015/01/artificial-intelligence-revolution-2.html
- YouTube (Bishop's talk on AI: The History and Future)
  - https://www.youtube.com/watch?v=8FHBh_OmdsM

Why Do Machines Need to Learn?

- Deluge of data (trillions of web pages, 10 years of video content in a single day on YouTube, retail sales, Facebook, twitter, blogs,…)
- Don’t know Concise I/O or cause-effect relationships
- Are there hidden relationships and correlations (“Patterns”) in large data?
- Need to adapt to an unknown environment (“Uncertainty”)
- New knowledge is constantly being rediscovered (“Dynamic”)
- Entering knowledge by humans is tedious

Machine Learning Applications

- Computational Biology
  - Classifying protein sequences, identifying protein-coding genes, genetic networks from gene expression data, …
- Medical Imaging
  - Computer-aided diagnosis, image-guided therapy
  - EEG classification, multi-model image fusion
- Information Retrieval/Natural Language Processing
  - Text/audio/image retrieval
  - Parsing/translation/text analysis/document classification/spam filters
- Robotics
  - Autonomous driving, planning, control
- Speech processing
  - Voice identification, speech recognition, speech → text → speech (in another language)
- Recommender Systems
  - Collaborative filtering, market basket analysis,….
- Scientific Data Analysis
  - E.g., NASA missions, Hadron Collider,…
- Financial Prediction and Automated Trading
- Computer Games, Gambling,….

What is Learning?

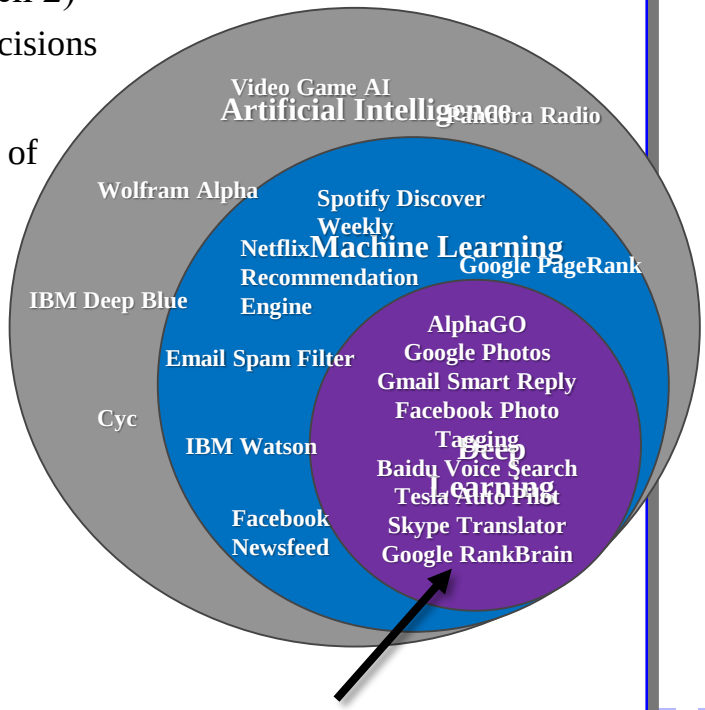
- Learning: *Improving* with *Experience* at some *Task*
  - *Improve over task, T*
  - *With Respect to Performance Measure, P*
  - *Based on Experience, E*
- Example: *Spam Filtering* (supervised learning)
  - *Spam: An E-mail that the user does not want to receive and has not asked to receive*
  - *T: Identify Spam E-mails*
  - *P: % Spam E-mails that were filtered (Detection, Sensitivity)*

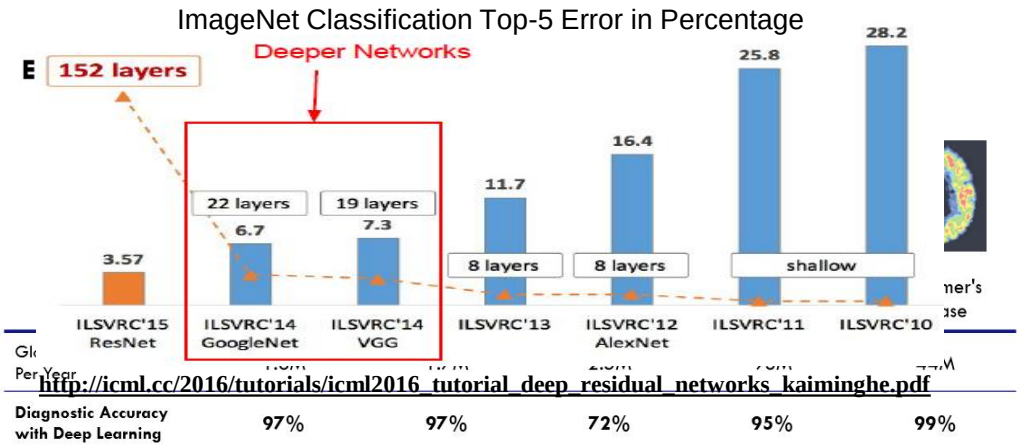
    % of ham (non-spam) E-mails that were incorrectly filtered-out (False Alarms)
  - *E: A Database of E-mails that were labelled by users/experts*
- Learning spans multiple fields: *EE, CS, Statistics, Cognitive Science, Computational Neuroscience, Computational Biology, Social Sciences, Financial Engineering, Economics, ….*

AI versus ML versus Deep Learning

AI: The study of the Computations that make it possible to **P**erceive, **R**eason and **A**ct

- Classic AI is based on deductive logic (Gen 1)
  - Rules are based on human ingenuity
- Machine Learning is based on learning "models" from data (Gen 2)
  - Rules/Models derived from data are used for predictions and decisions
- Deep learning seeks to "mimic" the biological brain (Gen 3)
  - Learn and recognize patterns of input data using multiple layers of abstraction
  - Applications:
    - Image classification: Facial recognition
    - Speech recognition in phones: Siri, Google assistant
    - Game playing: Chess, Atari, Alpha GO
    - Disease prediction
    - Self-driving cars





"Greedy, Brittle, Opaque and Shallow" Deep learning addresses the so-called '1-second' problem (narrow intelligence)

What Made Deep Learning Feasible?

- Deluge of data and ability to crowd source the labeling process for supervisory learning …Availability of *large scale (and often pre-trained) labeled images* (e.g., Alexnet, ImageNet, VGG16) and advances in *cross-modal* (image, text/data, video and audio) *representations*…. "*Big Data*"
- Availability of *Graphics Processing Units* (GPUs) that enable training of very large image/text/speech datasets from several weeks to a few days
- *New nonlinearities* (e.g., rectified linear units) and *signal normalization* that avoid numerical problems associated with gradient computations
- *Convolution and max-pooling* operations that exploit local connectivity to reduce the dimension of the weight space, control overfitting and make the network robust
- Concept of *dropout* to realize an exponentially large ensemble of networks from a single network
- Advances in *stochastic optimization, adversarial training and regularization* for robust network training
- *More layers capture more invariances* (Information-theoretic insights)
- *Much better software tools* (Keras, TensorFlow, Theano, Torch,…)
- **Result**:*Features learned rather than hand-crafted, better accuracy (but easily fooled!)*

Types of Learning

- Learning from Data: Model Parameters, W (weights)

$W = f(D)$ $\quad D \rightarrow$ data used to learn (training data)

There are at least four forms of learning (others: active, transfer)

a) Supervised Learning (or learning with a teacher)
b) Unsupervised Learning
c) Semi-supervised Learning
d) Reinforcement Learning (like learning with a critic)

a) Supervised Learning:

$w = f[\{\underline{x}_n, \underline{z}_n\}_{n=1}^{N}]$ (know inputs and correct outputs)

- Classification
- Regression
- Time series prediction

b) Unsupervised Learning:

$w = f[\{\underline{x}_n\}_{n=1}^{N}]$ (has input data only)

- Clustering; Density Estimation
- Outlier Detection
- Compression

c) Semi-supervised Learning: Practical setting; Active learning

$w = \{f[\{\underline{x}_n, \underline{z}_n\}_{n=1}^{N_1}]; \{f(\underline{x}_n)\}_{n=N_1+1}^{N}\}$ ($N_1 << N$: Few outputs are labeled, but most are not)

d) Reinforcement Learning: Useful in adaptive control and decision making

$w = f[\{\underline{x}_n\}_{n=1}^{N}$; correctness/effectiveness of outputs] (learning with a critic)

History of Learning Methods - 1

- Least Squares
  - Legendre (1805), Gauss (1795, 1809, 1821)
- Early NNs ("Hyped by its charismatic enthusiasts")
  - Binary unit-based Neuron Models: McCulloch & Pitts (1943)
  - Unsupervised learning: Hebb (1949)
  - Supervised learning
    - Classification: Rosenblatt's Perceptron (1958,1962)
    - Regression-based learning: Widrow & Hoff (1962)
    - Inability of Perceptron to model XOR (Minsky & Pappert, 1968)
- Visual Cortex in Cats (Wiesel & Hubel, 1959 & 1962)
  - Cells fire in response to visual sensory inputs
  - Certain cells exhibit spatial invariance ("features")
  - Basis for recent work in deep learning
- Group Method of Data Handling (Ivakhnenko, 1968 & 1971)
  - Polynomial activation functions
  - Layers are incrementally grown and trained by regression analysis
  - First feedforward multilayer perceptrons

History of Learning Methods - 2

- Neocognitron (Fukushima, 1979)
  - Convolution: A rectangular receptive field is shifted step by step over an image
  - Weight replication: smaller number of parameters
  - Subsampling: Spatial averaging or Max-pooling
- Backpropagation NNs
  - Optimal control (state and co-state equations) since 1960s
  - Incremental gradient-based BP (Werbos, 1974; Rumelhart, 1986)
  - Sequence processing recurrent NNs (Late 1980s)
  - Better BP through advanced gradient algorithms (1980s and 1990s)
  - Searching for simple, low-complexity NNs (1990s)
  - BP works for shallow networks only ("gradients vanish or explode")
  - Convolutional NNs (LeCun, 1989, 1990, 1998)
  - Rectilinear units (Hinton, 2010)
- Support vector machines (Vapnik, 1995)
  - Transform inputs into a higher dimensional feature space
  - Train a linear classifier in the transformed space
  - Variants: Relevance Vector Machines, Gaussian Processes,…

History of Learning Methods - 3

- Boosting
  - Stacking multiple classifiers
  - AdaBoost (2000)
- Benefits of Unsupervised Learning in Supervised Learning
  - UL methods provide distributed, sparse representation of input data
  - UL methods encode input data (“redundancy reduction”)
  - Autoencoders and stacked autoencoders (Ballard, 1987) fine tuned by BP (Bengio, 2009)
  - History compressing RNNs (Connor et al., 1994)
  - Restricted Boltzmann machines (RBMs) and stacked RBMs fine tuned by BPs (Hinton, 2006)
  - GPUs for deep NNs (2004 onwards)
  - GPU Max Pooling CNNs achieve superhuman vision performance
- Current Standards
  - GPU Max Pooling CNNs, GANs, ….
  - Long Short Term Memory RNNs (for reinforcement learning)

Things to Remember in ML Model Building

- Make Sure Training Data is Adequate ("Data Matters More than Algorithms")
- Make Sure Data is Representative ("No sampling Bias")
- Clean up the Data ("Detect and Fix/Remove Outliers", "Ignore/Estimate Missing Data", "Minimize Noise Effects in the Data")
- Feature Engineering ("Feature Selection", Feature Extraction", "Seek new Information")
- Avoid Overfitting and Underfitting ("Kiss" Principle, "Occam's Razor", "Theory of Parsimony", "Bias-Variance Tradeoff")
- Understand the difference between Training Error and Generalization (Out-of-sample) Error ("Split Data into Training, Validation and Test Data Sets", "No data snooping", "Test and Validate the Model", "Cross Validation", "Bootstrap", "Model Selection", "Bayesian Model Averaging")
- Exploit Domain Knowledge ("Relationships and Constraints", "No Free Lunch Theorem")

Some Interesting Datasets

Dataset	Accessible Location
IRIS	http://www.statlab.uni-heidelberg.de/data/iris/
MNIST	http://yann.lecun.com/exdb/mnist/
20Newsgroups	http://cs.nyu.edu/~roweis/data.html
Olivetti Faces	http://cs.nyu.edu/~roweis/data.html
Alarm Network	http://www.bnlearn.com/bnrepository/

UCI Machine Learning Repository: http://archive.ics.uci.edu/ml/index.php

Kaggle Data Sets: https://www.Kaggle.com/datasets

Amazon AWS: http://aws.amazon.com/fr/datasets

Open Source Data Portals: http://dataportals.org ; http://opendatamonitor.eu ; http://quandl.com;

Wiki: https://en.wikipedia.org/wiki/List_of_datasets_for_machine_learning_research

Deep Learning Datasets: http://deeplearning.net/datasets/

Blog: https://blog.algorithmia.com/machine-learning-datasets-for-data-scientists/

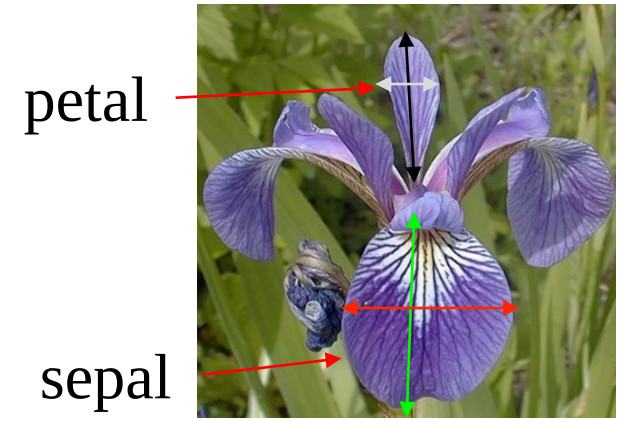
Fisher-Anderson's Iris Dataset

iris setosa

iris versicolor

iris virginica

What type of measurements will help distinguish these flowers well ?



petal length

petal width

sepal length

sepal width

http://www.statlab.uni-heidelberg.de/data/iris/

The answer to "what measurements to take" typically comes from subject matter experts, Botanists in this case.

Sample Data of Iris Dataset

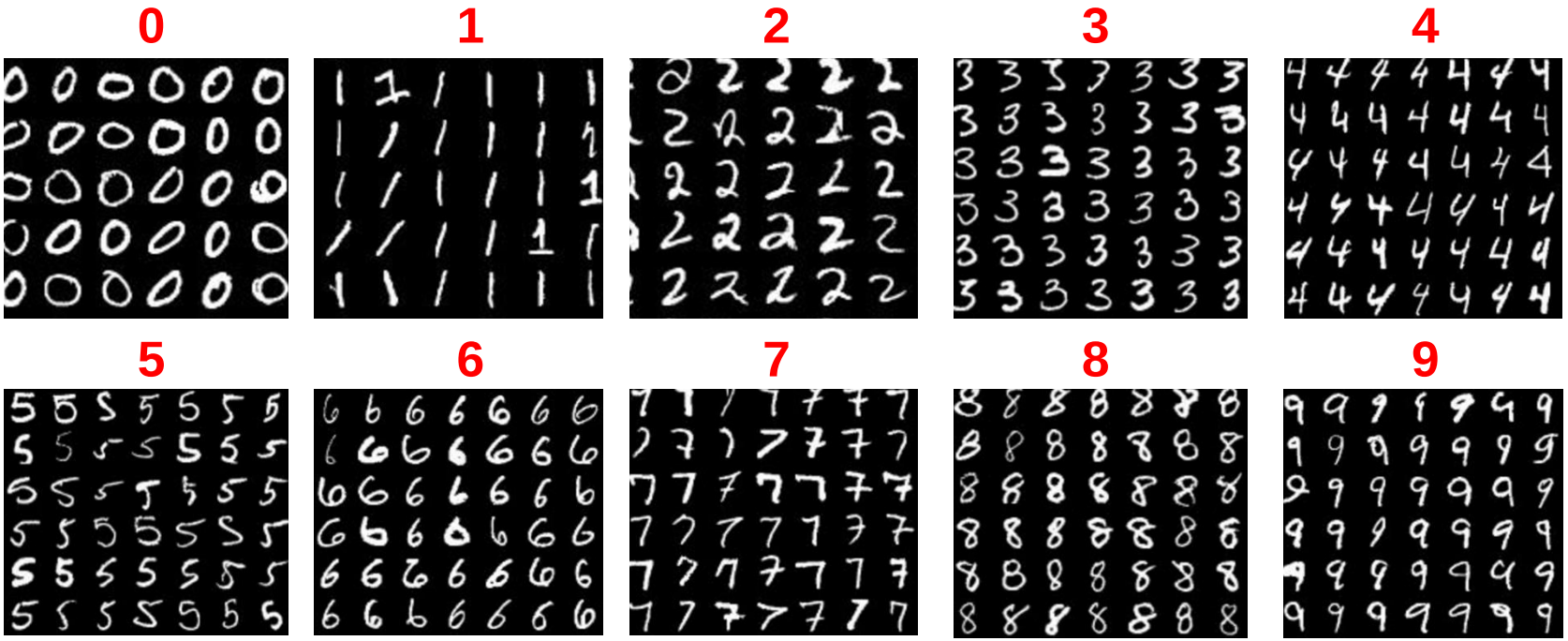
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5	3.6	1.4	0.2	setosa
7	3.2	4.7	1.4	versicolor
6.4	3.2	4.5	1.5	versicolor
6.9	3.1	4.9	1.5	versicolor
5.5	2.3	4	1.3	versicolor
6.5	2.8	4.6	1.5	versicolor
6.3	3.3	6	2.5	virginica
5.8	2.7	5.1	1.9	virginica
7.1	3	5.9	2.1	virginica
6.3	2.9	5.6	1.8	virginica
6.5	3	5.8	2.2	virginica

Both petal length and width clearly distinguish setosa; but the other two are harder to distinguish

1. R. A. Fisher (1936). "The use of multiple measurements in taxonomic problems." *Annals of Eugenics*. 7 (2): 179–188.

2. Edgar Anderson (1936). "The species problem in Iris." *Annals of the Missouri Botanical Garden*. 23 (3): 457–509.

MNIST Handwritten Digits Dataset



Data:

- Handwritten numbers {0,1, …, 9} of some 60,000 people is available in the form of 20x20 pixel images for training. There are 10,000 testing images.
- Objective is to use the above data for training

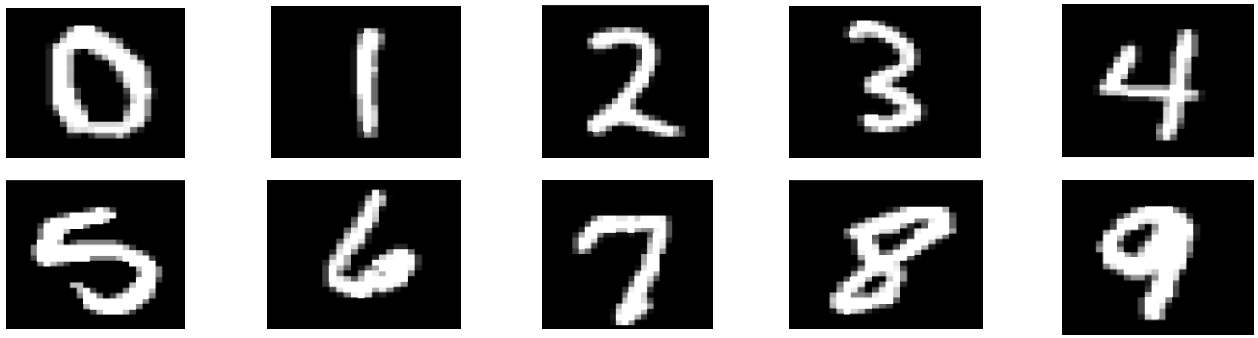
Challenge:

- Classification: Using the above training data, recognize ANY handwritten set of numbers.

http://yann.lecun.com/exdb/mnist/

Feature Definition in MNIST Dataset

- In iris data, each flower is specified by {sepal length, sepal width, petal length and petal width}
- Question: What features can be captured from each of the following images in order to capture the numbers they represent?



  - In the iris data, feature definition (selection) was done by subject matter experts
  - Feature definition for MNIST can be done through machine learning (e.g., deep autoencoders, stacked restricted Boltzmann machines (RBMs)) from the raw pixel data
  - Deep Convolutional Networks have achieved human-like performance in this area. Best accuracy is around 99.79%.

20 Newsgroup Dataset

- The objective is to group news reports into several categories (20 categories to be specific)
- News reports are assumed to be text (no graphics)
- Feature selection: The 100 words on the right are selected as features (by subject matter experts)
- For each news report, it was checked whether one of these words is present or absent in the report
- There are 18,828 news reports in the dataset
- Out of that, 4,000 are shown in a later slide (slide 30)

'aids'	'baseball'	'bible'	'bmw'	'cancer'
'car'	'card'	'case'	'children'	'christian'
'computer'	'course'	'data'	'dealer'	'disease'
'disk'	'display'	'doctor'	'dos'	'drive'
'driver'	'earth'	'email'	'engine'	'evidence'
'fact'	'fans'	'files'	'food'	'format'
'ftp'	'games'	'god'	'government'	'graphics'
'gun'	'health'	'help'	'hit'	'hockey'
'honda'	'human'	'image'	'insurance'	'israel'
'jesus'	'jews'	'launch'	'law'	'league'
'lunar'	'mac'	'mars'	'medicine'	'memory'
'mission'	'moon'	'msg'	'nasa'	'nhl'
'number'	'oil'	'orbit'	'patients'	'pc'
'phone'	'players'	'power'	'president'	'problem'
'program'	'puck'	'question'	'religion'	'research'
'rights'	'satellite'	'science'	'scsi'	'season'
'server'	'shuttle'	'software'	'solar'	'space'
'state'	'studies'	'system'	'team'	'technology'
'university'	'version'	'video'	'vitamin'	'war'
'water'	'win'	'windows'	'won'	'world'

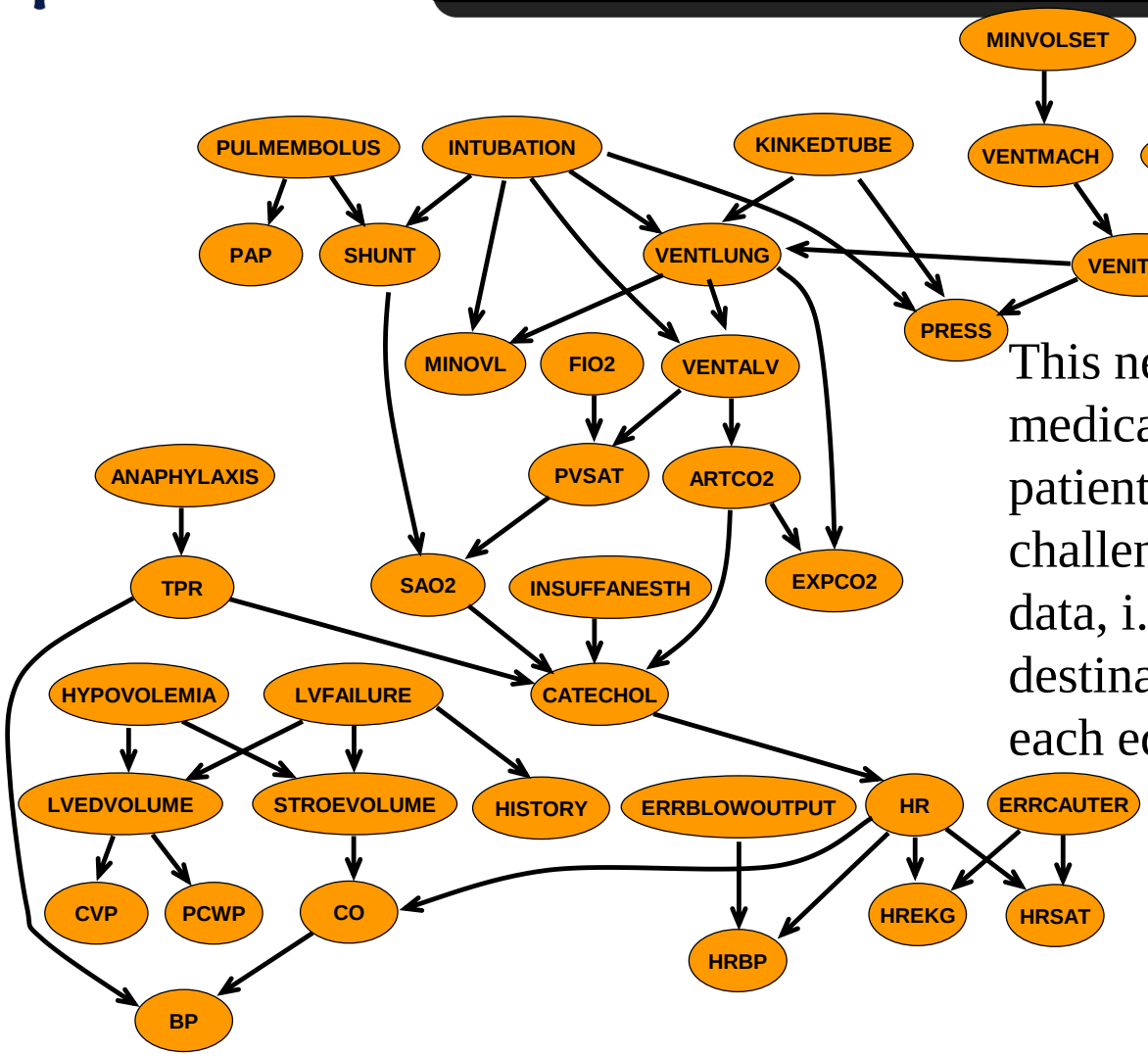
http://cs.nyu.edu/~roweis/data.html

Olivetti Face Dataset (Description)

http://cs.nyu.edu/~roweis/data.html

- The database contains 8-bit [0 - 255], gray scale faces of people
- Ten different faces of the same person in different poses are included in the database
- There are in total 400 faces in the database (10 different poses from 40 different people)
- Feature Selection: What features to extract and store for each face?
- Classification: Given the picture of a face, how to identify it from a database of millions (if not billions) of faces?

Alarm Network Dataset



37 variables
Discrete data

This network is created by medical experts for monitoring patients in intensive care. The challenge is to create this using data, i.e., decide on the source-destination and direction of each edge based on data.

http://www.bnlearn.com/bnrepository/

Key Questions on Datasets using Iris Dataset as an Example

1. Classification: A new iris flower is found and all 4 measurements are taken; given the training data with labels, how to tell whether this new flower is a setosa, versicolor or virginica? … Supervised learning
2. Clustering: Some iris flowers were found and it is not known which of the three different categories they belong to (no knowledge of the labels). How to group these flowers into *k* distinct groups (involves determining *k*, as well)… Unsupervised learning
3. Dimensionality Reduction/Feature Engineering: Using the sample data that is collected from a large number of iris flowers (all three types) and assuming the labels, find the minimum features to store.

Similar Questions for other Datasets; More on this Later

Get to Know the Data

- Continuous Features: For each feature, tabulate
  - Count
  - % Missing
  - Cardinality
  - Minimum and Maximum
  - 1st and 3rd Quartiles
  - Mean and Median
  - Variance and Standard Deviation
  - Skewness
  - Kurtosis
- Discrete (Categorical) Features
  - Count
  - % Missing
  - Cardinality
  - First and Second Mode, Mode Frequency and % Mode Frequency

Data Statistics: Mean

- Data Matrix X: an N by p matrix
  - N = Number of data points (data samples)
  - p = Number of features (e.g., attributes, measurements, variables)
- Central Tendencies
  - Mean of each column feature (MATLAB: `mean(X)`)

$$\underline{\hat{\mu}} = \frac{X^T \underline{e}}{N}; \quad \underline{e} \sim N \text{ column vector of } 1^s$$

  - Iris Dataset:
    - Overall mean: $\underline{\mu}^T$ = [5.8433 3.0573 3.7580 1.1993]
    - Setosa: μ_1^T = [5.0060 3.4280 1.4620 0.246]
    - Versicolor: μ_2^T = [5.9360 2.7700 4.2600 1.3260]
    - Virginica: μ_3^T = [6.5880 2.9740 5.5520 2.0260]
  - Setosa has smaller petal length and petal width. Easy to classify
  - Versicolor has smaller petal width than Virginica; somewhat harder to distinguish between Versicolor and Virginica

Mean Removed Data Matrix

$$X_1 = X - \underline{e}\,\underline{\hat{\mu}}^T = \underbrace{(I - \frac{\underline{e}\,\underline{e}^T}{N})}_{\text{Projection Matrix},P} X$$

$$Note: P^n = P; n = 2,3,..$$

Data Statistics: Median

- Central Tendencies
  - Median (`median(X)`)
    - Sort the values in each column j in increasing order: $x_{[1],j} \leq x_{[2],j} \leq .... \leq x_{[N],j}$
    - If N is odd

$$Md_j = x_{[\frac{N+1}{2}],j}$$

    - If N is even

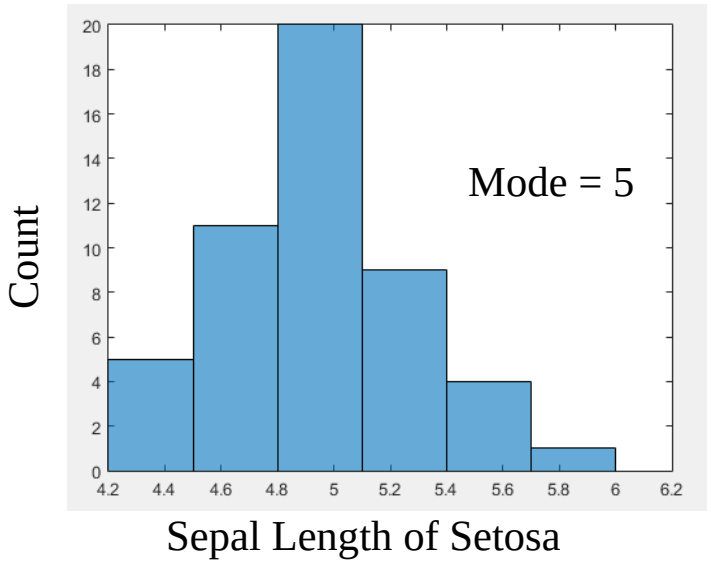
$$Md_j = \frac{1}{2}[x_{[\frac{N}{2}],j} + x_{[\frac{N}{2}+1],j}]$$

  - Iris Dataset
    - Overall Median: [5.8000 3.0000 4.3500 1.3000]
    - Setosa: [5.0000 3.4000 1.5000 0.2000]
    - Versicolor: [5.9000 2.8000 4.3500 1.3000]
    - Virginica: [6.5000 3.0000 5.5500 2.0000]
  - Similar conclusions as with mean

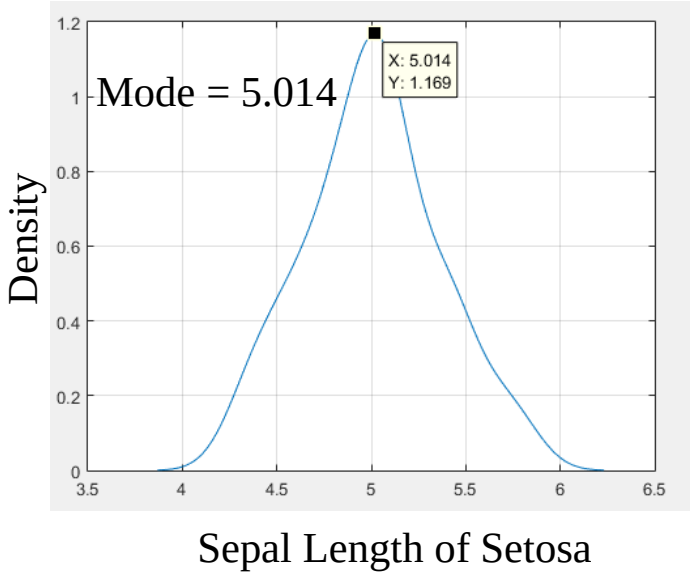
Data Statistics: Mode

- Central Tendencies
  - Mode of each feature ≡ most frequent value in each column
    - Good for discrete features: (`mode(X)`)
    - For continuous features, do a histogram or density estimate first and calculate the *peak* of the estimate (`histogram(X(:,j))` or `ksdensity(X(:,j))`; `j`=1,2,..,*p*)

Histogram



Density



Data Statistics: Variations -1

- Variations
  - Minimum: (`min(X)`)

$$x_{j,\min} = \min_{1 \le i \le N} \left(x_{ij} \right); \quad j = 1, 2, .., p$$

  - Maximum: (`max(X)`)

$$x_{j,\max} = \max_{1 \le i \le N} \left(x_{ij} \right); \quad j = 1, 2, .., p$$

  - Range: (`range(X)`)

$$r_j = x_{j,\max} - x_{j,\min}; \quad j = 1, 2, .., p$$

  - Sample Variance for *iid* samples: (`var(X)`)

$$\hat{\sigma}_j^2 = \frac{1}{N-1} \sum_{i=1}^{N} (x_{ij} - \hat{\mu}_j)^2; \hat{\mu}_j = \frac{1}{N} \sum_{i=1}^{N} x_{ij}; \quad j = 1, 2, .., p$$

Why? → You can also compute class-conditional variance

iid = independent and identically distributed

$Bernoulli: \sigma_j^2 = p(1-p)$

$P(x=1) = p$

$Show: E[\hat{\sigma}_j^2] = \sigma_j^2$

$\sigma_j^2 = \text{var}(x_{ij})$

  - An unbiased estimator of the covariance matrix (class-conditioned)

$$\hat{\Sigma} = \frac{X_{1c}^T X_{1c}}{n_c - 1} = \frac{X_c^T (I - \frac{e e^T}{n_c}) X_c}{n_c - 1}$$

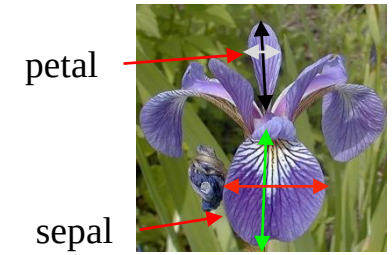
Computed from data of each class

For common (class-independent) covariance matrix across C classes, replace $(N\text{-}1)$ by $(N\text{-}C)$ for unbiasedness! See lectures 4-6.

Data Statistics: Variations - 2

- Variations
  - Correlation Coefficient Matrix (*corrcoef*(*X*))

$$C_r = \left[Diag(\hat{\Sigma}) \right]^{-1/2} \hat{\Sigma} \left[Diag(\hat{\Sigma}) \right]^{-1/2}$$



http://www.statlab.uni-heidelberg.de/data/iris/

  - Iris Dataset

$$\hat{\Sigma}_{versicolor} = \begin{bmatrix} 0.2664 & 0.0852 & 0.1829 & 0.0558 \\ 0.0852 & 0.0985 & 0.0827 & 0.0412 \\ 0.1829 & 0.0827 & 0.2208 & 0.0731 \\ 0.0558 & 0.0412 & 0.0731 & 0.0391 \end{bmatrix} \quad C_{r,versicolor} = \begin{bmatrix} 1.0000 & 0.5259 & 0.7540 & 0.5465 \\ 0.5259 & 1.0000 & 0.5605 & 0.6640 \\ 0.7540 & 0.5605 & 1.0000 & 0.7867 \\ 0.5465 & 0.6640 & 0.7867 & 1.0000 \end{bmatrix}$$

  - 1st and 3rd Quartiles of feature *j*
    - Sort data in increasing order: $x_{[1],j} \le x_{[2],j} \le .... \le x_{[N],j}$
    - The *lower half* of data: $x_{[1],j} \le x_{[2],j} \le .... \le x_{[\frac{N-1}{2}],j}$ for N odd; $x_{[1],j} \le x_{[2],j} \le .... \le x_{[\frac{N}{2}],j}$ for N even
    - The *upper half* of data: $x_{[\frac{N+1}{2}+1],j} \le x_{[\frac{N+1}{2}+2],j} \le .... \le x_{[N],j}$ for N odd; $x_{[\frac{N}{2}+1],j} \le x_{[\frac{N}{2}+2],j} \le .... \le x_{[N],j}$ for N even
    - The *first quartile* $Q_{1,j}$ is the median of the *lower half* of the data. This means that about 25% of the numbers in the data set lie below $Q_{1,j}$ and about 75% lie above $Q_{1,j}$
    - The *third quartile*, denoted $Q_{3,j}$, is the median of the *upper half* of the data set. This means that about 75% of the numbers in the data set lie below $Q_{3,j}$ and about 25% lie above $Q_{3,j}$.

For N=9: $Q_{1,j} = \frac{x_{[2],j} + x_{[3],j}}{2}; Q_{3,j} = \frac{x_{[7],j} + x_{[8],j}}{2}$

For N=10: $Q_{1,j} = x_{[3],j}; Q_{3,j} = x_{[8],j}$

Data Statistics: Skewness and Kurtosis

- Bias Corrected Skewness (*skewness*(*X*,0))

$$s_j = \left[\frac{\sqrt{N(N-1)}}{(N-2)}\right]\left[\frac{\frac{1}{N}\sum_{i=1}^{N}(x_{ij}-\hat{\mu}_j)^3}{\left(\frac{1}{N}\sum_{i=1}^{N}(x_{ij}-\hat{\mu}_j)^2\right)^{3/2}}\right]; \hat{\mu}_j = \frac{1}{N}\sum_{i=1}^{N}x_{ij};\ j=1,2,..,p; N \geq 3$$

$$Bernoulli: s_j = \frac{1-2p}{\sqrt{p(1-p)}}$$

$$P(x=1) = p$$

  - $s_j < 0 \Rightarrow$ left tail is longer; $s_j > 0 \Rightarrow$ right tail is longer; $s_j = 0 \Rightarrow$ symmetric
  - Skewness for versicolor: [0.1022 -0.3519 -0.5882 -0.0302]
  - Returns from financial stocks have larger-than-Normal skew and kurtosis
- Bias Corrected Kurtosis (*kurtosis*(*X*,0)) … useful for detecting outliers

$$\kappa_j = \left[\frac{(N-1)}{(N-2)(N-3)}\right]\left[(N+1)\left(\frac{\frac{1}{N}\sum_{i=1}^{N}(x_{ij}-\hat{\mu}_j)^4}{\left(\frac{1}{N}\sum_{i=1}^{N}(x_{ij}-\hat{\mu}_j)^2\right)^2}\right)-3(N-1)\right]+3; \hat{\mu}_j = \frac{1}{N}\sum_{i=1}^{N}x_{ij};\ j=1,2,..,p; N \geq 4$$

$$Bernoulli: \kappa_j = \frac{3p^2-3p+1}{p(1-p)}$$

  - $\kappa_j = 3 \Rightarrow$ Gaussian and *Mesokurtic* distributions; Excess Kurtosis = κ_j -3
  - $\kappa_j > 3 \Rightarrow$ *Leptokurtic* or *super-Gaussian* distributions (e.g., Logistic, Laplace, Student-t,...)
  - $\kappa_j < 3 \Rightarrow$ *Platykurtic* or *sub-Gaussian* distributions (e.g., Bernoulli with p=1/2, uniform)

Data Scaling Methods - 1

- For each Feature

1. *Zero mean and unit variance (Z-scores)*

$$x_{ij} \rightarrow s_{ij} = s_{ij}(x_{ij}) = \frac{(x_{ij} - \hat{\mu}_j)}{\hat{\sigma}_j}$$

$$In\ Matrix\ \ form: (X - \underline{e}\,\underline{\hat{\mu}}^T)\,Diag(1/\hat{\sigma}_j)$$

2. *Scale it to* [0,1] *or* [-1,1]

(Often times, [ε, 1- ε] or [-1+ε, 1-ε]) ε ≈ 0.15 ~ 0.20 to avoid the saturation of nonlinearities used in neural networks (e.g., sigmoid and tanh))

$$x_{ij} \rightarrow s_{ij} = s_{ij}(x_{ij}) = (1-2\varepsilon)\frac{(x_{ij} - x_{\min,j})}{(x_{\max,j} - x_{\min,j})} + \varepsilon \quad \Rightarrow [\varepsilon,\ \ 1-\varepsilon]$$

$$x_{ij} \rightarrow s_{ij} = s_{ij}(x_{ij}) = 2(1-\varepsilon)\frac{(x_{ij} - x_{\min,j})}{(x_{\max,j} - x_{\min,j})} + (\varepsilon - 1) \quad \Rightarrow [\varepsilon - 1,\ \ 1-\varepsilon]$$

Data Scaling Methods - 2

3. If small values are disproportionately close together and large values are spread out, use *log transformations* to obtain uniformized x^s

$$x_{ij} \rightarrow s_{ij} = s_{ij}(x_{ij}) = \ln\left[\frac{(x_{ij} - x_{\min,j})}{(x_{\max,j} - x_{\min,j})} + 1\right] \Big/ \ln 2$$

4. If large values are close together and small values are further apart, use *exponential transformations* (spreads out large values and pushes small values together)

$$x_{ij} \rightarrow s_{ij} = s_{ij}(x_{ij}) = \frac{1 - e^{-\left[\frac{(x_{ij} - x_{\min,j})}{(x_{\max,j} - x_{\min,j})}\right]}}{1 - e^{-1}}$$

- Coding Classes (Targets): Coding [1,2,..,C] is okay for some (e.g., decision trees)

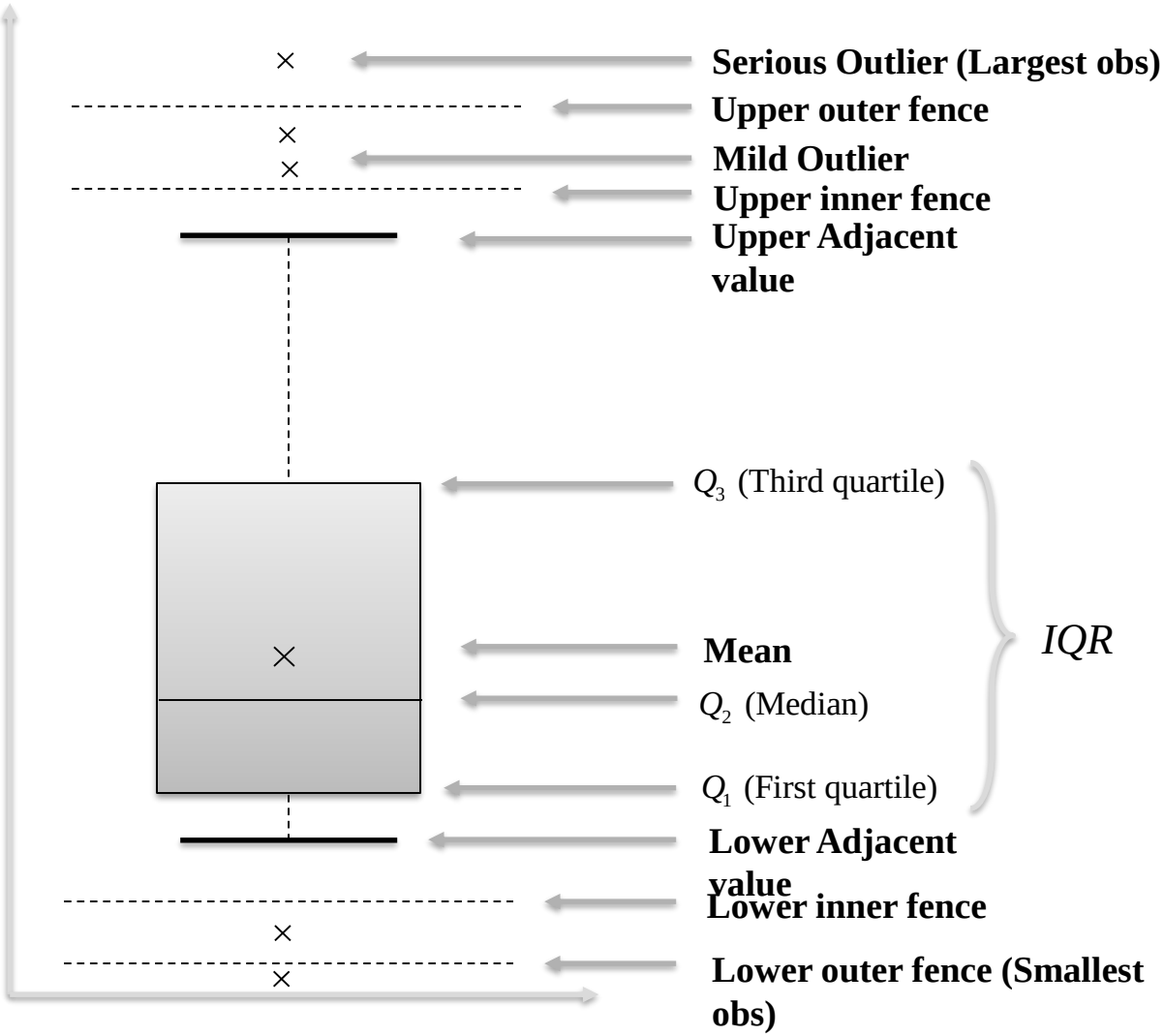
$$\underline{z}_k = \underline{e}_k = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \Rightarrow z = k \quad \text{or} \quad \underline{z}_k = \begin{bmatrix} -1/(C-1) \\ \vdots \\ 1 \\ \vdots \\ -1/(C-1) \end{bmatrix} \quad \text{or} \quad \underline{z}_k = \begin{bmatrix} -1 \\ \vdots \\ 1 \\ \vdots \\ -1 \end{bmatrix}$$

Good for MLP

Data Visualization: Box Plot

- Display data distribution and identify possible outliers
- Box plot = *a five-number summary of scalar data*
  - Median
  - The first quartile Q_1, the third quartile Q_3 $(IQR = Q_3 - Q_1)$
  - The upper adjacent value: $Q_3 + 1.5 \times IQR$
  - The lower adjacent value: $Q_1 - 1.5 \times IQR$
  - $Q_3 + 1.5 \times IQR$ is called the upper inner fence and $Q_1 - 1.5 \times IQR$is called the lower inner fence
- Outliers based on Box Plots
  - *Mild Outliers*: Any observations that lies in the interval $[Q_1 - 3 \times IQR, Q_1 - 1.5 \times IQR)$ or are in the interval $(Q_3 + 1.5 \times IQR, Q_3 + 3 \times IQR]$
  - *Serious Outliers*: Any observations that exceed $Q_3 + 3 \times IQR$ or are lower than $Q_1 - 3 \times IQR$

Interpretation of Box Plot



Other Simple ways of Detecting Outliers

- Visualization tools: Matrix plots, Box plots
- Z-score (assume Gaussian distribution): $Z_i = \frac{x_i - \mu}{\sigma}$
  - When not follow Gaussian distribution, could apply transformations
  - Usually, if $|Z_i| \geq 3$, possible outlier
  - Z-scores are not satisfactory for outlier labeling, especially when *n* is small → absolute value of a Z-score is at most $(n-1)/\sqrt{n}$ (Ronald Shiffler in American Statistician, Vol. 42, No. 1, Feb. 1988, pp. 79-80)
- Modified Z-Score

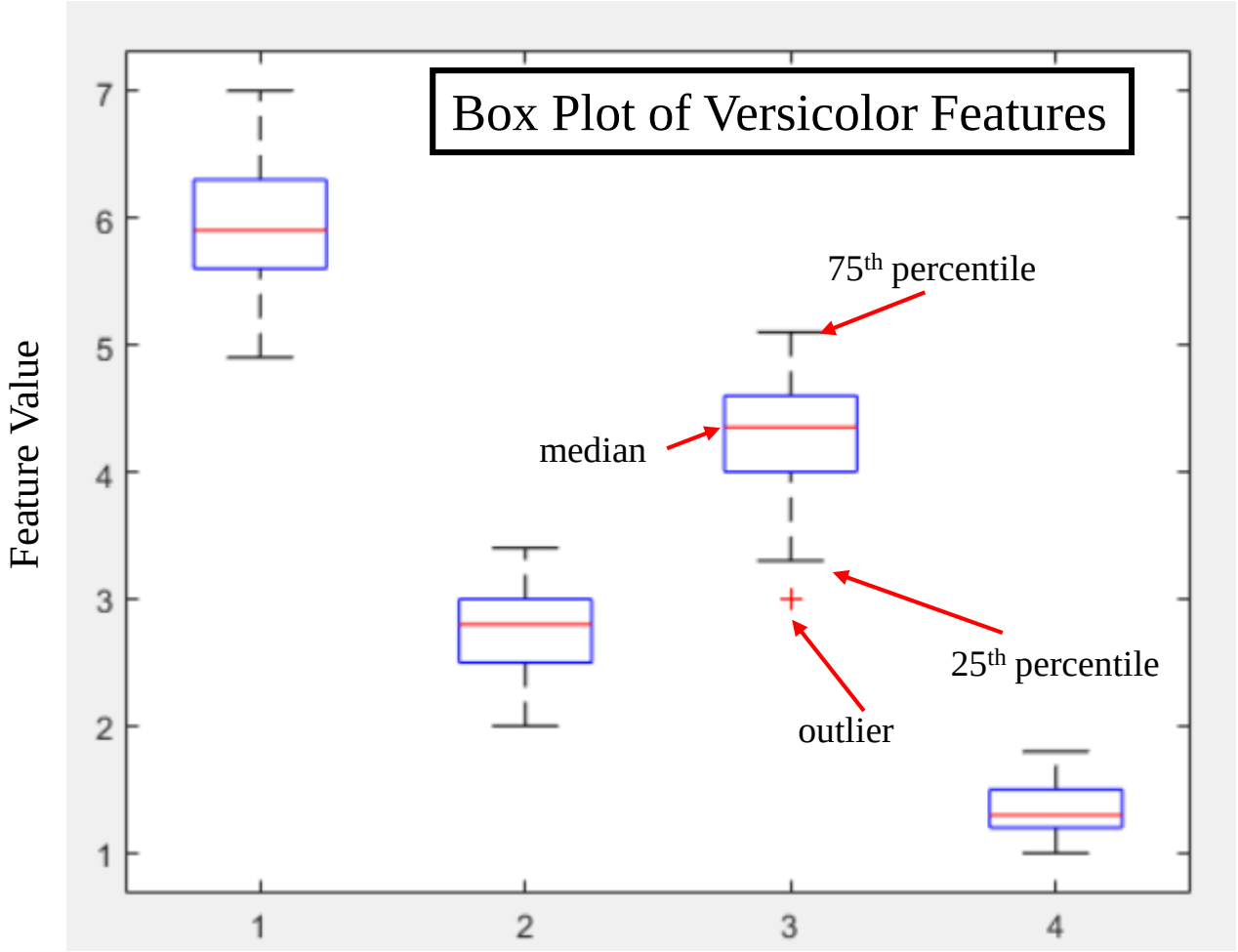
$$M_i = \frac{0.6745(x_i - \tilde{x})}{MAD}; \quad MAD = median\{|x_i - \tilde{x}|\}; \quad \tilde{x} - \text{sample median}$$

  - Usually, if $|M_i| \geq 3.5$, possible outlier
  - More robust than the standard Z score because it relies on the median, less influenced by outliers
  - If *MAD=0*, use

$$M_i = \frac{x_i - \tilde{x}}{1.253314 * \text{MeanAD}}; \quad \text{MeanAD} = \frac{1}{n}\sum_{i=1}^{n} |x_i - \bar{x}|$$

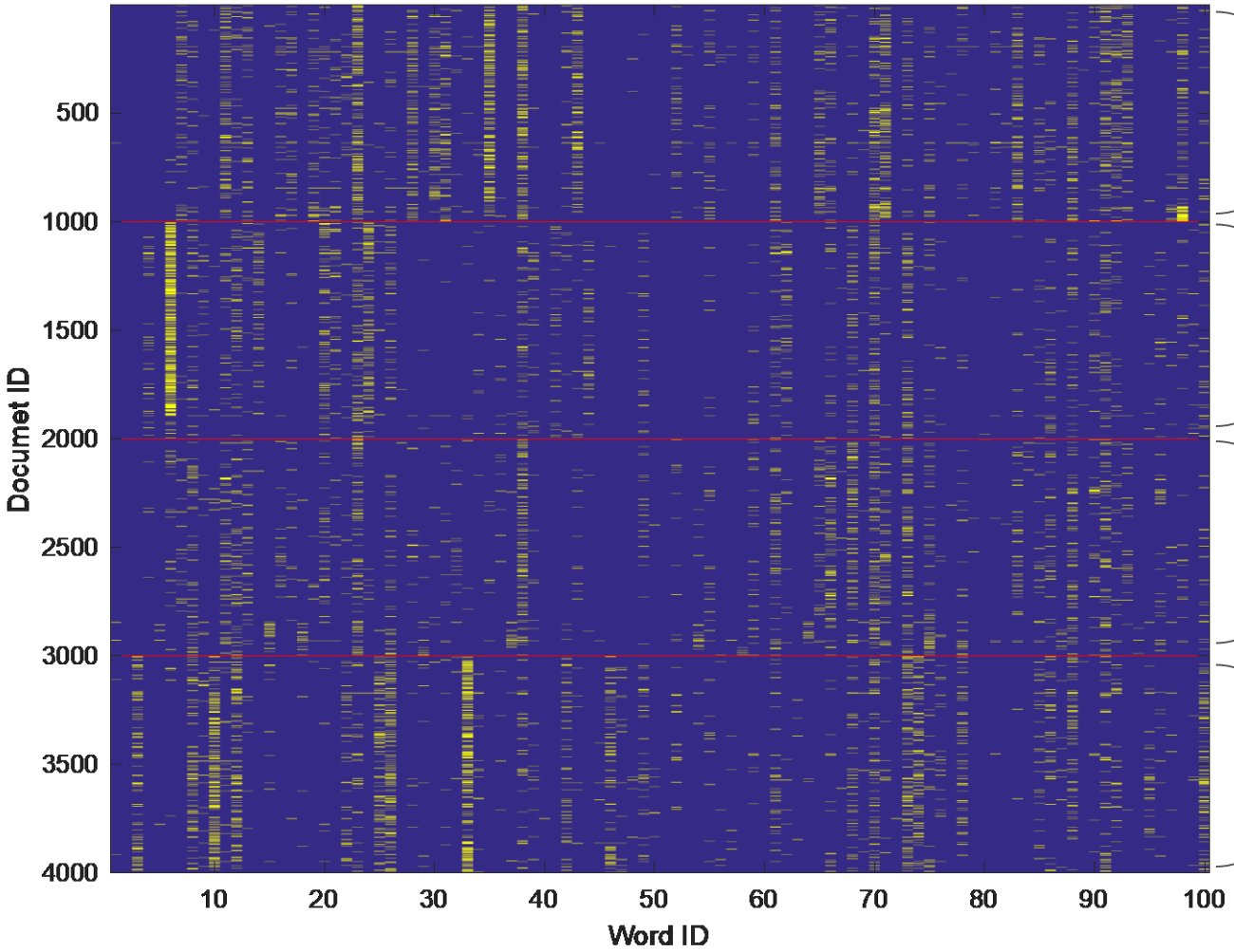
More sophisticated outlier detection methods as we learn new ML methods

Side-by-Side Box Plots of Versicolor Features



	Sepal Length	Sepal Width	Petal Length	Petal Width
Bias-corrected Kurtosis:	[2.4670	2.6338	3.0479	2.5899]

Matrix Visualization of 20 Newsgroup Dataset



Data in four different news groups are shown (see previous slide for description.)

Data Transformation/ Reduction/Cleaning

- Principal Component Analysis (PCA) is the best known and oldest technique for data reduction (Pearson, 1901)
  - Singular Value Decomposition (SVD) if done on data directly
  - Data compression and feature (indicator) selection

Choose a small number of linear combination of features based on their ability to reproduce the entire set of variables

- Data Matrix X: an N' by p matrix
  - N' = Number of data points (data samples)
  - p = Number of features (e.g., attributes, measurements)
- Remove *NaN* (Not a Number) elements from the matrix X
  - Use MATLAB function: `X(any(isnan(X),2),:) = [];`
  - N = Number of data points after removing *NaN* elements
  - New X matrix: an N by p matrix
- Compute the mean of each column feature

$$\underline{\hat{\mu}} = \frac{X^T \underline{e}}{N}; \quad \underline{e} \sim N \text{ column vector of } 1^s$$

Principal Component Analysis Explained

- Remove mean from each column and form a new data matrix X_1

$$X_1 = X - \underline{e}\,\underline{\hat{\mu}}^T = (I - \frac{\underline{e}\,\underline{e}^T}{N})X$$

- Compute the Covariance Matrix

$$P = \frac{X_1^T X_1}{N-1} = \frac{1}{N-1}\left[X^T X - N\,\underline{\hat{\mu}}\,\underline{\hat{\mu}}^T\right] = \frac{1}{N-1}\left[\sum_{n=1}^{N}(\underline{x}_n \underline{x}_n^T - \underline{\hat{\mu}}\,\underline{\hat{\mu}}^T)\right];\ \text{a } p \text{ by } p \text{ matrix}$$

- Perform Eigen decomposition of P and select the Eigen vectors corresponding to the top k eigen values ∋

$$\frac{\sum_{i=1}^{k}\lambda_i}{trace(P)} > Th \quad (0.95)$$

$$P = V\Lambda V^T;$$
$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k \geq \lambda_{k+1} \geq \cdots \geq \lambda_p$$

- Reduced Data Matrix

$X_2 = X_1 V_k$ an $N \times k$ matrix

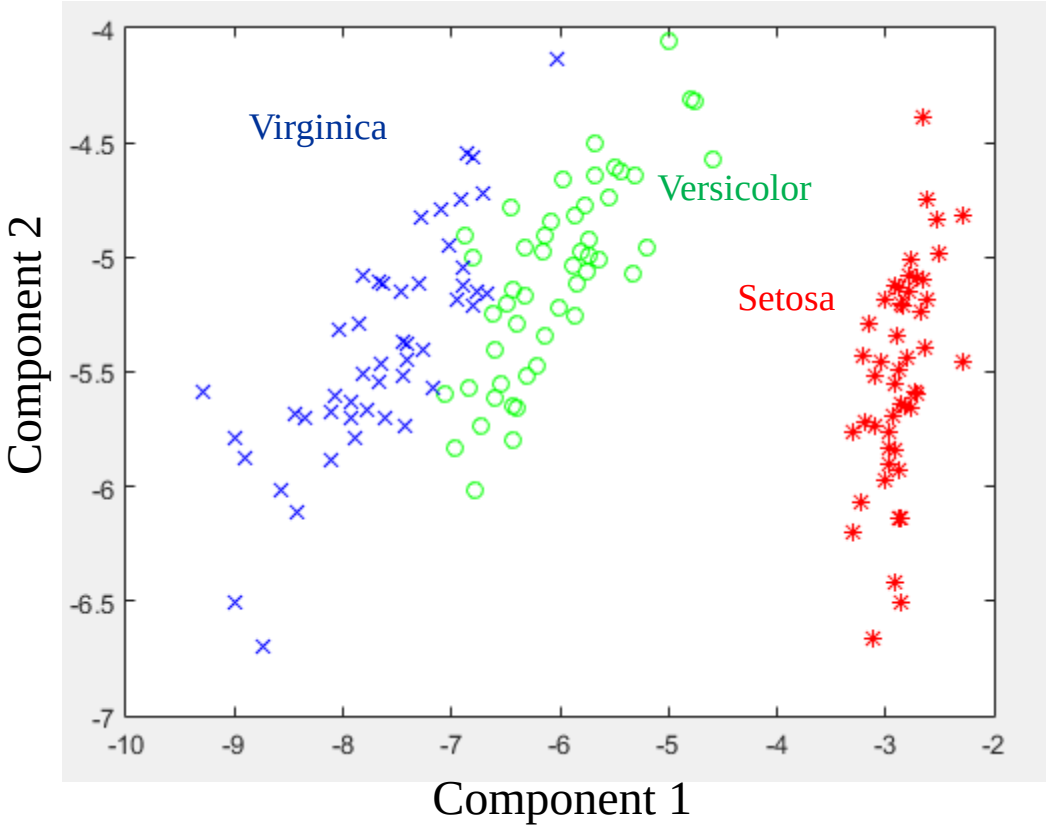
$V_k =$ first k columns of V; a $p \times k$ matrix

- Residual Matrix and Covariance

$$R = X_1[I_p - V_k V_k^T] \quad \text{an } N \times p \text{ matrix};\ \Sigma_r = [I_p - V_k V_k^T]P[I_p - V_k V_k^T] \text{ a } p \times p \text{ matrix}$$

Iris Data Visualization using PCA

- First principal component explains 92.46 % of variability in the data
- First 2 principal components explain 97.8% of variability in the data



- Setosa is easy to distinguish; some ambiguity between Versicolor & Virginica

Machine Learning Tasks

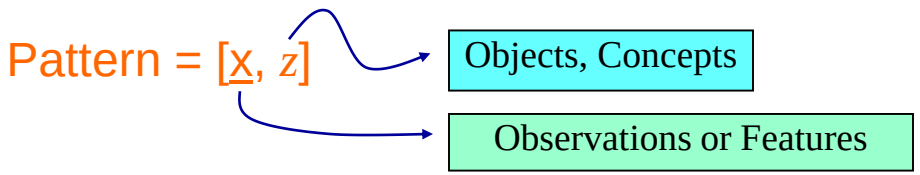
Machine Learning seeks to solve many generic tasks of interest.

1. Classification
2. Regression / Interpolation
3. Density Estimation
4. Forecasting / Prediction … basically regression
5. Optimization under uncertainty
6. Adaptive Control
7. Adaptive Communications

Problem 1: Pattern Classification

Pattern Classification

- **Pattern**: Pattern means something exhibiting certain regularities, something able to serve as a model, something representing the concept of what was observed. A pattern is a collection of observations connected in time or space or both.
- **Classification**: Assign an input pattern (e.g., speech waveform, handwritten character, etc.) represented by a feature vector $\underline{x} = [x_1\ x_2\ \ldots x_p]^T$ to one or more specified classes or concepts

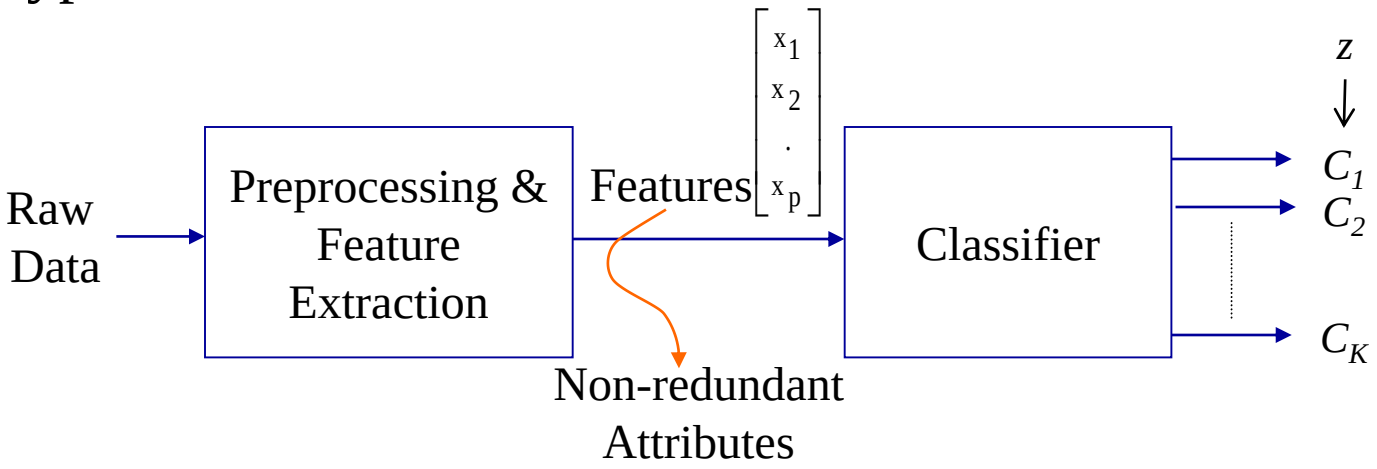


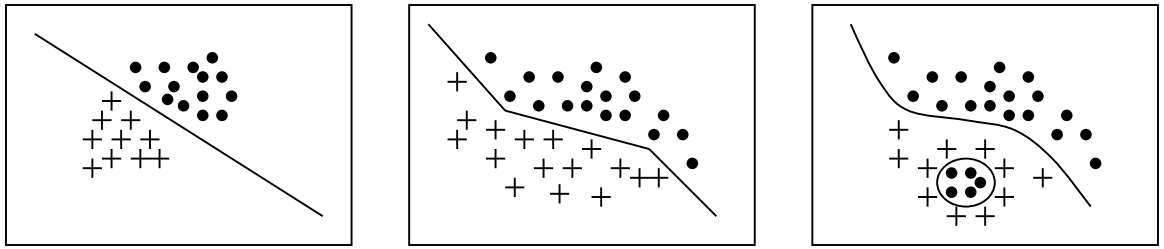
Pattern classification is the process of inferring z from $\underline{x}$: $\underline{x} \rightarrow z$

$\underline{x}$ can be documents, images, DNA sequences, or graphs,….
z can be classes, ranks, real values, part of speech tagging, graph,….

Pattern Classification Process

- Typical Process of Pattern Classification





IRIS Classification Results using MATLAB APP

Classifier	% Accuracy with 4 Features	% Accuracy with 2 PCA-Reduced Features	% Accuracy with 8 Features*	%Accuracy with 3 PCA-reduced Features*
Decision Tree	94.7%	93.3%	**96%**	92.7%
Weighted kNN	96.0%	96.7%	95.3%	**97.3%**
Linear SVM	96.0%	96%	94.7%	**98%**
Medium Gaussian SVM	96.7%	**97.3%**	95.3%	96.7%
Bagged Trees	95.3%	92%	96.7%	**97.3%**

- Add sepal and petal areas and ratios as 4 additional features

Overfitting is NOT Good!

From Duda, Hart and Stork

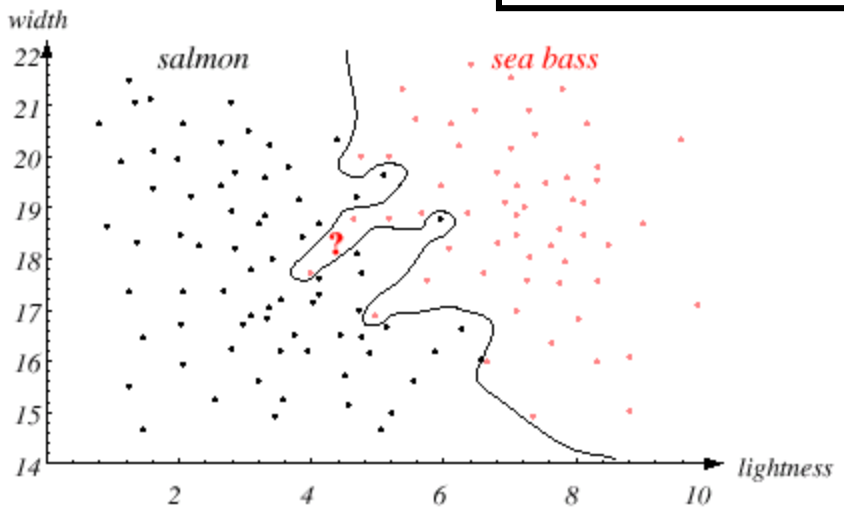


FIGURE 1.5. Overly complex models for the fish will lead to decision boundaries that are complicated. While such a decision may lead to perfect classification of our training samples, it would lead to poor performance on future patterns. The novel test point marked **?** is evidently most likely a salmon, whereas the complex decision boundary shown leads it to be classified as a sea bass. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

- **Tradeoff between overtraining and generalizability (accuracy in use)**
- **Complex classifiers <u>are not</u> necessarily the best ("Occam's Razor")**

How to Select the Right Decision Boundary?

From Duda, Hart and Stork

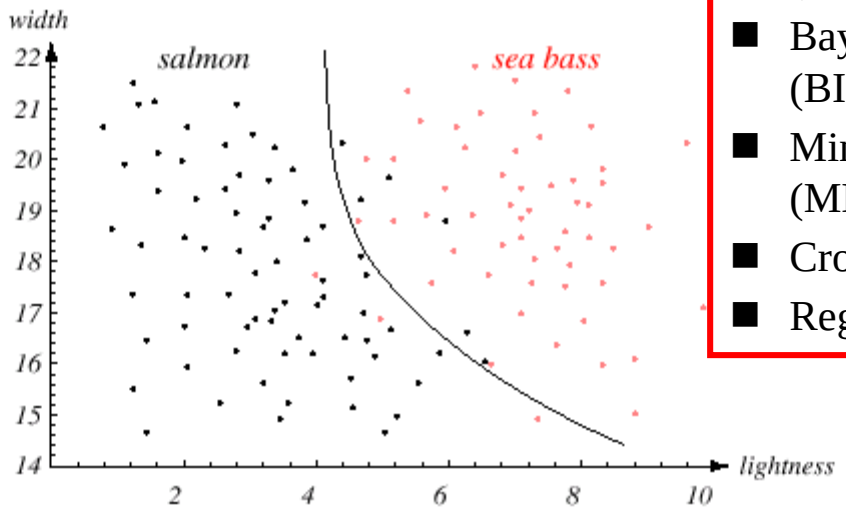


FIGURE 1.6. The decision boundary shown might represent the optimal tradeoff between performance on the training set and simplicity of classifier, thereby giving the highest accuracy on new patterns. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

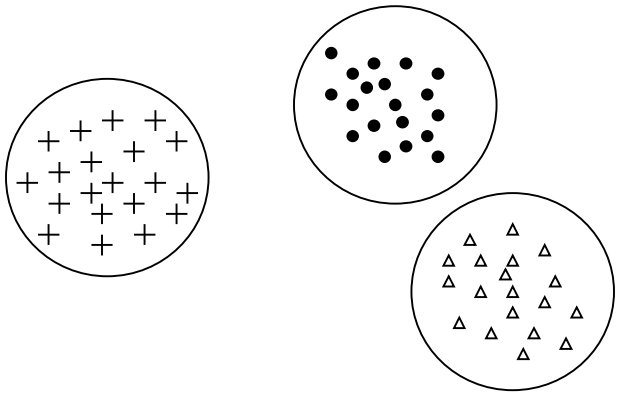
Model Selection Problem

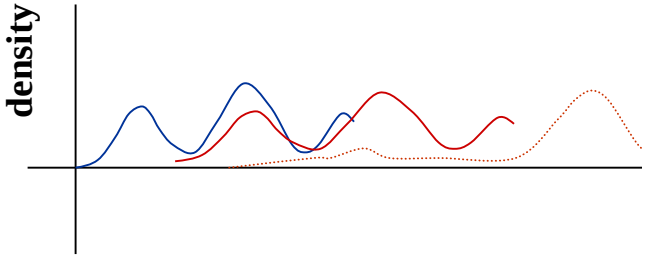
- Akaike Information Criterion (AIC)
- Bayesian Information Criterion (BIC)
- Minimum Description Length (MDL)
- Cross validation
- Regularization

- **Tradeoff between overtraining and generalizability**
- **How to find the "right" tradeoff?**

Problem 2: Clustering/Categorization - 1

- **Clustering / Categorization** -- also known as unsupervised classification
  - Explore similarity between the patterns and place similar things in a group
  - Derive a statistical model of the process
  - Density estimation





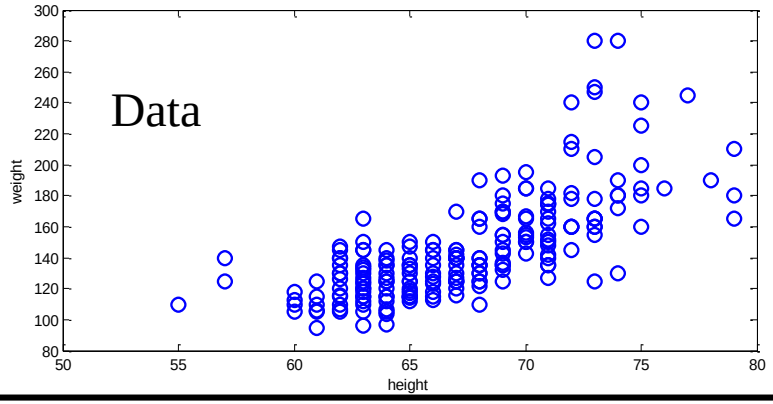
Is there a structure in the data?

Problem 2: Clustering Methods - 2

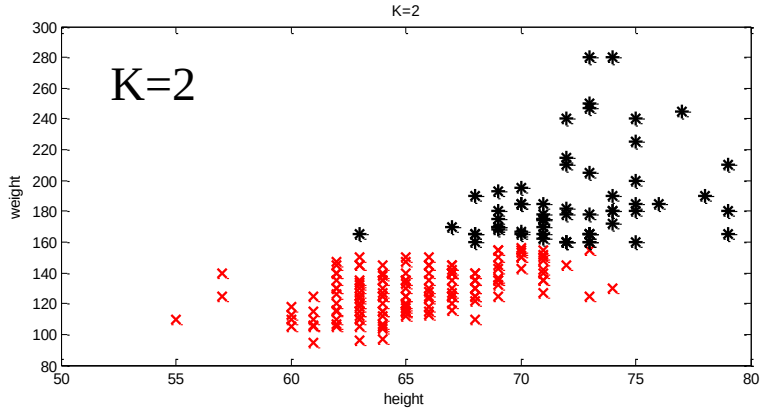
- Partitioning Algorithm
  - *K*-Means & *K*-Medoids Algorithm
  - Gaussian Mixture Models
    - Expectation Maximization and GMM
    - Variational Bayes and GMM
  - Fuzzy c-Means
- Hierarchical Clustering
  - Agglomerative
  - Divisive (Minimum Spanning Tree)
- Density Based Spatial Clustering
  - DBSCAN (Density-based Spatial Clustering of Applications with Noise)
  - Self Organizing Maps
- Spectral Clustering

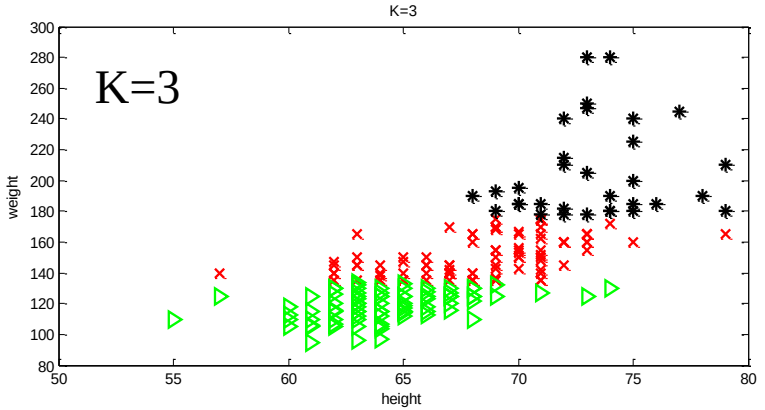
Problem 2: Clustering/Categorization - 3

- K-means clustering of height and weight data



Run `kmeansHeightWeight` in pmtk3-master\demos of Murphy to get these

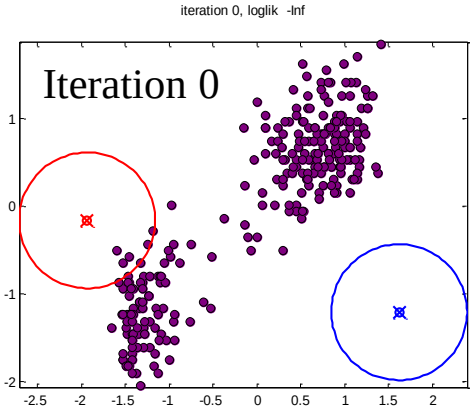


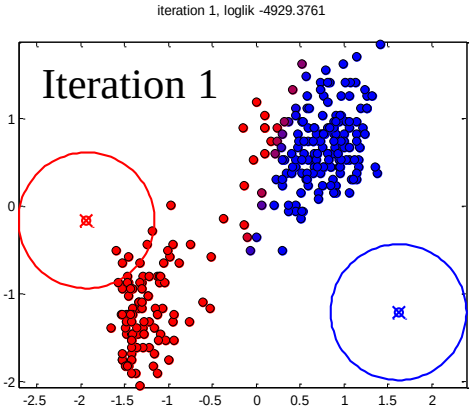


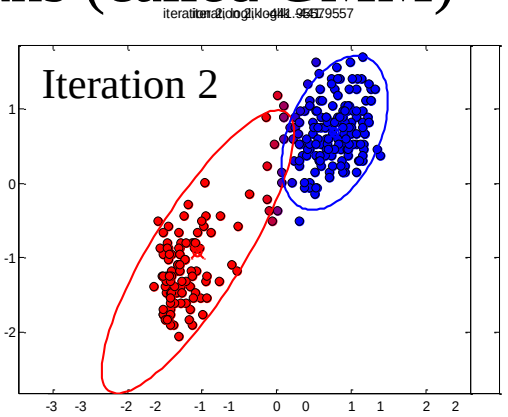
- How to select K? Model selection problem again!

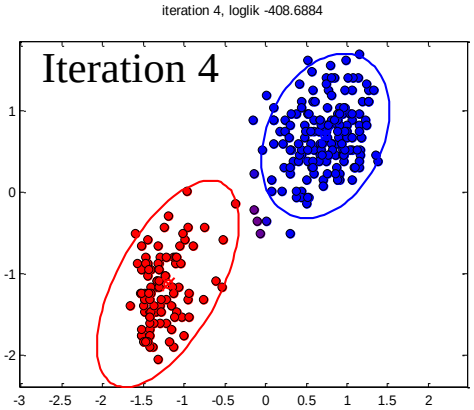
Problem 2: Clustering/Categorization - 4

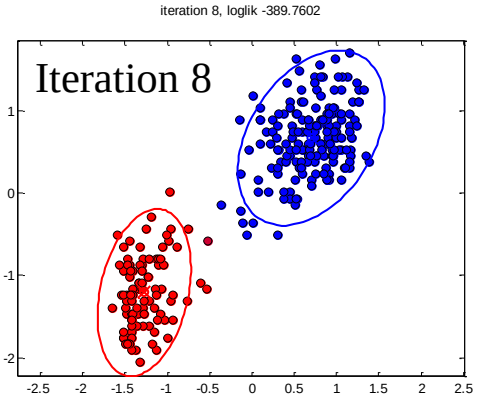
- How is clustering done? Iteratively by optimizing an objective function … Probabilistic version of K-means (called GMM)

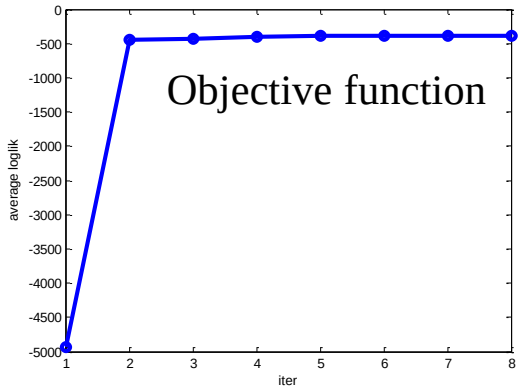










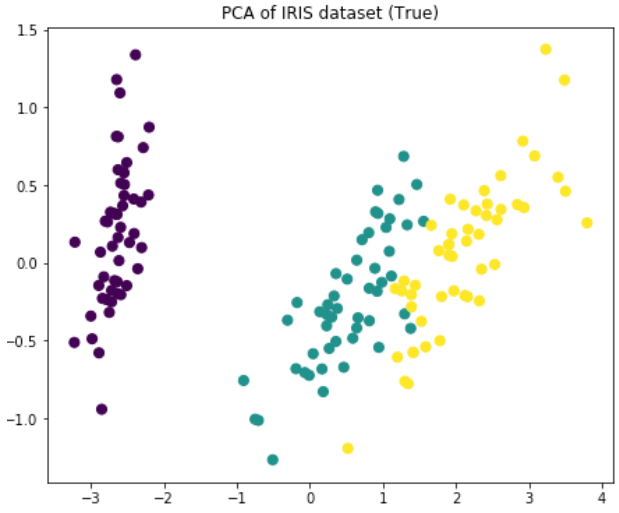


Old Faithful Data: Run `mixGaussDemoFaithful` in pmtk3-master\demos of Murphy to get these

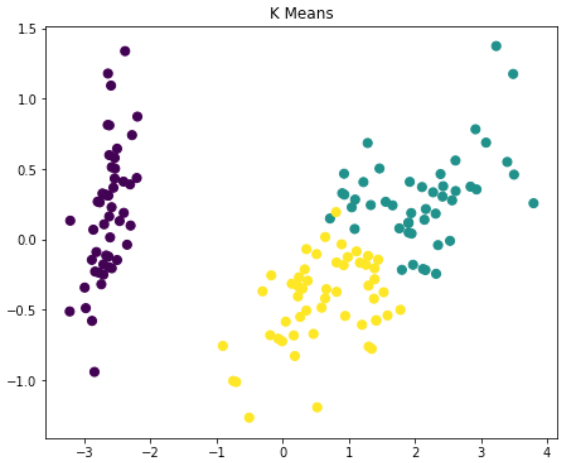
272 observations of waiting time between eruptions and the duration of the eruption for the Old Faithful geyser in Yellowstone National Park, Wyoming, USA.

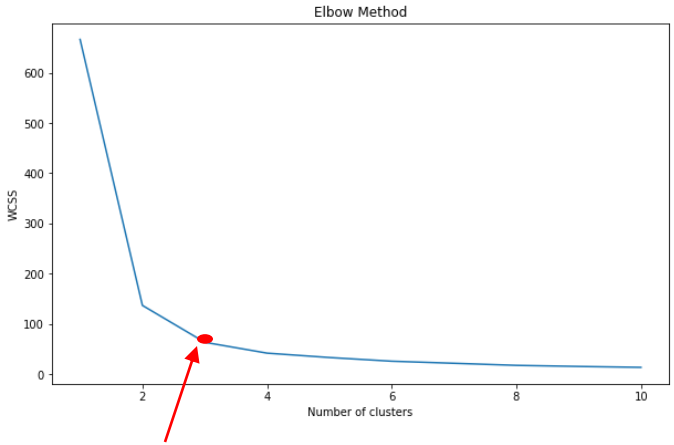
Problem 2: Clustering/Categorization - 5

PCA



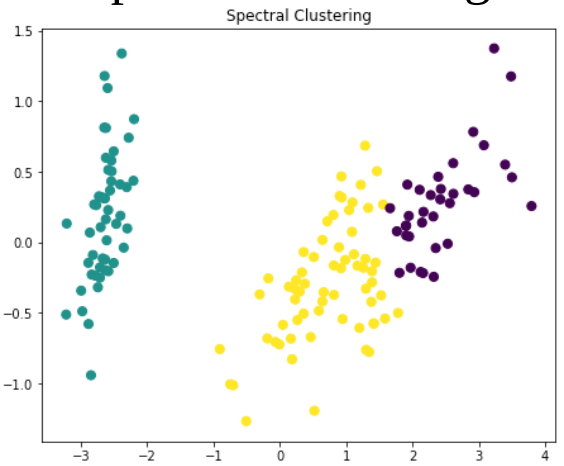
K-Means





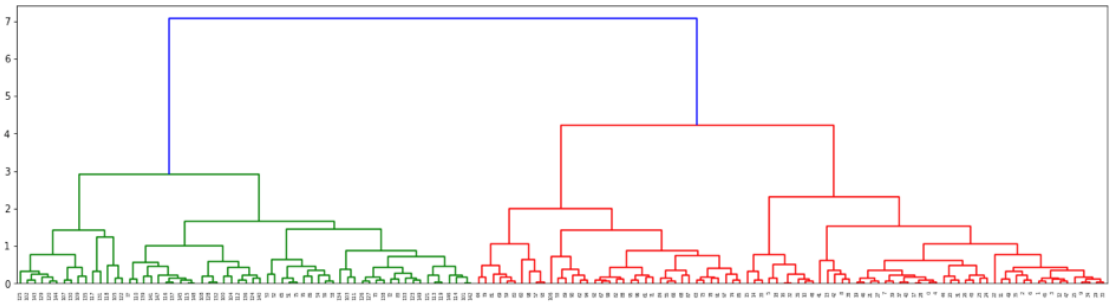
3 clusters

Spectral Clustering

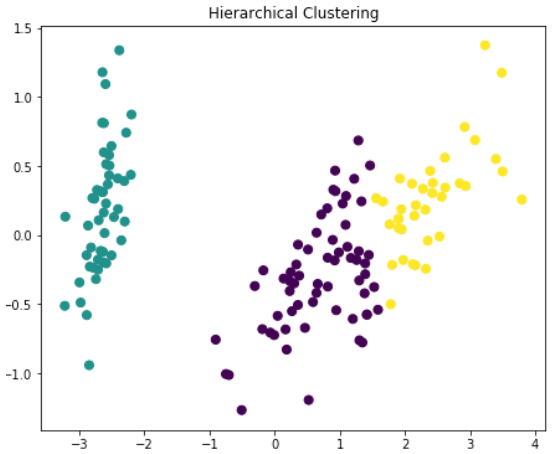


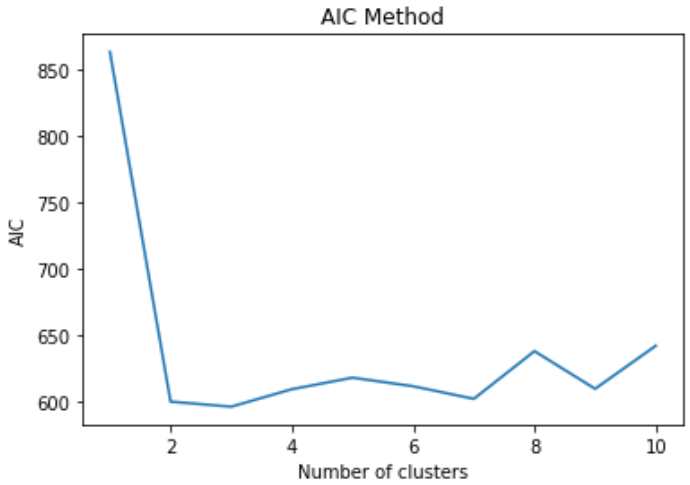
Problem 2: Clustering/Categorization - 6

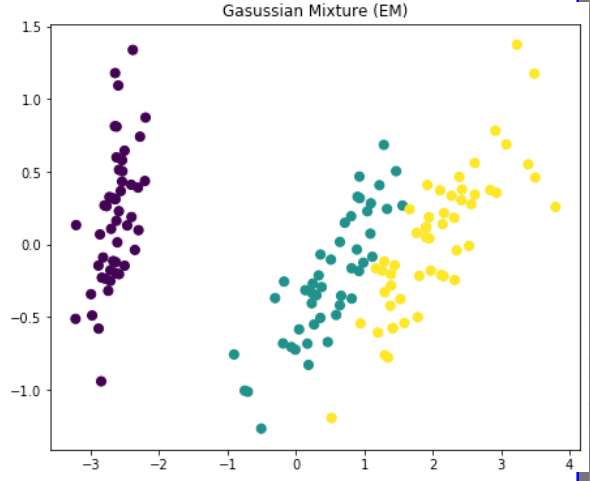
Dendrogram



Hierarchical Clustering







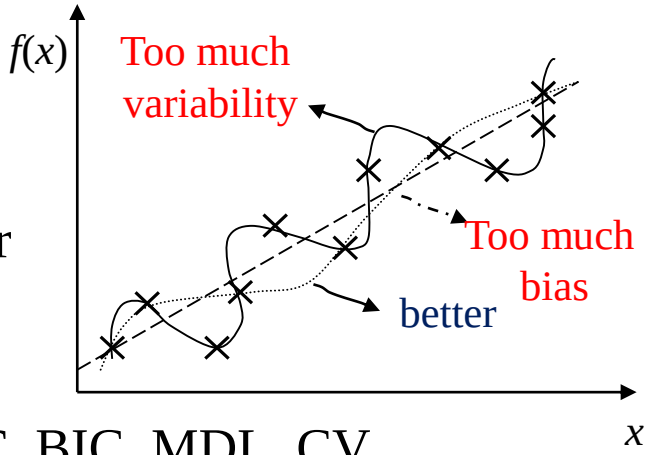
GMM

Problem 3: Function Approximation - 1

Function Approximation (Regression)

Given (x_1,z_1), (x_2,z_2) . . . ., (x_N,z_N), N data points

What is $z \approx f(x)$?



- Very low order models are not good
  $\Rightarrow$ too much bias
- Very high order models are not good either
  $\Rightarrow$ too much variability

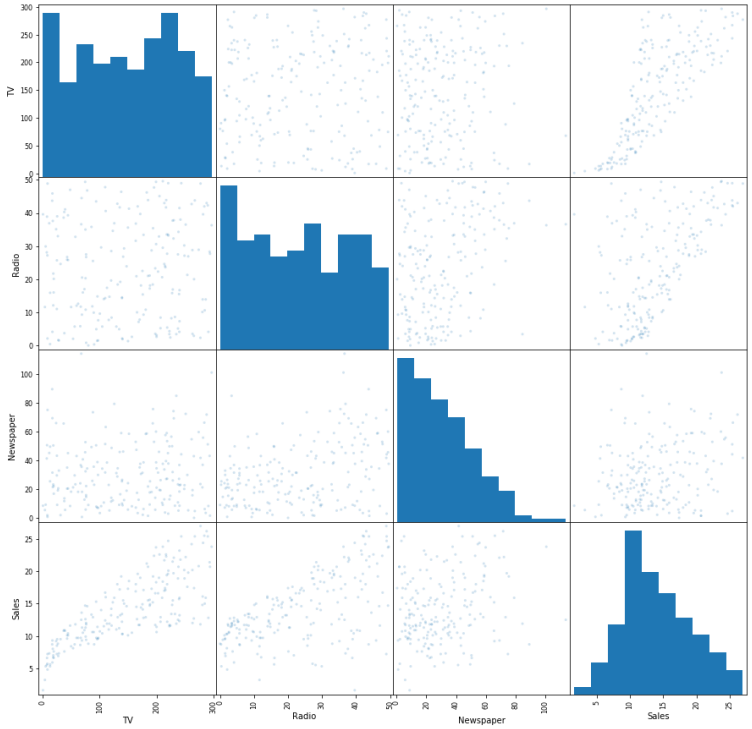
"*BIAS-VARIANCE DILEMMA*"

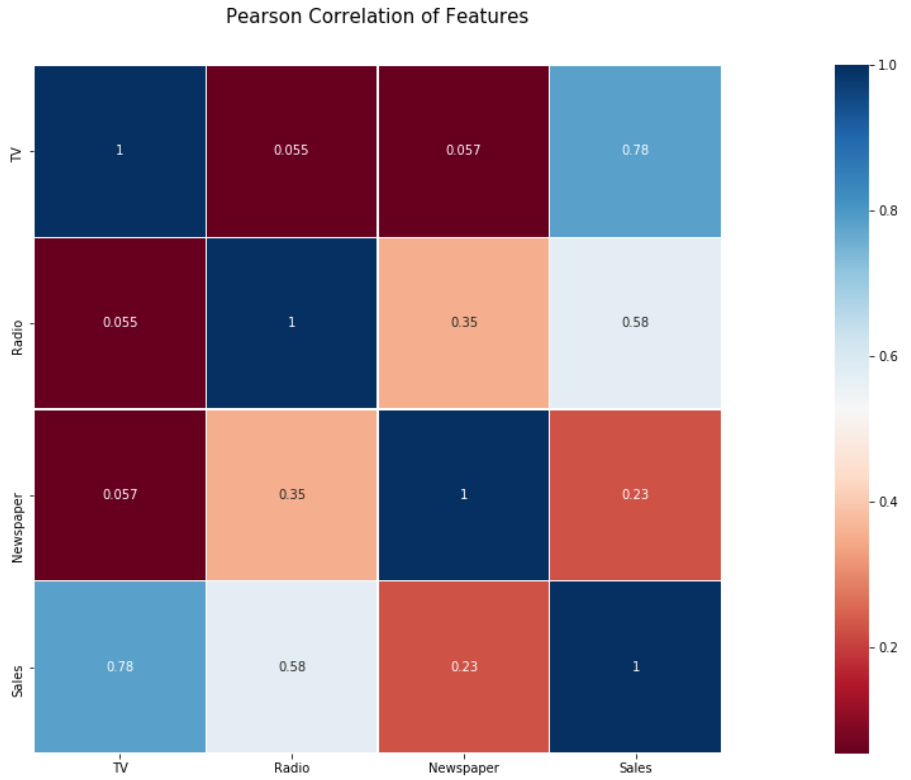
- How do we select the "right" model?? AIC, BIC, MDL, CV, ….
- Applications of function approximation
  - Regression
  - System Identification/Parameter Estimation
  - Neuro-dynamic Programming (NDP)/Approximate DP (ADP)
- What does $\approx$ mean? $\Rightarrow$ Criterion for optimization of weights

MSE: $\frac{1}{N}\sum_{i=1}^{N}[z_i - f(x_i)]^2$

This alone is not adequate! Why?

Problem 3: Advertising Example - 2

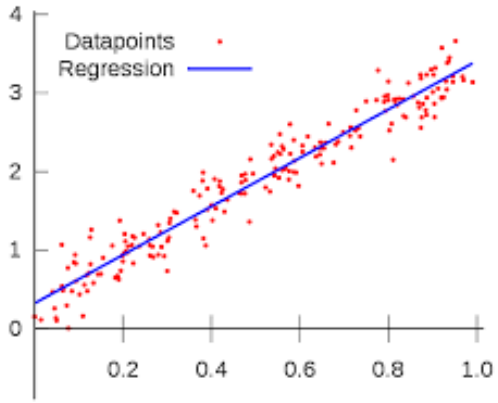




- Original Features: [TV, Radio, Newspaper]
- Transformed Feature: TV*Radio

Problem 3: Regression Example - 3

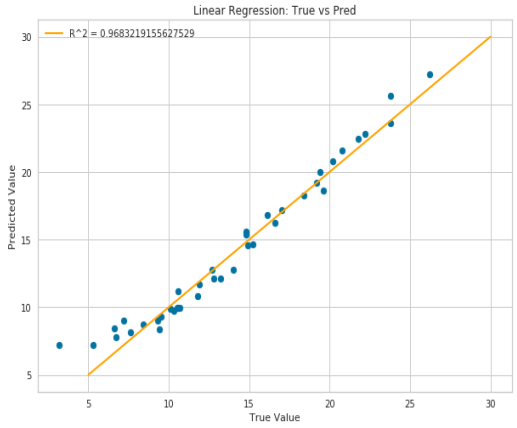
- Make predictions from data
  - Learn the relationship between features of data and continuous-valued response(s)
- **A Supervised Learning Method**

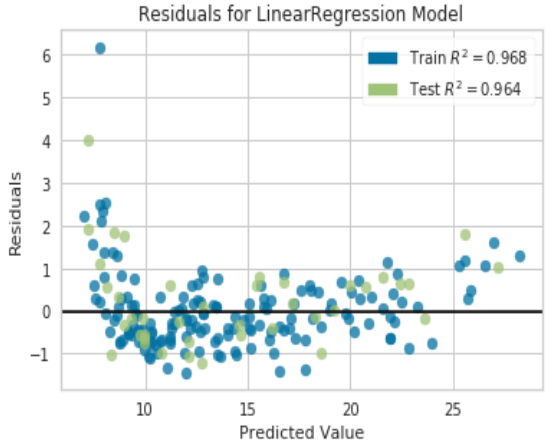


- Linear Regression
  - Least Squares Regression (LSR)
  - Lasso Regression (L_1)
  - Ridge Regression (L_2)
  - Elastic Net (Combination of L_1 and L_2)
- Generalized Linear Regression
- Multi-Layer Perceptron
- Kernel regression
  - Support Vector Machine (SVM & QP)
  - Radial Basis Function (RBF)
  - Gaussian Process Regression (GPR)
  - Relevance Vector Machine (RVM)
- Decision Trees

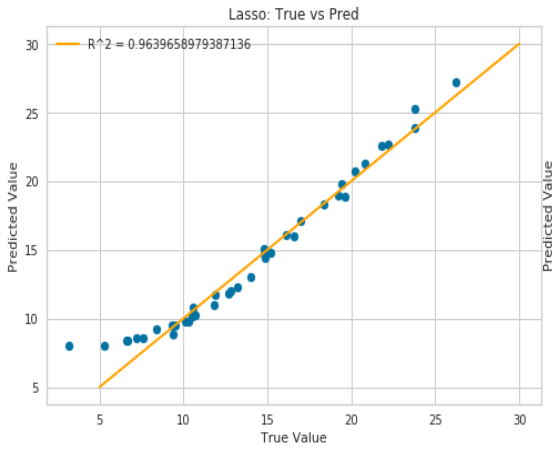
Problem 3: Regression Methods - 4

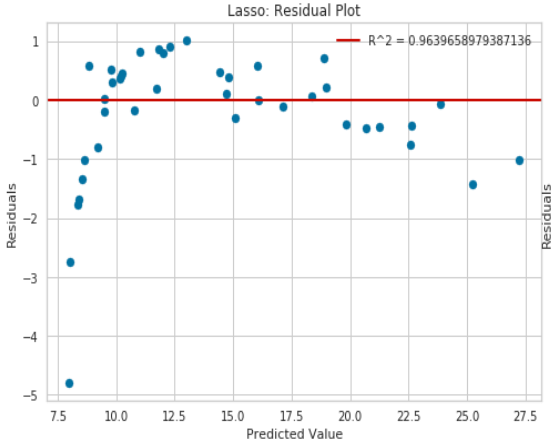
Linear Regression



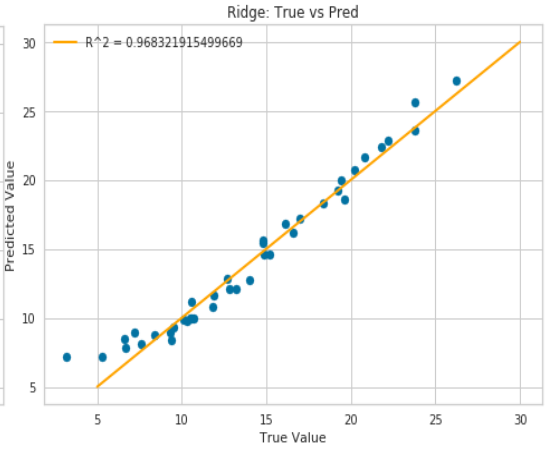


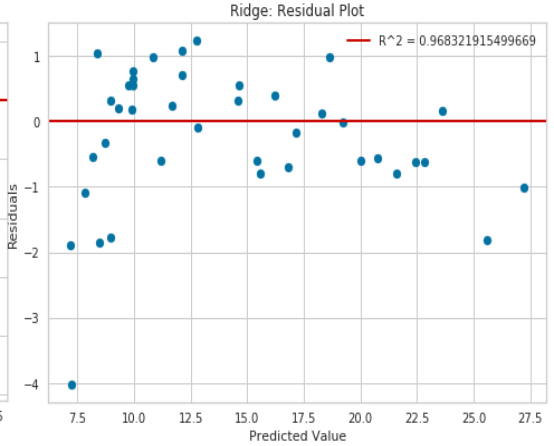
LASSO





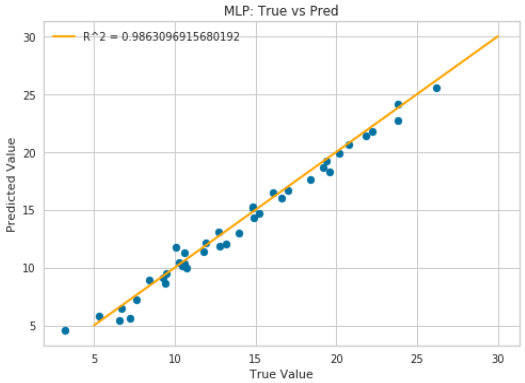
Ridge

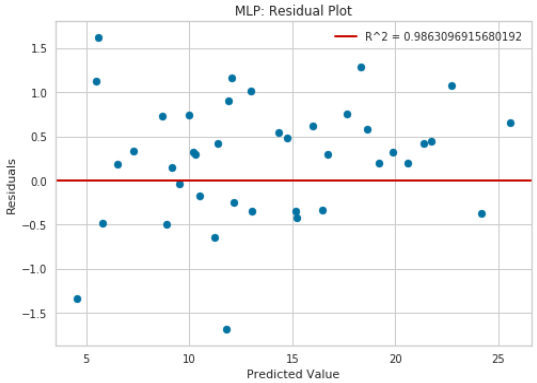




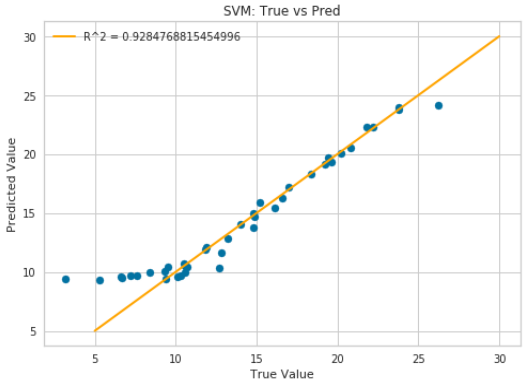
Problem 3: Regression Methods - 5

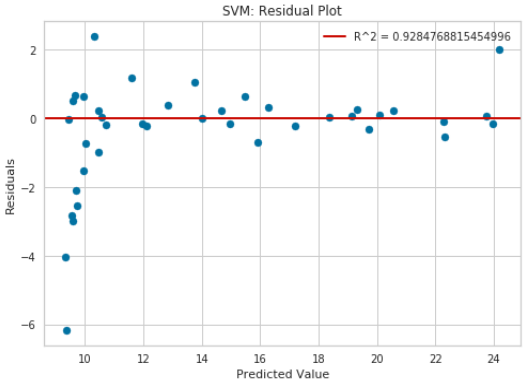
Multi-layer Perceptron (MLP)



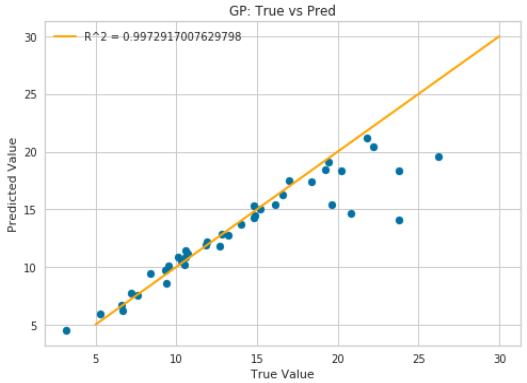


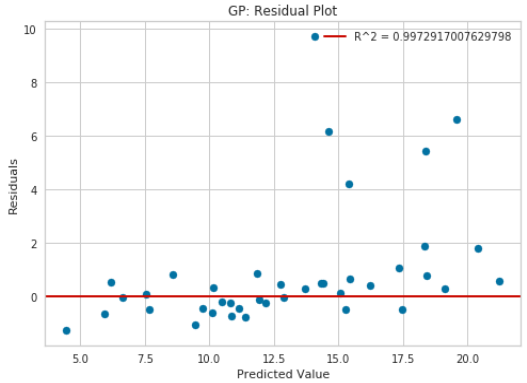
SVM





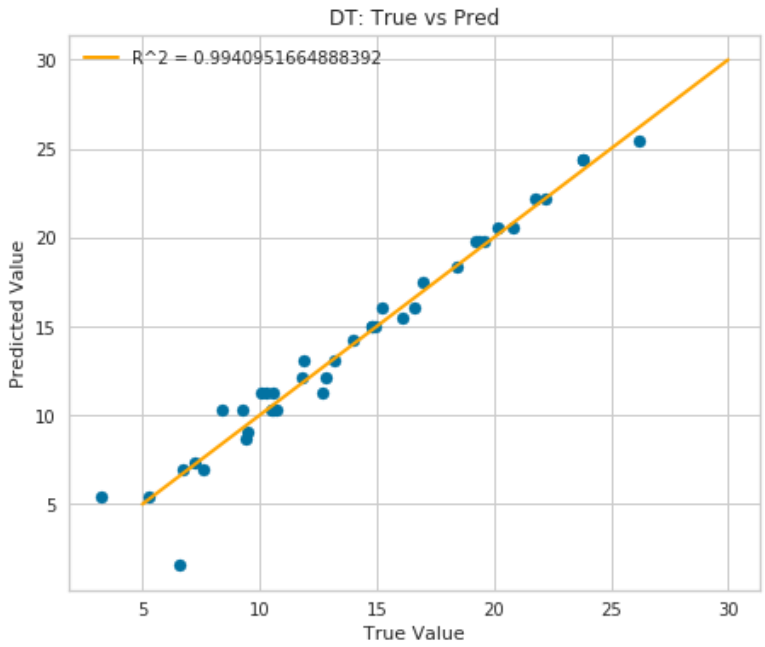
Gaussian Processes

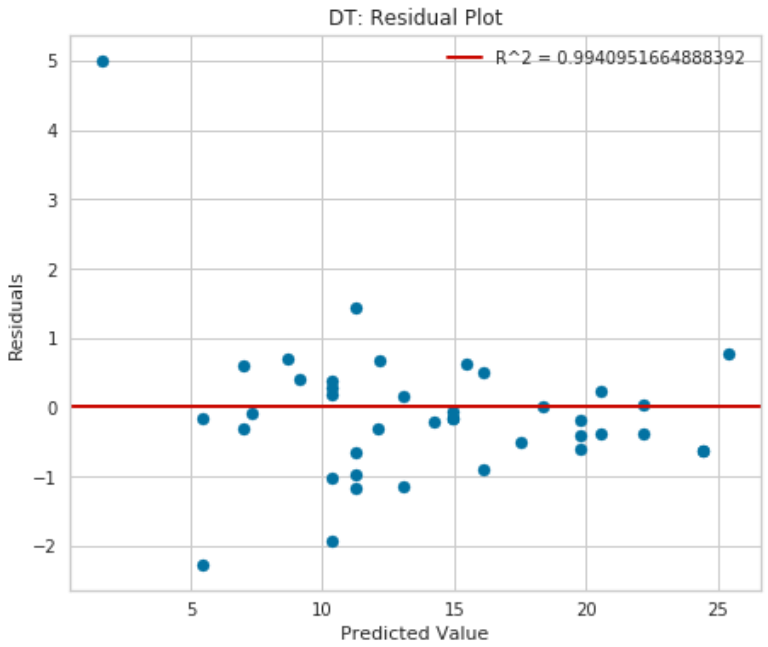




Problem 3: Regression Methods - 6

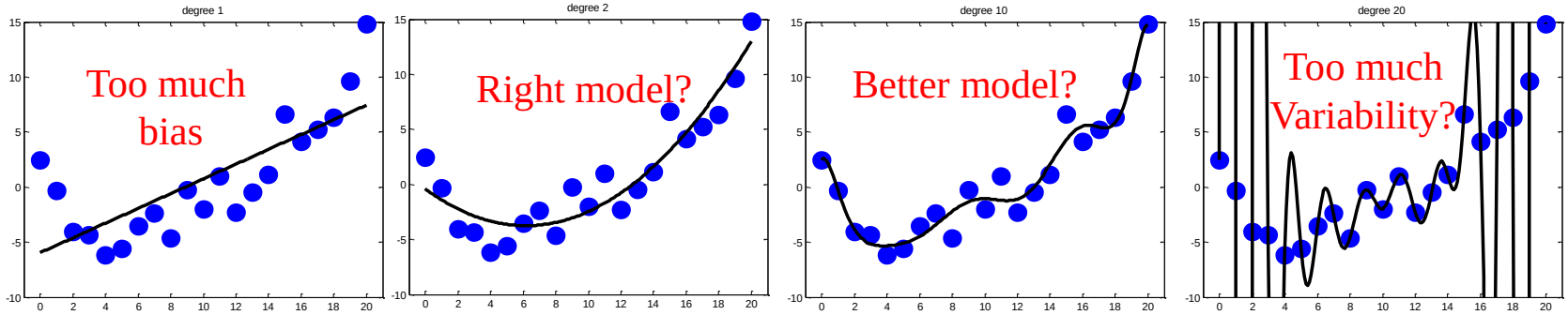
Decision Trees (DT)





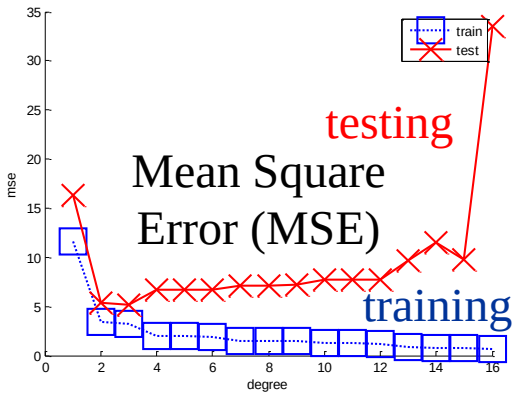
Problem 3: Function Approximation - 7

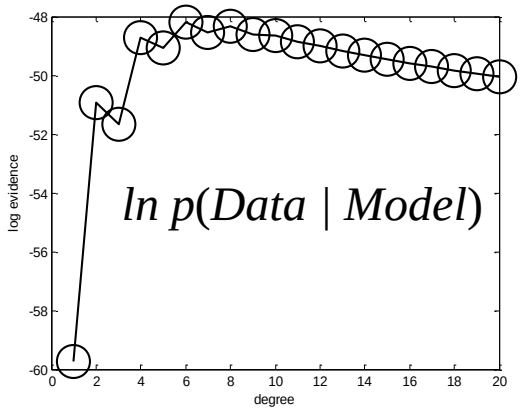
- What is the best polynomial for a 21 point noisy data?

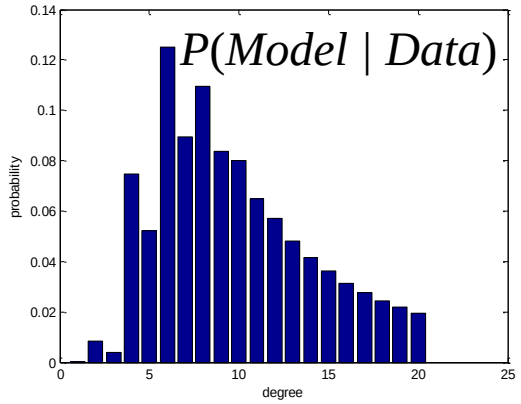


linear quadratic degree 10 degree 20

Run `linregPolyVsDegree` in pmtk3-master\demos of Murphy to get these







Model selection is a difficult problem!

Problem 3: Function Approximation - 8

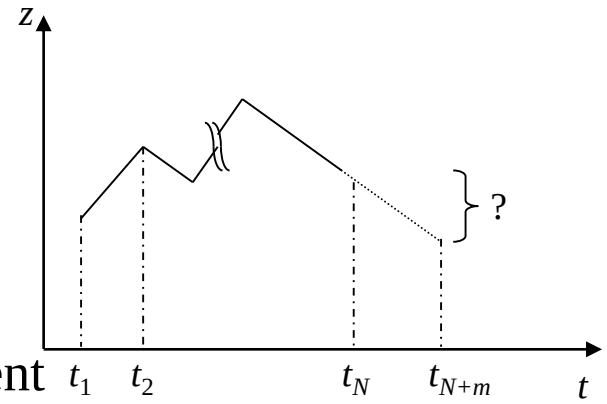
- Practical thing to remember: Model complexity should match the *quality and quantity* of the data
  - When the *data is noisy and sparse*, even if you know the model structure (e.g., 15 data samples generated from a 10^{th} order polynomial), often a simpler model (e.g., a 2^{nd} order model) provides better out-of-sample performance!
    - Noise amplifies the effects of overfitting
  - When the data is generated from a complicated model (e.g., a 50^{th} order model and *you do not know that*) and is *noiseless*, a simpler model (e.g., a 2^{nd} order model versus a 10^{th} order model) often wins out
    - Target complexity ("complex data generating mechanism") increases the effects of overfitting
  - Effects of overfitting decrease as the number of samples *N* increases
  - *Regularization*, *Model selection via validation datasets* and *Cross-validation* are the tools to combat overfitting

Problem 4: Estimation & Prediction - 1

- **Prediction/Forecasting (Regression)**

  Given $\{z(t_1),\ z(t_2),\ \ldots\ldots\ldots\ldots\ldots z(t_N)\}$, predict $\hat{z}(t_{N+m})$

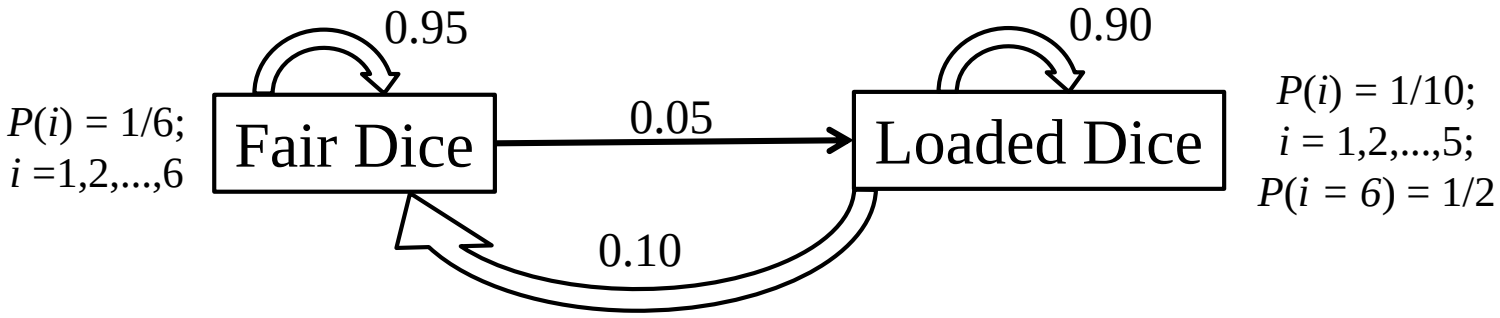
- **Areas:** Estimation theory, Adaptive Filtering



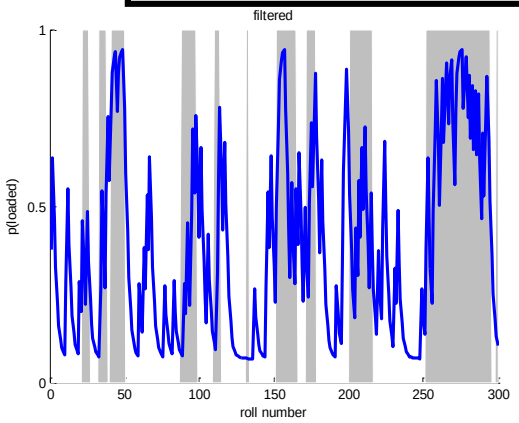
- **Typical Applications**
  - Stock market prediction
  - Weather forecasting
  - Prognostics for system health management
  - Ad selection systems
  - Medicine
  - Fraud detection
  - Power Demand Prediction

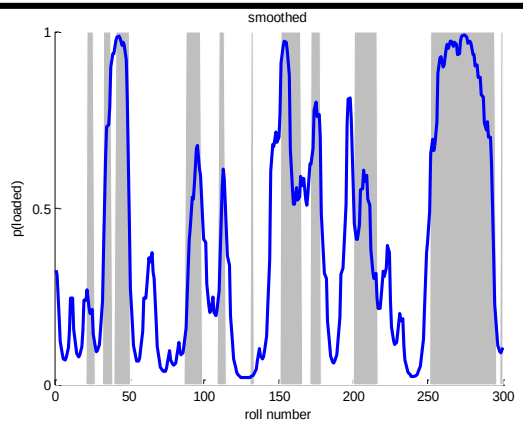
Problem 4: Estimation & Prediction - 2

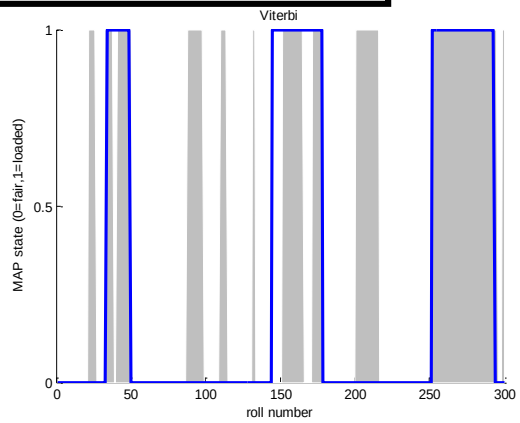
- How to detect an occasionally dishonest Casino? **"Classification"**



Run `casinoDemo` in pmtk3-master\demos of Murphy to get these







P (Loaded Dice | Data) *(On-line Filtered Estimate)*	*P (Loaded Dice | Data)* *(Off-line Smoothed Estimate)*	*Decisions: Loaded or not* *(Viterbi, MAP Estimate)*

- Learn the model based on data & use it to forecast future behavior of Casino

Problem 5: Optimization is Key to Learning

- **Optimization ("Regression")**

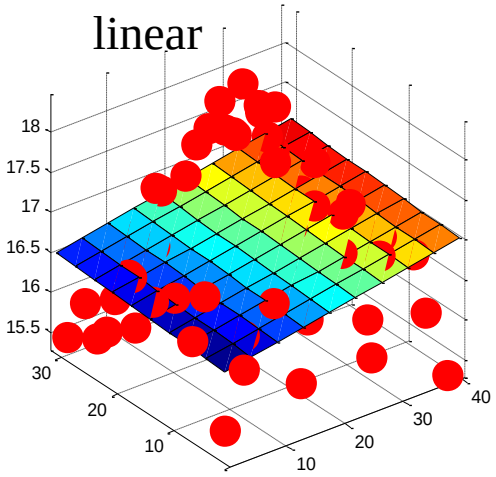
$$\text{"min } f(\underline{x}) \text{ s.t. } \underline{x} \in \Omega\text{"}$$

$$\Omega \begin{cases} h(\underline{x}) = \underline{0} \\ g(\underline{x}) \le \underline{0} \\ \underline{x}_{LB} \le \underline{x} \le \underline{x}_{UB} \\ x_i \text{ integer} \end{cases}$$

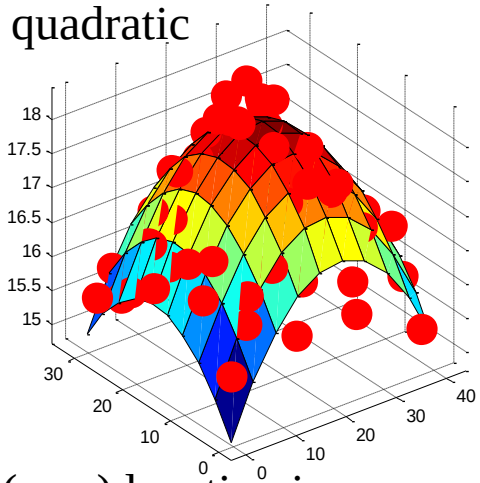
Non-Convex
Non-Differentiable
Combinational (e.g., TSP)

Run `surfaceFitDemo` in pmtk3-master\demos of Murphy to get these

linear



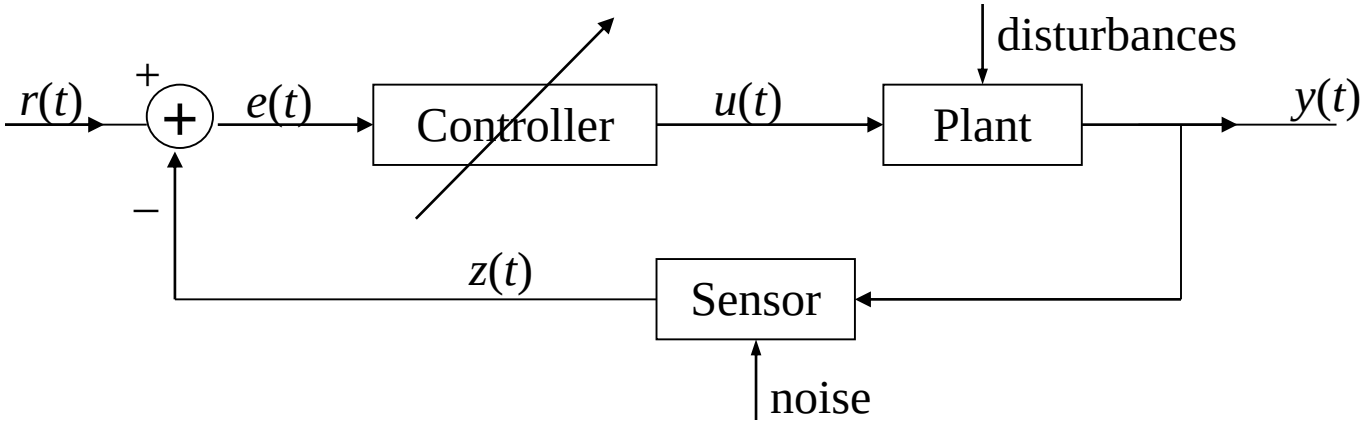
quadratic



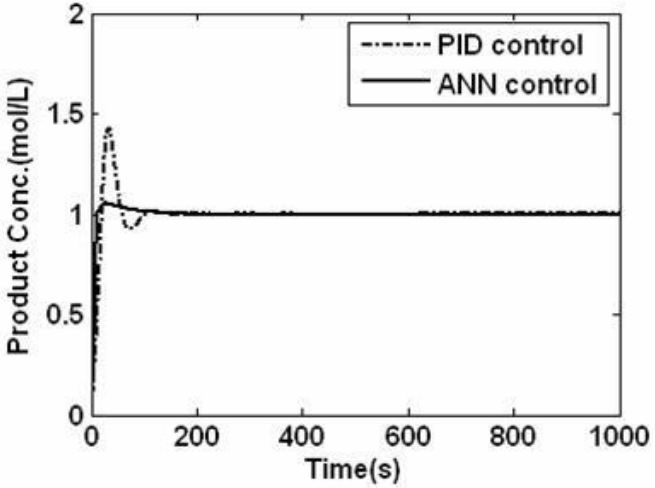
Temperature as a function of (x , y) location in a room

Problem 6: Adaptive Control

- Adaptive (Neuro) Control (Regression, Classification)



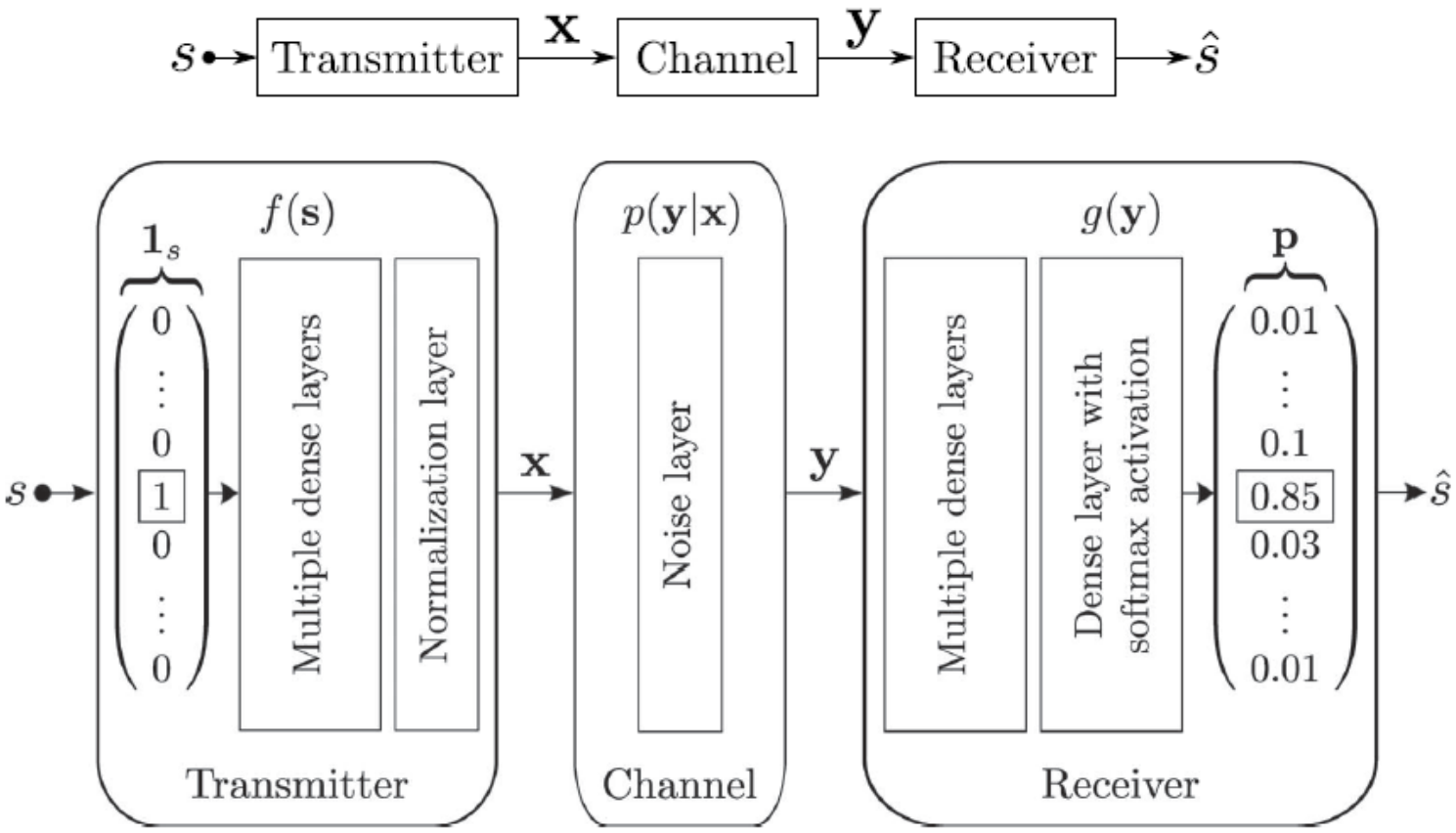
Design of a controller when have uncertain info. about the system



Lots of tuning required to make PID work

Problem 7: Adaptive Communications

- Deep Neural Networks for Physical Layer Design*
  (Mostly Classification)



* T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," *IEEE Trans. on Cognitive Communications and Networking*, Vol. 3, No. 4, pp. 563-575, Dec 2017.

Model Selection Problem is Ubiquitous

- How many clusters in the Data?
- What is the order of the system?
- What is the intrinsic dimension of the system?
- Is this feature relevant?
- How many states in a Hidden Markov Model (HMM)?
- What is the structure of a graphical model?
- How many independent sources in the input?

Model Selection Criteria

- Akaike Information Criterion (AIC)
- Bayesian Information Criterion (BIC)
- Minimum Description Length (MDL)
- Cross validation
- Regularization

Bayesian Methods provide a Principled Approach to Model Selection

Linear Regression Models

- Let us consider the linear case first

  $\rightarrow$ ADALINE (Adaptive Linear Element)

$$y(\underline{x}, \underline{w}) = \underline{w}^T \underline{x}$$

Also, linear if know the form of the polynomial

$w_0 + w_1 x_1 + \cdots\cdots + w_p x_p + w_{12} x_1 x_2 + \cdots + w_{pp} x_1 x_p + \cdots + w_{12} x_p^{\,2}$ etc.

or transformed $\underline{x}, \phi(\underline{x})$, e.g.,Chebyshev, Legendre polynomials

$$\min J(\underline{w}) = \frac{1}{2} \sum_{n=1}^{N} (z^n - \underline{w}^T \underline{x}^n)^2$$

$$\nabla J(\underline{w}) = \underline{g} = -\sum_{n=1}^{N} (z^n - \underline{w}^T \underline{x}^n) \underline{x}^n = -\sum_{n=1}^{N} e_n \underline{x}^n$$

Least Squares Solution

- Key: "Gradient is sum over training patterns"

$$\nabla^2 J(\underline{w}) = \sum_{n=1}^{N} (\underline{x}^n \left(\underline{x}^n\right)^T) = \text{ "information matrix"}$$

"Hessian is sum over training patterns as well"

Let us look at several possibilities:

Optimum $\Rightarrow \nabla J(\underline{w}) = 0$

$$\Rightarrow \sum_{n=1}^{N} \underline{x}^n z^n = \left[\sum_{n=1}^{N} \underline{x}_n \underline{x}_n^T\right] \underline{w}$$

Let $X = \begin{bmatrix} \left(\underline{x}^1\right)^T \\ \left(\underline{x}^2\right)^T \\ \vdots \\ \left(\underline{x}^N\right)^T \end{bmatrix} \sim N \text{ by } (p+1) \text{ matrix;} \quad \underline{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_N \end{bmatrix}$

Orthogonalization Methods

$$X^T X \underline{w} = X^T \underline{z}$$

$$\underline{w} = (X^T X)^{-1} X^T \underline{z} = X^\dagger \underline{z}$$

$X^\dagger =$ pseudo inverse of X, a $(p+1)$ by N matrix

$(generalized\ \ inverse)$

corresponds to solving

$X \underline{w} = \underline{z}$ in a **least squares** sense

How to find generalized inverse?

1. **Orthogonalization methods (Gram-Schmidt, Givens, Householder):**

- Gram-Schmidt: $X = QR$ $\quad Q$ is N by $(p+1)$, R is $(p+1)$ by $(p+1)$

Singular Value Decomposition

- Givens and Householder:

$$X = \tilde{Q}\tilde{R} \qquad \tilde{Q} \text{ is } N \text{ by } N,\ \tilde{R} \text{ is } N \text{ by } (p+1);\quad \tilde{Q} = N[\, Q \overset{p+1}{\vdots} Q_{1} \overset{N-p-1}{\,}];\ \tilde{R} = \begin{bmatrix} R \\ \cdots \\ 0 \end{bmatrix}$$

$$= QR$$

so, $\underline{w} = (X^T X)^{-1} X^T \underline{z} = (R^T R)^{-1} R^T Q^T \underline{z}$

or, $\underline{w} = R^{-1} Q^T \underline{z}$ if R is full rank

2. Singular Value Decomposition (SVD)

$$\underset{N}{X}^{p+1} = \underset{N}{U}^{N}\ \underset{N}{\Sigma}^{p+1}\ \underset{(p+1)}{V^T} \qquad U, V \text{ orthogonal} \Rightarrow U^{-1} = U^T,\ V^{-1} = V^T$$

$$= \sum_{i=1}^{p+1} \sigma_i \underline{u}_i \underline{v}_i^{\,T}$$

$$X^{+} = \sum_{i=1}^{p+1} \frac{\underline{v}_i \underline{u}_i^{\,T}}{\sigma_i} \qquad \underline{w} = \sum_{i=1}^{p+1} \left(\frac{\underline{u}_i^{\,T} \underline{z}^{N}}{\sigma_i} \right) \underline{v}_i \in R(X^T)$$

Can we process data sequentially?

Tests of Significance & Feature Selection

- Prediction at training inputs

$$\underline{\hat{z}} = X\,\underline{\hat{w}} = XX^{\dagger}\,\underline{z} = X\,(X^T X)^{-1} X^T\,\underline{z}$$

- Estimate of measurement noise variance

$$\hat{\sigma}^2 = \frac{1}{N-p-1}\sum_{n=1}^{N}(z^n - \hat{z}^n)^2$$

- To test if $w_j = 0$, form Z-score

$$\beta_j = \frac{\hat{w}_j}{\hat{\sigma}\sqrt{diag\left[(X^T X)^{-1}\right]_{jj}}} \sim t(\beta_j; N-p-1)$$

$$If\ |\beta_j| < 2,\ consider\ dropping\ \ feature\ x_j.$$

- Two feature sets: {S} and {S'}

$$F = \frac{\left[J(\underline{w}_S) - J(\underline{w}_{S'})\right] / \left(|S'| - |S|\right)}{J(\underline{w}_{S'}) / (N - |S'| - 1)}$$

Add feature with the largest F and stop when F<95 percentile of $F_{1,(N-|S'|-1)}$

Variations : Shrinkage Methods

- Ridge (L_2) Regression

$$J(\underline{w}) = \| \underline{z} - X\underline{w} \|_2^2 + \mu \| \underline{w} \|_2^2 \text{ .....assume input and output data is centered}$$

$$\Rightarrow \hat{\underline{w}} = \left(X^T X + \mu I_p \right)^{-1} X^T \underline{z} = X^T \left(XX^T + \mu I_N \right)^{-1} \underline{z} \text{ ....via Matrix Inversion Lemma}$$

$$(or)\, \hat{\underline{w}} = X^T \underline{\alpha}; \underline{\alpha} = \left(XX^T + \mu I_N \right)^{-1} \underline{z} \text{....Kernel form} \Rightarrow \hat{z} = \hat{\underline{w}}^T \underline{x} = \sum_{i=1}^{N} \alpha_i \underline{x}_i^T \underline{x}$$

$$\text{so, } \hat{\underline{z}} = \underbrace{X \left(X^T X + \mu I_p \right)^{-1} X^T}_{P_\mu} \underline{z} = \sum_{j=1}^{p} \left(\frac{\lambda_j^2}{\lambda_j^2 + \mu} \right) \underline{u}_j \underline{u}_j^T \underline{z}; \lambda_j^2 = \text{eigen value of } X^T X = \sigma_j^2$$

- L_1 Regression (Compressed Sensing, Lasso)

$$J(\underline{w}) = \| \underline{z} - X\underline{w} \|_2^2 + \mu \| \underline{w} \|_1$$

LASSO: least absolute shrinkage and selection operator

- Combined L_1 and L_2 Regression ("Elastic Nets"… related to SVM)

https://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9856/10002

$$J(\underline{w}) = \| \underline{z} - X\underline{w} \|_2^2 + \mu_1 \| \underline{w} \|_1 + \mu_2 \| \underline{w} \|_2^2$$

- You can use reduced dimensioned features

$$X_r = XV_r$$

PCA

V_r = first r singular vectors corresponding to largest r singular values

LOOCV for Ridge Regression

- Key: Can compute $\hat{z}_{i,-i}$, prediction of z_i when trained on all data except i, from $\hat{z}_i$ and $H = (X^T X + \mu I)^{-1}$ or $P_\mu = X(X^T X + \mu I)^{-1} X^T$

$$\hat{\underline{z}}_{-i} = \underline{x}_i^T \hat{\underline{w}}_{-i} = \underline{x}_i^T \left(X^T X + \mu I - \underline{x}_i \underline{x}_i^T \right)^{-1} (X^T \underline{z} - \underline{x}_i z_i)$$

$$= \underline{x}_i^T \left[H + \frac{H \underline{x}_i \underline{x}_i^T H}{1 - \underline{x}_i^T H \underline{x}_i} \right] (X^T \underline{z} - \underline{x}_i z_i); H = \left(X^T X + \mu I \right)^{-1}$$

$$= \underline{x}_i^T H X^T \underline{z} - \underline{x}_i^T H \underline{x}_i z_i + \frac{\underline{x}_i^T H \underline{x}_i \underline{x}_i^T H X^T \underline{z}}{1 - \underline{x}_i^T H \underline{x}_i} - \frac{\underline{x}_i^T H \underline{x}_i \underline{x}_i^T H \underline{x}_i z_i}{1 - \underline{x}_i^T H \underline{x}_i}$$

$$= (1 + \frac{\underline{x}_i^T H \underline{x}_i}{1 - \underline{x}_i^T H \underline{x}_i}) \left[\underline{x}_i^T \underbrace{H X^T \underline{z}}_{\hat{\underline{w}}} - \underline{x}_i^T H \underline{x}_i z_i \right]$$

$$= \frac{\hat{z}_i - \underline{x}_i^T H \underline{x}_i z_i}{1 - \underline{x}_i^T H \underline{x}_i} = \frac{\hat{z}_i - P_{\mu,ii} z_i}{1 - P_{\mu,ii}}$$

- Normalized Validation Error with LOOCV

$$E_{LOOCV} = \frac{1}{N} \sum_{i=1}^{N} (z_i - \hat{z}_{-i})^2 = \frac{1}{N} \sum_{i=1}^{N} \left(z_i - \frac{\hat{z}_i - \underline{x}_i^T H \underline{x}_i z_i}{1 - \underline{x}_i^T H \underline{x}_i} \right)^2$$

$$= \frac{1}{N} \sum_{i=1}^{N} \left(\frac{z_i - \hat{z}_i}{1 - \underline{x}_i^T H \underline{x}_i} \right)^2 = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{z_i - \hat{z}_i}{1 - P_{\mu,ii}} \right)^2$$

It is rare to have such closed-form expressions for validation error

Bayes' Theorem and Inference

- Sum Rule (Total Probability Theorem)

$$P(A)=\sum_B P(A,B)$$

- Product Rule

$$P(A,B)=P(B\mid A)P(A)=P(A\mid B)P(B)$$

- Basic Axioms of Probability
  - Probability of event A, $P(A)\in[0,1]$
  - $P(A)=1 \Leftrightarrow$ A is certain
  - $P(A\cup B)=P(A \textbf{ or } B)=P(A)+P(B)-P(A\,B)$
  - Bayes' theorem
    - $P(AB)=P(A\mid B)P(B)=P(B\mid A)P(A)$
    - $P(A\mid B)=\dfrac{P(B\mid A)P(A)}{P(B)}$

We are inferring A given data B

  - Interested in A
  - Begin with *a priori* probability $P(A)$ for our belief about A
  - Observe B
  - Bayes' theorem provides the revised belief about A, that is, the posterior probability $P(A|B)$
  - Likelihood of A: The quantity $P(B|A)$, as a function of varying A for a fixed B
  - $posterior \propto prior \times likelihood$

$$\Rightarrow P(A|B) \propto P(A)\,.\,P(B|A)$$

- P ~ Probability
- p ~ Probability Density Function (pdf)

Bayes' Theorem and Learning

- Data D; Model Parameters, $\underline{w}$; model set $M \in \{1,2,..,m,\ldots,M\}$

$$p(\underline{w} \mid D, m) = \frac{p(D \mid \underline{w}, m)\, p(\underline{w} \mid m)}{p(D \mid m)}$$

$p(D \mid \underline{w}, m)$ = Likelihood of $\underline{w}$ under model m

$p(\underline{w} \mid m)$ = prior of $\underline{w}$ under model m

$p(\underline{w} \mid D, m)$ = posterior of $\underline{w}$ given data D under model m

- Prediction

$$p(\underline{x} \mid D, m) = \int_{\underline{w}} p(\underline{x}, \underline{w} \mid D, m)\, d\underline{w} = \int_{\underline{w}} p(\underline{x} \mid \underline{w}, m)\, p(\underline{w} \mid D, m)\, d\underline{w}$$

- Model Comparison

$$P(m \mid D) = \frac{p(D \mid m)P(m)}{\sum_{k=1}^{M} p(D \mid k)P(k)}; \quad p(D \mid m) = \int_{\underline{w}} p(D \mid \underline{w}, m)\, p(\underline{w} \mid m)\, d\underline{w} \sim \text{Model Evidence}$$

Interpretations of Model Evidence:

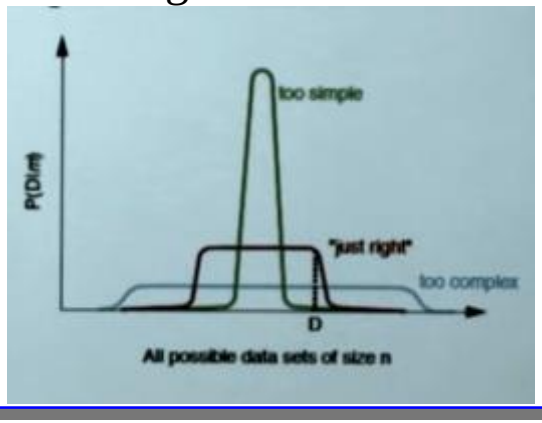
- Pdf that randomly selected model parameters from the prior generate D
- $-\log_2 p(D|m)$ is the *number of bits of surprise* at observing D under model m
- Bayesian model averaging …. Computationally intractable in general

Bayesian Recursive Learning and Inference

- Learning and Inference are not different in a Bayesian approach
- Recursion

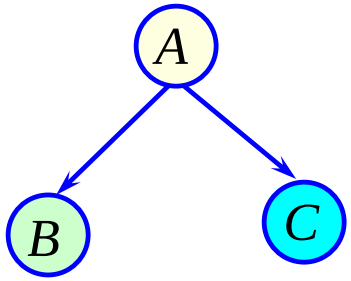
$$p(\underline{w} \mid D, \underline{x}, m) = \frac{p(\underline{w}, D, \underline{x} \mid m)}{p(D, \underline{x} \mid m)} = \frac{p(\underline{x} \mid \underline{w}, m)\, p(\underline{w} \mid D, m)}{\int_{w} p(\underline{x} \mid \underline{w}, m)\, p(\underline{w} \mid D, m)\, d\underline{w}}$$

- Provides a principled method for Model Selection
  - Very Simple model structures are unlikely to generate observed data ("large bias")
  - Very Complex model structures are unlikely to generate the particular data set observed because they can generate many possible data sets ("large variance")
  - The Right model is one with large model evidence for the observed data set *D…. Large p(D|m)*



Graphical Modeling of Statistical Dependencies – 1

- Example 1 (Common Cause, Fork)



- Naïve Bayes
- iid observations
- prediction (sufficient statistics)

When A is not observed, B & C are dependent.
When A is observed, B &C are conditionally independent!

"tail-to-tail" dependency

$B \perp C \mid A$

"B is independent of C given A"

Why?

$$P(B,C) = \sum_A P(B \mid A)P(C \mid A)P(A) \neq P(B)P(C)$$

$$P(B,C \mid A) = \frac{P(A,B,C)}{P(A)} = P(B \mid A)P(C \mid A)$$

Discuss chain rule:

$$P(ABC) = P(A)P(B \mid A)P(C \mid A,B)$$

$$P(ABC) = P(A)P(B \mid A)P(C \mid A)$$

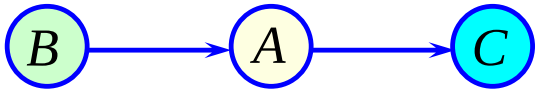
$$= \frac{P(AB)P(AC)}{P(A)}$$

Exploiting graph structure

Examples: 1. Smoking (A) causes Bronchitis (B) and Cancer (C).
2. Learning the probability of heads P(A=1), given the outcomes of a number of coin tosses (in this case, two (B & C))

Graphical Modeling of Statistical Dependencies – 2

- Example 2 (Indirect Cause, Chain)



"head-to-tail" dependency

$$P(ABC) = P(B)P(A|B)P(C|A)$$

$$Also,\ P(ABC) = \frac{P(AB)P(CA)}{P(A)}$$

$\{AB\}$, $\{CA\}$ "cliques"
A "separator"
$P(AB)$, $P(CA)$ are "clique potentials"
$P(A)$ "separator potential"

When A is not observed, B & C are dependent.
When A is observed, B & C are conditionally independent!

$B \perp C \mid A$

"B is independent of C given A"

Why?

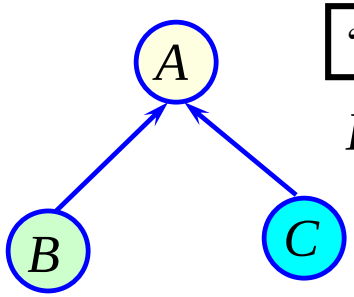
$$P(B,C) = P(B)\sum_{A} P(A|B)P(C|A) \neq P(B)P(C)$$

$$P(B,C|A) = \frac{P(A,B,C)}{P(A)} = \frac{P(A|B)P(B)P(C|A)}{P(A)} = P(B|A)P(C|A)$$

Example: B: Clouds; A: Rain; C: Grass is wet

Graphical Modeling of Statistical Dependencies – 3

- Example 3 (Common Effect, Collider)



"head-to-head" dependency

$$P(ABC) = P(B)P(C)P(A \mid B, C)$$

When A is not observed, B & C are **independent**!!
When A is observed, B & C are conditionally **dependent**!!

$B \perp C \mid \varnothing$

"B is independent of C given no evidence"

Why?

$$P(B,C) = \sum_A P(B)P(C)P(A \mid B,C) = P(B)P(C)$$

They are not independent given A

$$P(B,C \mid A) = \frac{P(A,B,C)}{P(A)} = \frac{P(B)P(C)P(A \mid B,C)}{P(A)}$$

When A is not observed, A blocks the path from B to C. However, when A is observed, it unblocks the path from B to $C \Rightarrow$ *they become dependent*

Example: A: Engine Does Not Start; B: Battery; C: Fuel

Noisy ("Soft") XOR

B	C	P(A=1|B,C)	P(A=0|B,C)
0	0	0.10	0.90
0	1	0.99	0.01
1	0	0.80	0.20
1	1	0.25	0.75

P(B=1)=0.65
P(C=1)=0.77

Suppose we observed $A=0$. What is $P(B=1|A=0)$?

$$P(B=1|A=0)=\frac{P(A=0|B=1)P(B=1)}{P(A=0)}=\frac{P(A=0|B=1)P(B=1)}{P(A=0|B=1)P(B=1)+P(A=0|B=0)P(B=0)}$$

$$P(A=0|B=1)=\sum_{C=0}^{1}P(A=0,C|B=1)=\sum_{C=0}^{1}P(A=0|B=1,C)P(C|B=1)$$

$$=\sum_{C=0}^{1}P(A=0|B=1,C)P(C)=0.20*0.23+0.75*0.77=0.6235$$

Similarly, $P(A=0|B=0)=\sum_{C=0}^{1}P(A=0|B=0,C)P(C)=0.90*0.23+0.01*0.77=0.2147$

$$P(B=1|A=0)=\frac{0.6235*0.65}{0.6235*0.65+0.2147*0.35}=0.8436$$

$$P(B=0|A=0)=1-P(B=1|A=0)=0.1564$$

Exploiting Dependency Structure: Graphical Models- 1

- Chain rule (general dependency)

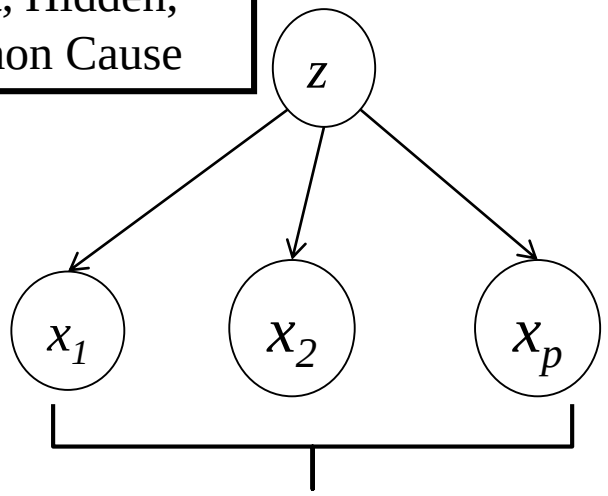
$$p(x_1, x_2, x_3, ..., x_{p-1}, x_p) = p(x_1)\,p(x_2 \mid x_1)\,p(x_3 \mid x_1, x_2)...p(x_p \mid x_1, x_2, ..., x_{p-1})$$

  - Storage complexity $O(K^p)$, assuming each variable has K states

- Conditional Independence
  - Naïve Bayes

$$p(\underline{x}, z) = p(z)\left(\prod_{i=1}^{p} p(x_i \mid z)\right)$$

Unknown, Latent, Hidden, Common Cause



- Storage complexity $[(K\text{-}1)(pC+1)]$

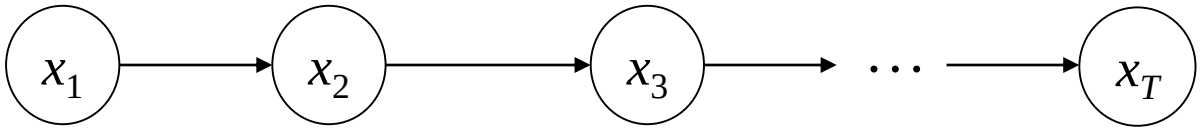
$$x_i \in \{1, 2, ..., K\}$$

$$z \in \{1, 2, ..., C\}$$

- Classifiers based on Naïve Bayes can work quite well

Exploiting Dependency Structure: Graphical Models - 2

- Conditional Independence (continued)
  - Markov chain (used to model sequential first order dependencies)



bigram

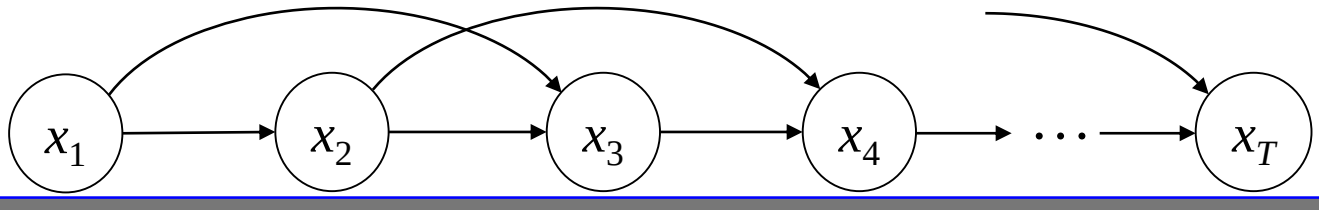
$$p(x_{1:T}) = p(x_1)\prod_{t=2}^{T} p(x_t \mid x_{t-1})$$

- Homogeneous: K^2
- Non-homogeneous: TK^2

Applications:
- Sentence completion
- Data compression
- Text classification
- Automatic essay writing
- Google's page rank

  - n^{th} order Markov chain ((**n+1)-gram** models)

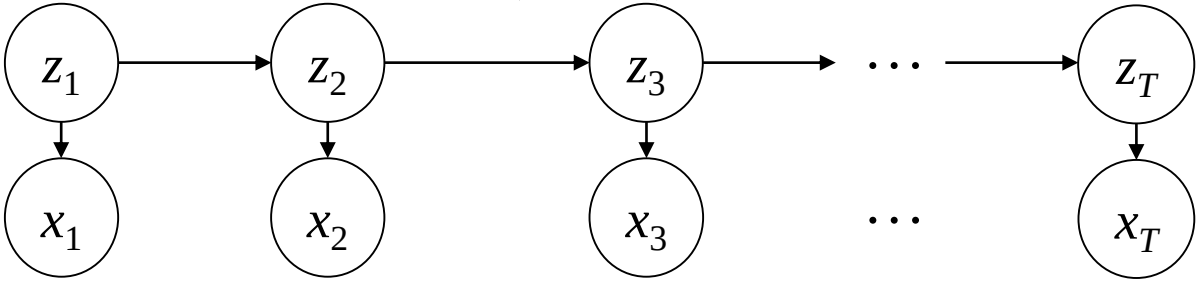
$$p(x_{1:T}) = p(x_1, x_2, .., x_n)\prod_{t=n+1}^{T} p(x_t \mid x_{t-1}, x_{t-2}, ..., x_{t-n}); T > n$$



$n = 2$
trigram

Exploiting Dependency Structure: Graphical Models - 3

- Conditional Independence (continued)
  - Hidden Markov model (HMM)

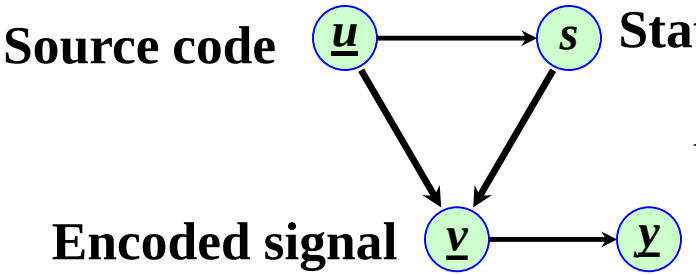


$$p(z_{1:T}, x_{1:T}) = p(z_{1:T})\, p(x_{1:T} \mid z_{1:T}) = p(z_1).\left(\prod_{t=2}^{T} p(z_t \mid z_{t-1})\right).\left(\prod_{t=1}^{T} p(x_t \mid z_t)\right)$$

  - **Applications:** speech recognition, activity recognition, gene finding, protein sequence alignment, diagnosis,…
  - **Algorithms:** Forwards-backwards algorithms, Viterbi decoding, Baum-Welch algorithm for learning HMM parameters from data
  - **Generalizations**: Higher order, Factorial, Coupled, semi-Markov, Hierarchical, I/O, Auto-regressive, Buried, state space, Dynamic Bayesian networks (DBNs),…

Three Broad Classes of Graphical Models - 1

- Bayesian Networks
  - Represented in terms of directed acyclic graphs (DAGs)



$$p(\underline{u}, s, \underline{v}, \underline{y}) = p(\underline{u})p(s \mid \underline{u})p(\underline{v} \mid s, \underline{u})p(\underline{y} \mid \underline{v})$$

- In general,

$$\mathbf{z} = [z_1\ z_2 \cdots\cdots z_N]$$

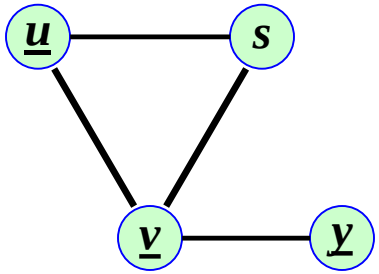
$$P(z_k \mid a_k) \qquad a_k = \text{parents of } z_k = pa(z_k)$$

$$P(\mathbf{z}) = \prod_{k=1}^{N} P(z_k \mid pa(z_k)) = \prod_{k=1}^{N} P(z_k \mid a_k)$$

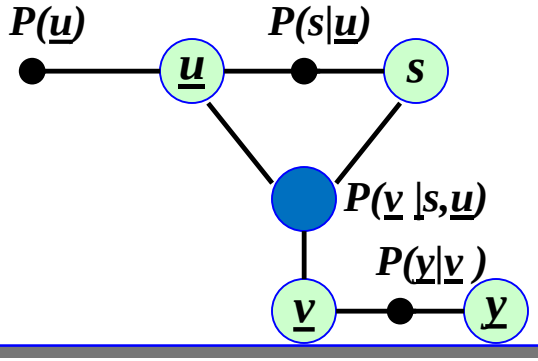
- Exploit graph properties
  - d-separation, moralization, cliques,…
- Algorithms
  - Variable elimination
  - Junction tree
  - Variational inference
  - Belief propagation (BP)
  - Expectation propagation (EP)
  - Markov Chain Monte Carlo

Three Broad Classes of Graphical Models - 2

- Markov random fields (MRF) and conditional MRFs… Undirected graphical model (UGM) or Markov network



- Factor Graphs
  - Useful when dealing with large graphs
  - Both BNs and UGMs can be converted into factor graphs

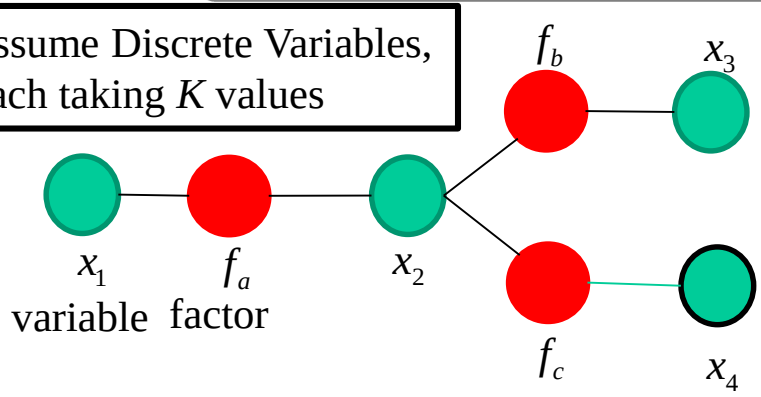


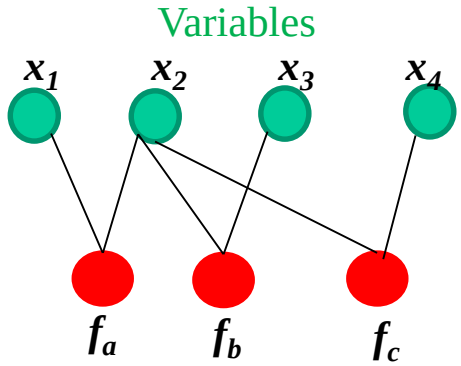
$$p(u,s,v,y) = \underbrace{p(u)}_{factor}\underbrace{p(s\mid u)}_{factor}\underbrace{p(v\mid s,u)}_{factor}\underbrace{p(y\mid v)}_{factor}$$

Propagate beliefs between variables → factors & factors → variables

Efficient Inference via Message Passing - 1

Assume Discrete Variables,
Each taking K values





Corresponding Bipartite Graph

$$P(x_1, x_2, x_3, x_4) = f_a(x_1, x_2)\, f_b(x_2, x_3)\, f_c(x_2, x_4)$$

Suppose want marginal distribution $P(x_2)$. By sum formula or Total Probability Theorem

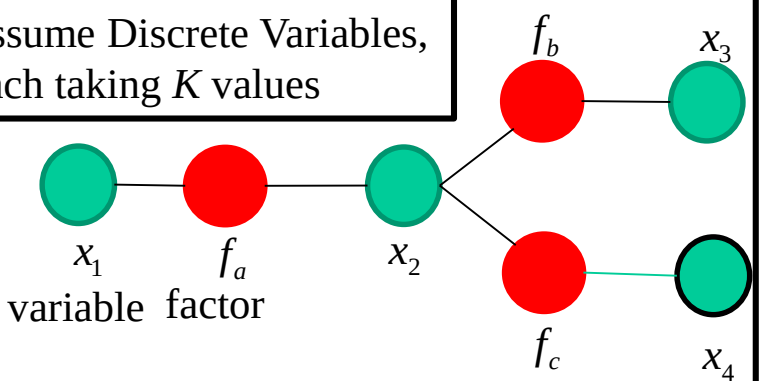
$$P(x_2) = \sum_{x_1}\sum_{x_3}\sum_{x_4} f_a(x_1, x_2)\, f_b(x_2, x_3)\, f_c(x_2, x_4)$$

$$= \underbrace{\left(\sum_{x_1} f_a(x_1, x_2)\right)}_{\mu_{f_a \to x_2}(x_2)} \underbrace{\left(\sum_{x_3} f_b(x_2, x_3)\right)}_{\mu_{f_b \to x_2}(x_2)} \underbrace{\left(\sum_{x_4} f_c(x_2, x_4)\right)}_{\mu_{f_c \to x_2}(x_2)} = \mu_{f_a \to x_2}(x_2)\mu_{f_b \to x_2}(x_2)\mu_{f_c \to x_2}(x_2)$$

Note : $2K^3$ multiplications and K^3 additions versus 2 multiplications and $3K$ additions for each x_2! In fact, can compute any marginal distribution by exchanging messages from variables to factors and factors to variables.

Efficient Inference via Message Passing - 2

Assume Discrete Variables, Each taking K values



To compute all marginals, form an arbitrary root node (say x_1), propagate and store messages from the leaves to the root, and then propagate and store messages from the root to the leaves. Marginals are products of the factor messages received at each variable nodes. Normalize marginal. Exact for trees. For arbitrary graphs, iterate (approximate belief propagation (ABF)). In fact, you can pass messages asynchronously from nodes $\rightarrow$ factors and factors $\rightarrow$ nodes until convergence.

$$P(x_1) = \mu_{f_a \rightarrow x_1}(x_1); P(x_3) = \mu_{f_b \rightarrow x_3}(x_3); P(x_4) = \mu_{f_c \rightarrow x_4}(x_4)$$

$$P(x_2) = \mu_{f_a \rightarrow x_2}(x_2)\mu_{f_b \rightarrow x_2}(x_2)\mu_{f_c \rightarrow x_2}(x_2)$$

Leaves $\rightarrow$ Root

$$\mu_{x_3 \rightarrow f_b}(x_3) = 1 \forall x_3$$

$$\mu_{f_b \rightarrow x_2}(x_2) = \sum_{x_3} f_b(x_2, x_3)\mu_{x_3 \rightarrow f_b}(x_3)$$

$$\mu_{x_4 \rightarrow f_c}(x_4) = 1 \forall x_4$$

$$\mu_{f_c \rightarrow x_2}(x_2) = \sum_{x_4} f_c(x_2, x_4)\mu_{x_4 \rightarrow f_c}(x_4)$$

$$\mu_{x_2 \rightarrow f_a}(x_2) = \mu_{f_b \rightarrow x_2}(x_2)\mu_{f_c \rightarrow x_2}(x_2)$$

$$\mu_{f_a \rightarrow x_1}(x_1) = \sum_{x_2} f_a(x_1, x_2)\mu_{x_2 \rightarrow f_a}(x_2)$$

Root $\rightarrow$ Leaves

$$\mu_{x_1 \rightarrow f_a}(x_1) = 1 \forall x_1$$

$$\mu_{f_a \rightarrow x_2}(x_2) = \sum_{x_1} f_a(x_1, x_2)\mu_{x_1 \rightarrow f_a}(x_1)$$

$$\mu_{x_2 \rightarrow f_b}(x_2) = \mu_{f_a \rightarrow x_2}(x_2)\mu_{f_c \rightarrow x_2}(x_2)$$

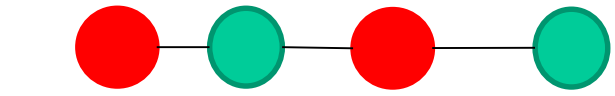
$$\mu_{x_2 \rightarrow f_c}(x_2) = \mu_{f_a \rightarrow x_2}(x_2)\mu_{f_b \rightarrow x_2}(x_2)$$

$$\mu_{f_b \rightarrow x_3}(x_3) = \sum_{x_2} f_b(x_2, x_3)\mu_{x_2 \rightarrow f_b}(x_2)$$

$$\mu_{f_c \rightarrow x_4}(x_4) = \sum_{x_2} f_c(x_2, x_4)\mu_{x_2 \rightarrow f_c}(x_2)$$

Evidence is easy to include

Bayes Rule as Message Passing



$$P(x) = \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix}; x \in \{1,2\}$$

$$P(y \mid x) = \begin{bmatrix} 0.05 & 0.65 & 0.30 \\ 0.8 & 0.10 & 0.10 \end{bmatrix}; y \in \{1,2,3\}$$

Want $P(y)$ and $P(x \mid y=1)$

Bishop's "Who done it" Example

Marginal $P(y)$

$$\mu_{f_a \to x}(x) = P(x) \Rightarrow \mu_{f_a \to x}(x=1) = 0.8; \mu_{f_a \to x}(x=2) = 0.2$$

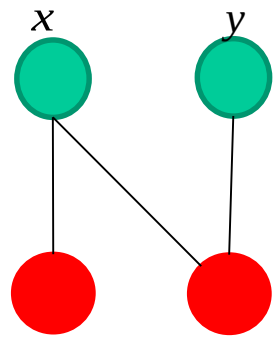
$$\mu_{x \to f_b}(x) = \mu_{f_a \to x}(x)$$

$$\mu_{f_b \to y}(y) = \sum_x f_b(x,y)\mu_{x \to f_b}(x) \Rightarrow \mu_{f_b \to y}(y=1) = 0.2; \mu_{f_b \to y}(y=2) = 0.54; \mu_{f_b \to y}(y=3) = 0.26$$

$$P(y) = \mu_{f_b \to y}(y)$$

Evidence, $y=1$. What is $P(x \mid y=1)$?

Treat x as the root.



$f_a = P(x)$ $f_b = P(y \mid x)$

$$\mu_{y \to f_b}(y=1) = 1; \mu_{y \to f_b}(y=2) = 0; \mu_{y \to f_b}(y=3) = 0$$

$$\mu_{f_b \to x}(x) = \sum_{y=1}^{3} f_b(x,y)\mu_{y \to f_b}(y) \Rightarrow \mu_{f_b \to x}(x=1) = 0.05; \mu_{f_b \to x}(x=2) = 0.8$$

$$\mu_{f_a \to x}(x) = P(x) \Rightarrow \mu_{f_a \to x}(x=1) = 0.8; \mu_{f_a \to x}(x=2) = 0.2$$

$$P(x \mid y=1) = \mu_{f_a \to x}(x)\mu_{f_b \to x}(x) \Rightarrow \begin{bmatrix} \mu_{f_a \to x}(1)\mu_{f_b \to x}(1) \\ \mu_{f_a \to x}(2)\mu_{f_b \to x}(2) \end{bmatrix} = \begin{bmatrix} 0.04 \\ 0.16 \end{bmatrix} \Rightarrow Normalize: \begin{bmatrix} 0.20 \\ 0.80 \end{bmatrix}$$

Sum-Product Belief Propagation Algorithm

- **Factors → Variables Messages (SUM)**

  messages sent by a factor node to a variable node involves multiplying all the incoming messages (except variable node x) with the factor and summing over all the variables except x

$$\mu_{f_a \to x_1}(x_1) = \sum_{x_2} f_a(x_1, x_2)\mu_{x_2 \to f_a}(x_2); \mu_{f_a \to x_2}(x_2) = \sum_{x_1} f_a(x_1, x_2)\mu_{x_1 \to f_a}(x_1)$$

$$\mu_{f_b \to x_2}(x_2) = \sum_{x_3} f_b(x_2, x_3)\mu_{x_3 \to f_b}(x_3); \mu_{f_b \to x_3}(x_3) = \sum_{x_2} f_b(x_2, x_3)\mu_{x_2 \to f_b}(x_2)$$

$$\mu_{f_c \to x_2}(x_2) = \sum_{x_4} f_c(x_2, x_4)\mu_{x_4 \to f_b}(x_4); \mu_{f_c \to x_4}(x_4) = \sum_{x_2} f_c(x_2, x_4)\mu_{x_2 \to f_c}(x_2)$$

- **Variables → Factors Messages (PRODUCT)**

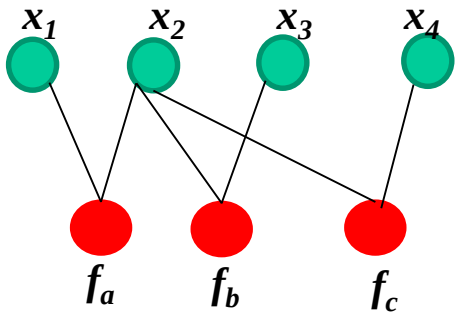
  Message sent by a variable node to a factor node is the product of all the incoming messages along all of the other links (factors)

$$\mu_{x_1 \to f_a}(x_1) = 1; \mu_{x_3 \to f_b}(x_3) = 1; \mu_{x_4 \to f_c}(x_3) = 1$$

$$\mu_{x_2 \to f_a}(x_2) = \mu_{f_b \to x_2}(x_2)\mu_{f_c \to x_2}(x_2);$$

$$\mu_{x_2 \to f_b}(x_2) = \mu_{f_a \to x_2}(x_2)\mu_{f_c \to x_2}(x_2);$$

$$\mu_{x_2 \to f_c}(x_2) = \mu_{f_a \to x_2}(x_2)\mu_{f_b \to x_2}(x_2)$$



Exponential Family

- **Exponential family:**

  Gaussian, Poisson, Gamma, Binomial, Exponential, Beta, Weibull,…

$$p(\underline{x}\,|\,\underline{w})^{\dagger} = \frac{1}{Z(\underline{w})} h(\underline{x}) \exp\{\underline{w}^T \underline{T}(\underline{x})\} = h(\underline{x}) \exp\{\underline{w}^T \underline{T}(\underline{x}) - A(\underline{w})\}$$

$$Z(\underline{w}) = \int_{\underline{x}} h(\underline{x}) \exp\{\underline{w}^T \underline{T}(\underline{x})\} d\,\underline{x} \quad \text{..... called partition function}$$

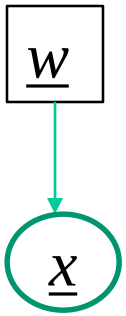
$A(\underline{w}) = \ln Z(\underline{w})$ ... log partition function or cumulant function

Gradients of $A(\underline{w})$ give moments of density

$\underline{w} \sim$ canonical parameters

$\underline{T}(\underline{x}) =$ "sufficient statistics"

$h(\underline{x}) =$ scaling constant



† most often $\underline{\theta}$ is used to denote model parameters. Since Neural Networks use $\underline{w}$ for weights, we use $\underline{w}$ here.

Exponential Family and Conjugate Priors

❑ Conjugate Prior

$p(\underline{w}) = g(\eta, \underline{v}) \exp\{\underline{w}^T \underline{v} - \eta A(\underline{w})\}; \eta, \underline{v}$ are hyper parameters

Data $D = \{\underline{x}_1, \underline{x}_2, ..., \underline{x}_N\}$

$$p(\underline{x}_n \mid \underline{w}) = h(\underline{x}_n) \exp\{\underline{w}^T \underline{T}(\underline{x}_n) - A(\underline{w})\}$$

$$p(\underline{w} \mid D) = g(\eta + N, \underline{v} + \sum_{n=1}^{N} \underline{T}(\underline{x}_n)) \exp\{\underline{w}^T [\underline{v} + \sum_{n=1}^{N} \underline{T}(\underline{x}_n)] - (\eta + N) A(\underline{w})\}$$

Posterior belongs to the exponential family

❑ Examples of Likelihoods and Conjugate Priors

- (Bernoulli or Binomial or Geometric) –Beta
- Scalar Normal-(Normal, Inverse-Gamma)
- Multivariate Normal-(Normal, Inverse Wishart)
- (Poisson, Pareto, Exponential)-Gamma

Bernoulli Distribution

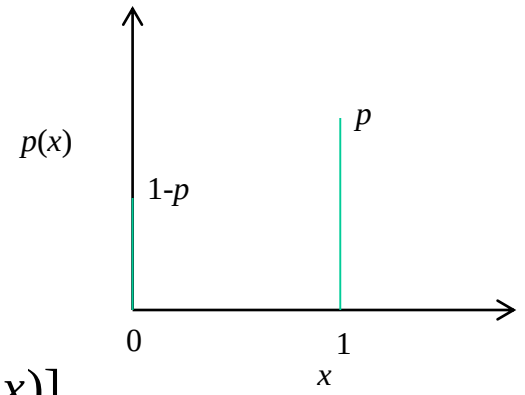
Bernoulli

$$p(x) = p^x (1-p)^{1-x} = \exp\left[x \ln p + (1-x)\ln(1-p)\right] = \exp\left[x \ln \frac{p}{1-p} + \ln(1-p)\right]$$

Let $T(x) = x;\ w = \ln \frac{p}{1-p} \Rightarrow p = \frac{1}{1+e^{-w}} = \sigma(w)$

$$\Rightarrow 1-p = \frac{1}{1+e^{w}} = \sigma(-w) = 1-\sigma(w)$$

so, $p(x) = \exp[x \ln \frac{p}{1-p} - \ln(1+e^{w})]$



$$A(w) = \ln(1+e^{w})$$

$$\frac{dA(w)}{dw} = \frac{e^{w}}{1+e^{w}} = \frac{1}{1+e^{-w}} = p = E(x) = E[T(x)]$$

$$\frac{d^2 A(w)}{dw^2} = p(1-p) = Var(x) = Var[T(x)]$$

Binomial Likelihood -Beta Prior for Learning *p*

- Multiple trials, each with $x_i \in \{0,1\}$

$$s_n = \sum_{i=1}^{n} x_i ; s_{n+1} = s_n + x_{n+1} ; s_0 = 0$$

Beta Function: $B(\alpha,\beta) = \int_0^1 q^{\alpha-1}(1-q)^{\beta-1} dq = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} = \frac{(\alpha-1)!(\beta-1)!}{(\alpha+\beta-1)!}$

$$p(s_n \mid p) = \binom{n}{s} p^{s_n}(1-p)^{n-s_n} = \frac{p^{s_n}(1-p)^{n-s_n}}{(n+1)B(s_n+1, n+1-s_n)}$$

$$p(p) = \frac{p^{\alpha-1}(1-p)^{\beta-1}}{B(\alpha,\beta)} = Beta(\alpha,\beta)$$

$$p(p \mid s_n) = \frac{p^{s_n+\alpha-1}(1-p)^{n-s_n+\beta-1}}{\int_0^1 q^{s_n+\alpha-1}(1-q)^{n-s_n+\beta-1} dq} = \frac{p^{s_n+\alpha-1}(1-p)^{n-s_n+\beta-1}}{B(s_n+\alpha, n-s_n+\beta)} = Beta(s_n+\alpha, n-s_n+\beta)$$

Estimate: $\hat{p}_n = \frac{s_n+\alpha}{n+\alpha+\beta}$

Recursion: $\hat{p}_{n+1} = \frac{s_{n+1}+\alpha}{n+\alpha+\beta+1} = \frac{s_n+x_{n+1}+\alpha}{n+\alpha+\beta+1} = \frac{n+\alpha+\beta}{n+\alpha+\beta+1}\hat{p}_n + \frac{1}{n+\alpha+\beta+1}x_{n+1}$

So, $\hat{p}_{n+1} = \hat{p}_n + \frac{1}{n+\alpha+\beta+1}(x_{n+1} - \hat{p}_n)$ Similar to Least Squares and Kalman Filter

Univariate Gaussian (Normal) Distribution

❑ Univariate Gaussian Distribution

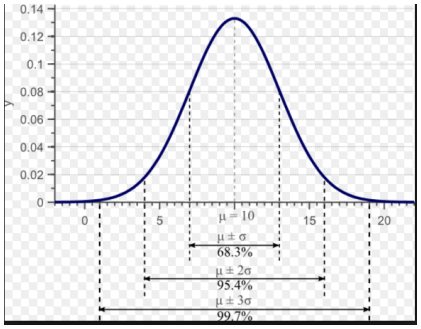
$$p(\underline{x}\,|\,\underline{w}) = \frac{1}{Z(\underline{w})} h(\underline{x}) \exp\{\underline{w}^T \underline{T}(\underline{x})\} = h(\underline{x}) \exp\{\underline{w}^T \underline{T}(\underline{x}) - A(\underline{w})\}$$

$$p(x\,|\,\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\{-\frac{1}{2\sigma^2}(x-\mu)^2\}$$

$$= \exp\{-\frac{1}{2\sigma^2}(x-\mu)^2 - \ln\sigma - \frac{1}{2}\ln(2\pi)\}$$

$$= \exp\{\frac{\mu}{\sigma^2}x - \frac{1}{2\sigma^2}x^2 - \frac{\mu^2}{2\sigma^2} - \ln\sigma - \frac{1}{2}\ln(2\pi)\}$$

$$= \underset{h(x)}{1} \exp\{\underbrace{\begin{pmatrix} \frac{\mu}{\sigma^2} \\ -\frac{1}{2\sigma^2} \end{pmatrix}}_{\underline{w}}{}^T \underbrace{\begin{pmatrix} x \\ x^2 \end{pmatrix}}_{\underline{T}(x)} - \underbrace{\left(\frac{\mu^2}{2\sigma^2} + \ln\sigma + \frac{1}{2}\ln(2\pi)\right)}_{A(\underline{w})}\}$$



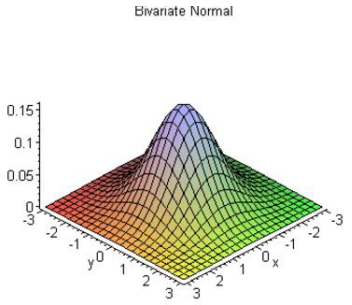
μ : mean
σ^2 : variance
$1/\sigma^2$: "information"

Multivariate Gaussian (Normal) Distribution

❑ Multivariate Gaussian Distribution

$$p(\underline{x}\,|\,\underline{w}) = \frac{1}{Z(\underline{w})} h(\underline{x}) \exp\{\underline{w}^T \underline{T}(\underline{x})\} = h(\underline{x}) \exp\{\underline{w}^T \underline{T}(\underline{x}) - A(\underline{w})\}$$

$$p(\underline{x}\,|\,\underline{\mu}, \Sigma) = \frac{1}{(2\pi)^{p/2} |\Sigma|^{1/2}} \exp\{-\frac{1}{2}(\underline{x}-\underline{\mu})^T \Sigma^{-1} (\underline{x}-\underline{\mu})\}$$



$$= \exp\{tr[-\frac{1}{2}\Sigma^{-1}(\underline{x}-\underline{\mu})(\underline{x}-\underline{\mu})^T - \frac{1}{2}\ln|\Sigma| - \frac{p}{2}\ln(2\pi)\}$$

$$= \exp\{tr[\underline{\mu}^T\Sigma^{-1}\underline{x} - \frac{1}{2}\Sigma^{-1}\underline{x}\underline{x}^T] - \frac{1}{2}\Sigma^{-1}\underline{\mu}\underline{\mu}^T - \frac{1}{2}\ln|\Sigma| - \frac{p}{2}\ln(2\pi)\}$$

$$= \underset{h(\underline{x})}{1} \exp\{tr[\underbrace{\begin{pmatrix} \Sigma^{-1}\underline{\mu} \\ -\frac{1}{2}\Sigma^{-1} \end{pmatrix}}_{\underline{w}}{}^T \underbrace{\begin{pmatrix} \underline{x} \\ \underline{x}\underline{x}^T \end{pmatrix}}_{\underline{T}(\underline{x})}] - \underbrace{\left(\frac{1}{2}\underline{\mu}^T\Sigma^{-1}\underline{\mu} + \frac{1}{2}\ln|\Sigma| + \frac{p}{2}\ln(2\pi)\right)}_{A(\underline{w})}\}$$

❑ Information form

$$p(\underline{x}; \underline{\mu}, J^{-1}) = \frac{|J|^{1/2}}{(2\pi)^{p/2}} \exp\{-\frac{1}{2}(\underline{x}-\underline{\mu})^T J (\underline{x}-\underline{\mu})\}$$

$\underline{\mu}$: mean
Σ : covariance matrix
$J = \Sigma^{-1}$: "information" matrix

Examples of Sufficient Statistics

$Normal: \sum_{i=1}^{N} x_i; \sum_{i=1}^{N} x_i^2$ for scalars; $\sum_{i=1}^{N} \underline{x}_i; \sum_{i=1}^{N} \underline{x}_i \underline{x}_i^T$ for vectors

(or) sample mean and sample covariance

Bernoulli: $\sum_{i=1}^{N} x_i$

$Poisson: \sum_{i=1}^{N} x_i$

$Exponential: \sum_{i=1}^{N} x_i$

$Uniform: \max(x_i)$

$Gamma: \sum_{i=1}^{N} x_i; \sum_{i=1}^{N} \ln x_i$ (or) $\sum_{i=1}^{N} x_i; \prod_{i=1}^{N} x_i$

$Beta: \sum_{i=1}^{N} \ln x_i; \sum_{i=1}^{N} \ln(1-x_i)$ (or) $\prod_{i=1}^{N} x_i; \prod_{i=1}^{N} (1-x_i)$

Sufficient statistics help in experimenting with data transformations and data reduction

More at: https://en.wikipedia.org/wiki/Exponential_family

Gaussian Likelihood-Gaussian Prior for Learning the Mean

- Want to learn the mean $\underline{\mu}$ (covariance Σ known)

$$p(\underline{x}^n \mid \underline{\mu}) = N(\underline{\mu}, \Sigma_v) \text{ and } p(\underline{\mu}) = N(\underline{\hat{\mu}}^0, \hat{\Sigma}^0)$$

- Data: $\text{Let } D^n = \{\underline{x}^1, \underline{x}^2, ..., \underline{x}^{n-1}, \underline{x}^n\} = \{D^{n-1}, \underline{x}^n\}$

$$p(D^n \mid \underline{\theta}) = p(\underline{x}^n \mid \underline{\theta})\, p(D^{n-1} \mid \underline{\theta}) \text{ ...conditionally i.i.d.}$$

- Gaussian density reproduces itself $\Rightarrow$ Posterior density is also Gaussian

$$p(\underline{x}^n) = N(\underline{\hat{\mu}}^{n-1}, \hat{\Sigma}^{n-1} + \Sigma_v)$$

$$p(\underline{\mu} \mid D^{n-1}) = N(\underline{\hat{\mu}}^{n-1}, \hat{\Sigma}^{n-1}) \text{ and } p(\underline{x}^n \mid \underline{\mu}) = N(\underline{\mu}, \Sigma_v)$$

Model:

$$\underline{x}^n = \underline{\mu} + \underline{v}^n;$$

$$\underline{v}^n = N(\underline{0}, \Sigma_v)$$

$$\underline{\hat{\mu}}^n = \underline{\hat{\mu}}^{n-1} + \hat{\Sigma}^{n-1}[\hat{\Sigma}^{n-1} + \Sigma_v]^{-1}(\underline{x}^n - \underline{\hat{\mu}}^{n-1}) \Rightarrow \textit{RLS or a simple Kalman filter}$$

$$\hat{\Sigma}^n = [(\hat{\Sigma}^{n-1})^{-1} + \Sigma_v^{-1}]^{-1} = \hat{\Sigma}^{n-1} - \hat{\Sigma}^{n-1}[\hat{\Sigma}^{n-1} + \Sigma_v]^{-1}\hat{\Sigma}^{n-1}$$

$$= \hat{\Sigma}^{n-1}[\hat{\Sigma}^{n-1} + \Sigma_v]^{-1}\Sigma_v$$

Add & subtract Σ_v

$$= \Sigma_v[\hat{\Sigma}^{n-1} + \Sigma_v]^{-1}\hat{\Sigma}^{n-1}; \hat{\Sigma}^0 \textit{ initial value}$$

Explict solution:

$$\underline{\hat{\mu}}^n = \hat{\Sigma}_n\left(\hat{\Sigma}_0^{-1}\underline{\hat{\mu}}^0 + \Sigma_v^{-1}\sum_{j=1}^{n}\underline{x}^j\right)$$

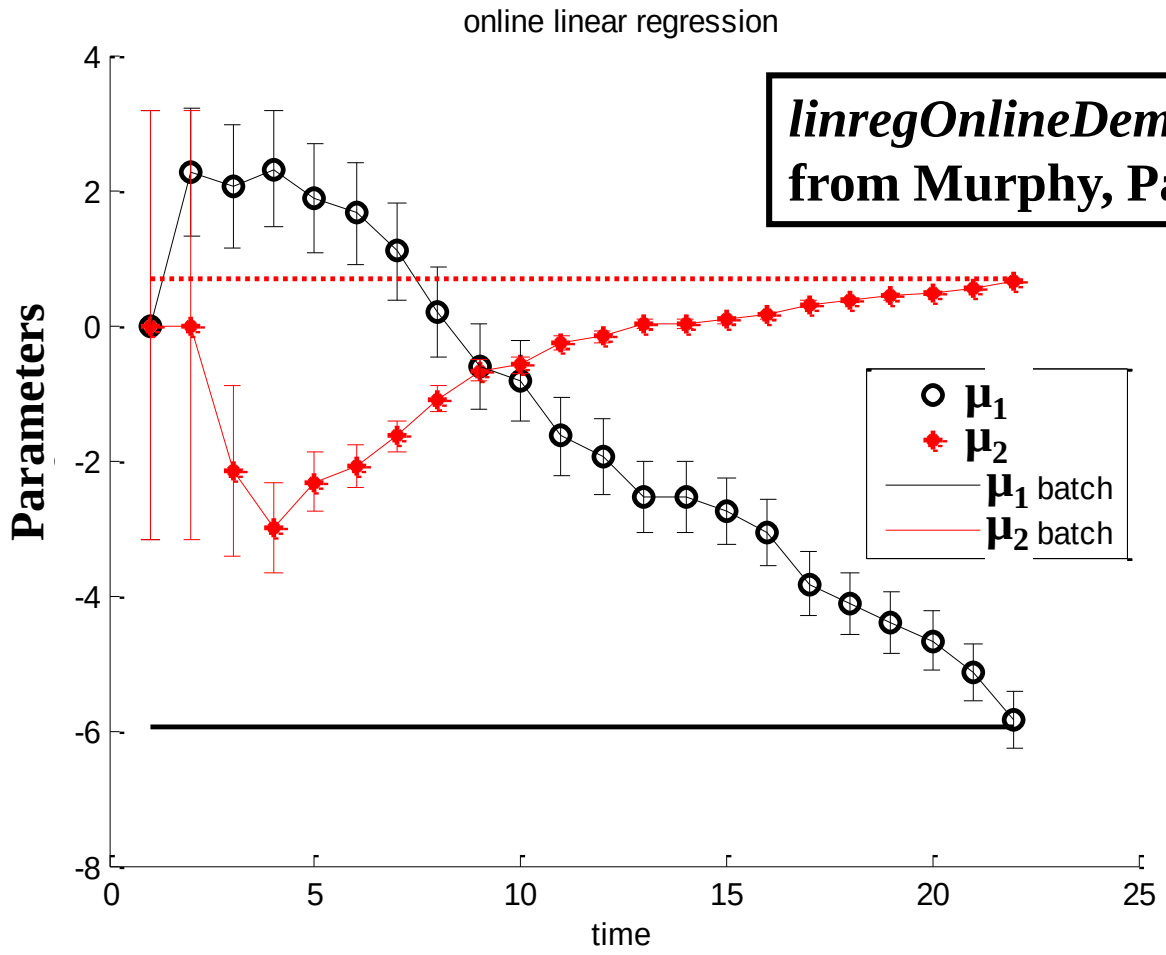
$$= \hat{\Sigma}_n\left(\hat{\Sigma}_0^{-1}\underline{\hat{\mu}}^0 + n\Sigma_v^{-1}\underline{\bar{x}}_n\right)$$

$$\hat{\Sigma}_n^{-1} = \hat{\Sigma}_0^{-1} + n\Sigma_v^{-1}$$

Information filter

What if mean drifts? KF,MM, IMM,..

Recursive Estimation of Constant Parameters



***linregOnlineDemoKalman* from Murphy, Page 636**

Measurement can be a linear combination of parameters corrupted by noise

Recursive Estimation of Drifting Parameters

Model :

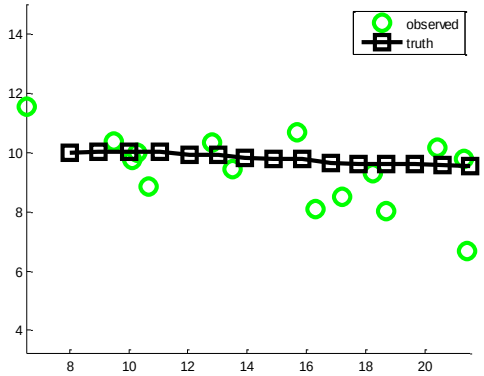
$$\underline{\mu}^{n+1} = \underline{\mu}^{n} + \underline{\delta}^{n}\Delta + \underline{w}_1^{n}; \Delta = \text{sampling interval}; \underline{w}_1^{n} \sim N(\underline{0}, Q_1); \underline{\delta}^{n} = \text{ drift rate at sample } n$$

$$\underline{\delta}^{n+1} = \underline{\delta}^{n} + \underline{w}_2^{n}; \underline{w}_2^{n} \sim N(\underline{0}, Q_2)$$

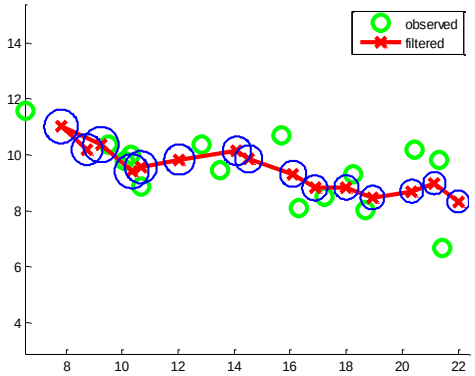
$$\underline{x}^{n} = \underline{\mu}^{n} + \underline{v}^{n}; \underline{v}^{n} = N(\underline{0}, \Sigma_v)$$

2-state nearly-constant velocity model

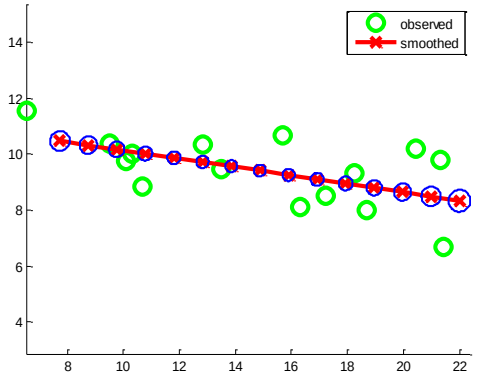
***kalmanTrackingDemo* from Murphy, Page 632**



True trajectory and measurements



Filtered Estimate



Smoothed Estimate

Entropy and Conditional Entropy

- Suppose $X \in \{x_1, x_2, .., x_m\}$ and $Y \in \{y_1, y_2, .., y_n\}$ are discrete random variables with the joint probability mass function $p(x,y)$
  - **Example**: In the text categorization problem, X is a random variable over the set of terms, and Y is a random variable over the set of documents. Evidently, $p(x,y)$ is an m by n matrix, termed a two-dimensional contingency table or a two-way frequency table; this is often estimated from the co-occurrence data.
- **Entropy of X**

$$H(X) = E[-\log_2 p(X)] = -\sum_{x \in \{x_1, x_2, .., x_m\}} p(x) \log_2 p(x)$$

  - A measure of the average amount of information (in bits) required to describe the random variable.
- **Conditional Entropy $H(X|Y)$**
  - Entropy of random variable X given another random variable, Y
- Entropy of a pair of random variables (X,Y) is the entropy of one plus the conditional entropy of the other, i.e.,

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y)$$

We find it convenient to use *ln* instead of log_2

$$Gaussian: H(X) = \log(\sigma\sqrt{2\pi e})$$

Mutual Information (MI)

- **Mutual Information (Information Gain, MI)**
  - Reduction in uncertainty due to knowledge of another random variable

$$
\begin{aligned}
I(X;Y) &= H(X) - H(X \mid Y) = H(Y) - H(Y \mid X) \\
&= H(X) + H(Y) - H(X,Y) = I(Y;X) \\
&= \sum_{x,y} p(x,y) \log_2 \frac{p(x,y)}{p(x)\,p(y)} = \sum_{x} p(x) \sum_{y} p(y \mid x) \log_2 \frac{p(y \mid x)}{p(y)} \\
&= \sum_{y} p(y) \sum_{x} p(x \mid y) \log_2 \frac{p(x \mid y)}{p(x)}
\end{aligned}
$$

  - X and Y are independent, i.e., $p(x,y)=p(x)p(y) \Rightarrow$ mutual information is zero
  - When there is a deterministic one-to-one relationship between X and Y, mutual information is the largest (i.e., equal to the entropy of the random variable)
  - In general, the mutual information is bounded by

$$0 \le I(X;Y) \le \min(H(X), H(Y))$$

Mutual Information Between Gaussian Variables

$$p(\underline{x},\underline{y}) = N\left(\begin{bmatrix}\underline{\mu}_x \\ \underline{\mu}_y\end{bmatrix}; \begin{bmatrix}\Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{xy}^T & \Sigma_{yy}\end{bmatrix}\right)$$

$$\Rightarrow p(\underline{x}) = N(\underline{\mu}_x, \Sigma_{xx});\ p(\underline{y}) = N(\underline{\mu}_y, \Sigma_{yy})$$

$$p(\underline{x} \mid \underline{y}) = N(\underline{\mu}_x + \Sigma_{xy}\Sigma_{yy}^{-1}(\underline{y} - \underline{\mu}_y), \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T)$$

$$I(X;Y) = H(X) + H(Y) - H(X,Y)$$

A greedy forward/ backward feature selection algorithm based on (1) and (2) for regression.

1. **Remove redundant features (large ρ).**
2. **Pick features $\{x\}$ with large correlation with respect to dependent variable (output) y.**

$$H(X) = E_{p(\underline{x})}[-\ln p(\underline{x})] = \frac{1}{2}[n_x \ln(2\pi e) + \ln|\Sigma_{xx}|];\ H(Y) = \frac{1}{2}[n_y \ln(2\pi e) + \ln|\Sigma_{yy}|]$$

$$H(X,Y) = \frac{1}{2}[(n_x + n_y)\ln(2\pi e) + \ln|\Sigma_{xx}| + \ln|\Sigma_{yy} - \Sigma_{xy}^T\Sigma_{xx}^{-1}\Sigma_{xy}|]$$

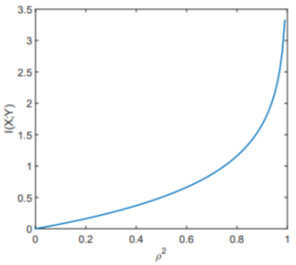
$$= \frac{1}{2}[(n_x + n_y)\ln(2\pi e) + \ln|\Sigma_{yy}| + \ln|\Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T|]$$

$$I(X;Y) = \frac{1}{2}[\ln|\Sigma_{xx}| - \ln|\Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T|] = \frac{1}{2}[\ln|\Sigma_{yy}| - \ln|\Sigma_{yy} - \Sigma_{xy}^T\Sigma_{xx}^{-1}\Sigma_{xy}|] \quad (1)$$

$$= -\frac{1}{2}\ln|I_{\dim(x)} - \Sigma_{xx}^{-1}\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T| = -\frac{1}{2}\ln|I_{\dim(y)} - \Sigma_{yy}^{-1}\Sigma_{xy}^T\Sigma_{xx}^{-1}\Sigma_{xy}|$$

x, y *scalars* : $\Sigma_{xx} = \sigma_x^2; \Sigma_{xy} = \rho_{xy}\sigma_x\sigma_y; \Sigma_{yy} = \sigma_y^2$

$$I(X;Y) = \frac{1}{2}[\ln\sigma_x^2 - \ln(\sigma_x^2(1-\rho_{xy}^2))] = -\frac{1}{2}\ln(1-\rho_{xy}^2) = \ln\frac{1}{\sqrt{1-\rho_{xy}^2}} \quad (2)$$



Mutual Information for Classification

$$z \in \{1,2,..,C\}; \underline{x} \in R^p; p(\underline{x}) = \sum_{i=1}^{C} P[\underline{x}, z=i] = \sum_{i=1}^{C} P(z=i)\, p(\underline{x} \mid z=i) = \sum_{i=1}^{C} \pi_i N(\underline{x}; \underline{\mu}_i, \Sigma_i)$$

$$I(\underline{x}, z) = H(z) - H(z \mid \underline{x}) = H(\underline{x}) - H(\underline{x} \mid z)$$

It is not possible to compute $H(\underline{x})$ or $H(z \mid \underline{x})$ in closed-form. $H(z)$ and $H(\underline{x} \mid z)$ are easy to compute.

$$H(\underline{x} \mid z) = \frac{1}{2} \sum_{i=1}^{C} \pi_i (p \ln(2\pi e) + \ln |\Sigma_i|)$$

Upper Bound on $H(\underline{x})$: $H(\underline{x}) = E_{p(\underline{x})}\{-\ln\left(\sum_{i=1}^{C} \pi_i N(\underline{x}; \underline{\mu}_i, \Sigma_i)\right)\} \le \sum_{i=1}^{C} \pi_i E_{p(\underline{x})}\{-\ln\left(N(\underline{x}; \underline{\mu}_i, \Sigma_i)\right)\}$

$$= \frac{1}{2} \sum_{i=1}^{C} \sum_{j=1}^{C} \pi_i \pi_j [\, p \ln(2\pi) + \ln |\Sigma_i| + (\underline{\mu}_j - \underline{\mu}_i)^T \Sigma_i^{-1} (\underline{\mu}_j - \underline{\mu}_i) + tr(\Sigma_i^{-1} \Sigma_j)]$$

So, $I(\underline{x}, z) \le \frac{1}{2} \sum_{i=1}^{C} \sum_{j=1}^{C} \pi_i \pi_j [(\underline{\mu}_j - \underline{\mu}_i)^T \Sigma_i^{-1} (\underline{\mu}_j - \underline{\mu}_i) + tr(\Sigma_i^{-1} \Sigma_j) - p]$ (1a)

When $\Sigma_i = \Sigma$ for all i, $I(\underline{x}, z) \le \frac{1}{2} \sum_{i=1}^{C} \sum_{j=1}^{C} \pi_i \pi_j [(\underline{\mu}_j - \underline{\mu}_i)^T \Sigma^{-1} (\underline{\mu}_j - \underline{\mu}_i)]$ (1b)

Best single feature: $k = \arg \max_{l \in \{1,2,..,p\}} \frac{1}{2} \left(\sum_{i=1}^{C} \sum_{j=1}^{C} \pi_i \pi_j [\frac{(\mu_{jl} - \mu_{il})^2}{\sigma_l^2}] \right)$ (2a)

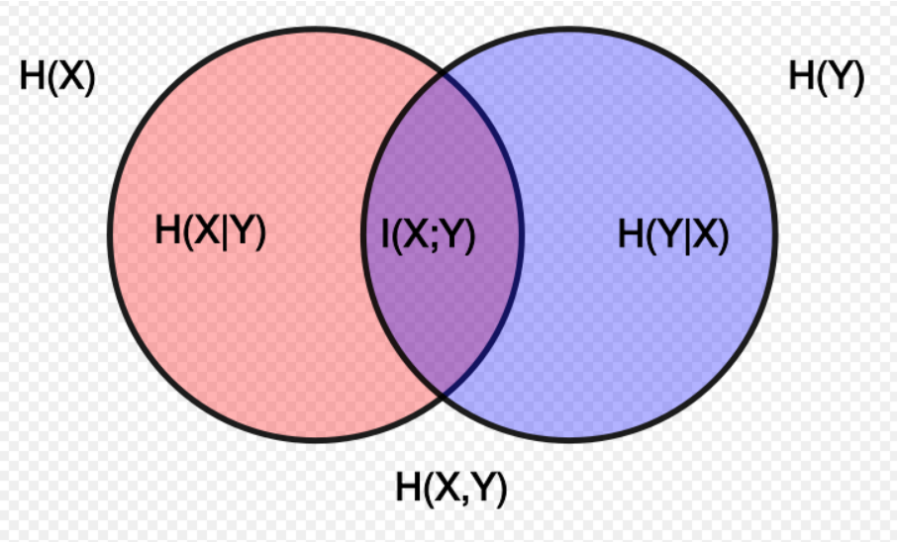
For binary classes: $k = \arg \max_{l \in \{1,2,..,p\}} \left(\frac{(\mu_{1l} - \mu_{2l})^2}{\sigma_l^2} \right) \sim d'^2$ (2b)

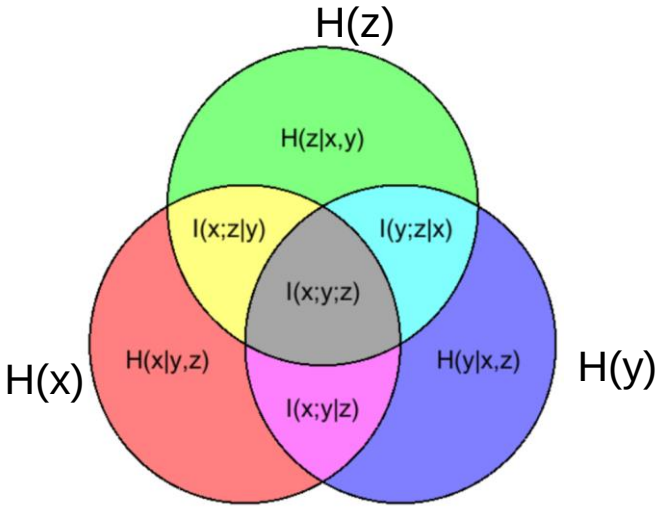
Discriminability measure

A greedy forward/backward feature selection algorithm based on (1) and (2) for classification

Measures of Information: Venn Diagrams

- Information (Venn) Diagram illustrates relationships among measures of information: Entropy, Joint Entropy, Conditional Entropy and MI





$$I(X;Y) = H(X) - H(X \mid Y) = H(Y) - H(Y \mid X)$$
$$= H(X) + H(Y) - H(X,Y) = I(Y;X)$$

$$I(X;Y;Z) = I(X;Y) - I(X;Y \mid Z)$$
$$= I(Y;Z) - I(Y;Z \mid X)$$
$$= I(X;Z) - I(X;Z \mid Y)$$

- $H(X|Y)$ is the novel information in X
- $H(Y|X)$ is the novel information in Y
- $I(X;Y)$ is the Redundant information between X and Y

Joint Mutual Information Between Gaussian Variables - 1

Consider three Gaussian random variables x, y, z

$$p(x,y,z) = N\left(\begin{bmatrix}\mu_x\\ \mu_y\\ \mu_z\end{bmatrix};\begin{bmatrix}\sigma_x^2 & \sigma_{xy} & \sigma_{xz}\\ \sigma_{xy} & \sigma_y^2 & \sigma_{yz}\\ \sigma_{xz} & \sigma_{yz} & \sigma_z^2\end{bmatrix}\right) = N\left(\begin{bmatrix}\mu_x\\ \mu_y\\ \mu_z\end{bmatrix};\begin{bmatrix}\sigma_x^2 & \rho_{xy}\sigma_x\sigma_y & \rho_{xz}\sigma_x\sigma_z\\ \rho_{xy}\sigma_x\sigma_y & \sigma_y^2 & \rho_{yz}\sigma_y\sigma_z\\ \rho_{xz}\sigma_x\sigma_z & \rho_{yz}\sigma_y\sigma_z & \sigma_z^2\end{bmatrix}\right)$$

Recall $\quad I(X;Y) = \frac{1}{2}[\ln|\Sigma_{xx}| - \ln|\Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T|] = \frac{1}{2}[\ln|\Sigma_{yy}| - \ln|\Sigma_{yy} - \Sigma_{xy}^T\Sigma_{xx}^{-1}\Sigma_{xy}|]$

x, y *scalars* : $\Sigma_{xx} = \sigma_x^2; \Sigma_{xy} = \rho_{xy}\sigma_x\sigma_y; \Sigma_{yy} = \sigma_y^2$

$$\Rightarrow I(X;Y) = \frac{1}{2}[\ln\sigma_x^2 - \ln(\sigma_x^2(1-\rho_{xy}^2))] = -\frac{1}{2}\ln(1-\rho_{xy}^2)$$

Recall $I(X;Y;Z) = I(X;Y) - I(X;Y|Z) = -\frac{1}{2}\ln(1-\rho_{xy}^2) \; - I(X;Y|Z)$

Now, $I(X;Y|Z) = H(X|Z) + H(Y|Z) - H(X,Y|Z)$

Simpler one:
$I(X;Y|Z) = H(X|Z) - H(X|Y,Z)$...Try it!

$$p(x|z) = N(\mu_x + \frac{\sigma_{xz}}{\sigma_z^2}(z-\mu_z); \sigma_x^2 - \frac{\sigma_{xz}^2}{\sigma_z^2}); \; p(y|z) = N(\mu_y + \frac{\sigma_{yz}}{\sigma_z^2}(z-\mu_z); \sigma_y^2 - \frac{\sigma_{yz}^2}{\sigma_z^2})$$

$$p(x,y|z) = N\left(\begin{bmatrix}\mu_x + \frac{\sigma_{xz}}{\sigma_z^2}(z-\mu_z)\\ \mu_y + \frac{\sigma_{yz}}{\sigma_z^2}(z-\mu_z)\end{bmatrix};\begin{bmatrix}\sigma_x^2 & \sigma_{xy}\\ \sigma_{xy} & \sigma_y^2\end{bmatrix} - \frac{1}{\sigma_z^2}\begin{bmatrix}\sigma_{xz}\\ \sigma_{yz}\end{bmatrix}\begin{bmatrix}\sigma_{xz} & \sigma_{yz}\end{bmatrix}\right)$$

Joint Mutual Information Between Gaussian Variables - 2

$$I(X;Y\mid Z)=\frac{1}{2}[\ln(2\pi e)+\ln(\sigma_x^2-\frac{\sigma_{xz}^2}{\sigma_z^2})+\ln(2\pi e)+\ln(\sigma_y^2-\frac{\sigma_{yz}^2}{\sigma_z^2})$$

$$-2\ln(2\pi e)-\ln|\begin{bmatrix}\sigma_x^2 & \sigma_{xy}\\ \sigma_{xy} & \sigma_y^2\end{bmatrix}-\frac{1}{\sigma_z^2}\begin{bmatrix}\sigma_{xz}\\ \sigma_{yz}\end{bmatrix}\begin{bmatrix}\sigma_{xz} & \sigma_{yz}\end{bmatrix}|]$$

$$=\frac{1}{2}\ln\frac{(\sigma_x^2-\frac{\sigma_{xz}^2}{\sigma_z^2})(\sigma_y^2-\frac{\sigma_{yz}^2}{\sigma_z^2})}{\sigma_x^2\sigma_y^2-\left(\frac{\sigma_{xz}^2\sigma_y^2+\sigma_{yz}^2\sigma_x^2}{\sigma_z^2}\right)+\frac{\sigma_{xz}^2\sigma_{yz}^2}{\sigma_z^4}-(\sigma_{xy}-\frac{\sigma_{xz}\sigma_{yz}}{\sigma_z^2})^2}$$

$$=\frac{1}{2}\ln\frac{(1-\rho_{xz}^2)(1-\rho_{yz}^2)}{\underbrace{1-\rho_{xy}^2-\rho_{yz}^2-\rho_{xz}^2+2\rho_{xy}\rho_{xz}\rho_{yz}}_{\det er\min ant\ of\ 3x3\ correlation\ coeff.matrix\ \geq 0`}}$$

$I(X;Y\mid Z)=0$ if $\rho_{xz}=1$
and $I(X;Y;Z)=I(X;Y)$

So, $I(X;Y;Z)=I(X;Y)-I(X;Y\mid Z)=-\frac{1}{2}\ln\frac{(1-\rho_{xy}^2)(1-\rho_{xz}^2)(1-\rho_{yz}^2)}{1-\rho_{xy}^2-\rho_{yz}^2-\rho_{xz}^2+2\rho_{xy}\rho_{xz}\rho_{yz}}$

By symmetry, $I(X;Z\mid Y)=\frac{1}{2}\ln\frac{(1-\rho_{xy}^2)(1-\rho_{yz}^2)}{1-\rho_{xy}^2-\rho_{yz}^2-\rho_{xz}^2+2\rho_{xy}\rho_{xz}\rho_{yz}}$

$$I(Y;Z\mid X)=\frac{1}{2}\ln\frac{(1-\rho_{xy}^2)(1-\rho_{xz}^2)}{1-\rho_{xy}^2-\rho_{yz}^2-\rho_{xz}^2+2\rho_{xy}\rho_{xz}\rho_{yz}}$$

Kullback-Leibler (KL) Divergence

- **Kullback-Leibler (K-L) Divergence ("Relative Entropy")**
  - A measure of the distance between two pmfs, *p(x)* and *q(x)*

$$KL(p(x)\,||\,q(x)) = D(p(x)\,||\,q(x)) = E_{p(x)}[\log_2 \frac{p(X)}{q(X)}] = \sum_{x\in\{x,x_2,..,x_{m1}\}} p(x)\log_2 \frac{p(x)}{q(x)}$$

  - A measure of inefficiency of assuming that the pmf is $q(x)$ when the true pmf is *p(x)*

Compute for Bernoulli p and q

  - K-L divergence is non-negative, but not necessarily symmetric

- **Relation to Mutual Information**

$$I(X,Y) = E_{p(x,y)}[\log_2 \frac{p(X,Y)}{p(X)\,p(Y)}] = D(p(x,y)\,||\,p(x)\,p(y)) = \sum_{x,y} p(x,y)\log_2 \frac{p(x,y)}{p(x)\,p(y)}$$

$$= \sum_x p(x) \sum_y p(y\,|\,x)\log_2 \frac{p(y\,|\,x)}{p(y)} = E_{p(x)}[D(p(y\,|\,x)\,||\,p(y)] = E_{p(y)}[D(p(x\,|\,y)\,||\,p(x)]$$

This is very useful in Variational Bayes approximation where we approximate *p(x,y)* by the best possible *q(x) q(y)* by minimizing the KL divergence

K-L Divergence Between Gaussian PDFs

$$KL(p \,\|\, q) = -\int p(x) \ln\{\frac{q(x)}{p(x)}\} dx = -E_p\{\ln q(x)\} - H_p(x)$$

$$p(x) = N(\mu, \sigma^2); q(x) = N(m, s^2)$$

$$\ln p(x) = -\frac{1}{2}\ln(2\pi) - \ln\sigma - \frac{(x-\mu)^2}{2\sigma^2}$$

$$\ln q(x) = -\frac{1}{2}\ln(2\pi) - \ln s - \frac{(x-m)^2}{2s^2}$$

$$\Rightarrow KL(p \,\|\, q) = -E_p\{\ln\frac{\sigma}{s} + \frac{(x-\mu)^2}{2\sigma^2} - \frac{(x-m)^2}{2s^2}\} = \ln\frac{s}{\sigma} - \frac{1}{2} + \frac{\mu^2 + \sigma^2 - 2\mu m + m^2}{2s^2}$$

$$= \ln\frac{s}{\sigma} - \frac{1}{2} + \frac{(\mu - m)^2 + \sigma^2}{2s^2} = \frac{1}{2}[\ln\frac{s^2}{\sigma^2} + \frac{(\mu - m)^2}{s^2} + \frac{\sigma^2}{s^2} - 1]$$

Multivariate case (n vectors): $\frac{1}{2}[\ln\frac{\det(S)}{\det(\Sigma)} + (\underline{\mu} - \underline{m})^T S^{-1}(\underline{\mu} - \underline{m}) + tr(S^{-1}\Sigma) - n]$

Minimum when $\underline{\mu} = \underline{m}$ and $S = \Sigma \Rightarrow$ moment matching; $p(x)=q(x)$

K-L Divergence and MI for MVN Variables

$$p(\underline{x},\underline{y}) = N(\begin{bmatrix}\underline{\mu}_x \\ \underline{\mu}_y\end{bmatrix};\begin{bmatrix}\Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{xy}^T & \Sigma_{yy}\end{bmatrix})$$

$$\Rightarrow p(\underline{x}) = N(\underline{\mu}_x, \Sigma_{xx}); p(\underline{y}) = N(\underline{\mu}_y, \Sigma_{yy})$$

$$p(\underline{x}\,|\,\underline{y}) = N(\underline{\mu}_x + \Sigma_{xy}\Sigma_{yy}^{-1}(\underline{y}-\underline{\mu}_y), \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T)$$

$$I(X;Y) = E_{p(\underline{y})}[KL(p(\underline{x}\,|\,\underline{y}) \,\|\, p(\underline{x}))]$$

$$= \frac{1}{2}E_{p(\underline{y})}[\ln\frac{|\Sigma_{xx}|}{|(\Sigma_{xx}-\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T)|} + \|\Sigma_{xy}\Sigma_{yy}^{-1}(\underline{y}-\underline{\mu}_y)\|_{\Sigma_{xx}^{-1}} + tr(\Sigma_{xx}^{-1}(\Sigma_{xx}-\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T)) - \dim(x)]$$

$$= \frac{1}{2}[\ln\frac{|\Sigma_{xx}|}{|(\Sigma_{xx}-\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T)|} + tr(\Sigma_{yy}^{-1}\Sigma_{xy}^T\Sigma_{xx}^{-1}\Sigma_{xy}) + tr(I_{\dim(x)} - \Sigma_{xx}^{-1}\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T) - dim(x)]$$

$$= \frac{1}{2}\ln\frac{|\Sigma_{xx}|}{|(\Sigma_{xx}-\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T)|} = -\frac{1}{2}\ln|(I_{\dim(x)} - \Sigma_{xx}^{-1}\Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{xy}^T)| \quad \because tr(AB) = tr(BA)\ \&\ tr(I_{\dim(x)}) = \dim(x)$$

$$x, y\ scalars: \Sigma_{xx} = \sigma_x^2; \Sigma_{xy} = \rho_{xy}\sigma_x\sigma_y; \Sigma_{yy} = \sigma_y^2$$

$$I(X;Y) = -\frac{1}{2}[\ln(1-\rho_{xy}^2))] = \ln\frac{1}{\sqrt{1-\rho_{xy}^2}}$$

$$I(X;Y\,|\,Z) = E_{p(z)}\{KL(p(x,y\,|\,z) \,\|\, p(x\,|\,z)p(y\,|\,z))\}$$

Try it for Gaussian!

Classification vs. Regression

- Classification vs. Regression
  - Set of input variables $\underline{x}$ ……. *observed data/features*
  - Target variables $\underline{z}$ …. *What you are interested in inferring*

$\underline{z}$ → classification → Values from a discrete set: 0,1,2,...,C-1

$\underline{z}$ → regression → Continuous variables

$\underline{x}$ can be discrete or continuous *features* or *data*

The goal is to evaluate:

♦ In classification $P(z = k|\underline{x})$ $\Rightarrow$ It is a probability (posterior probability)

♦ In regression $p(\underline{z}|\underline{x})$ $\Rightarrow$ It is a density (posterior density)

Binary Classification

- Let us consider a binary classification case first:

$$z \in (0,1)$$

Likelihood Function

$$P(z=1|\underline{x}) = \frac{p(\underline{x}|z=1)P(z=1)}{p(\underline{x})}$$

(Here, $\underline{x}$ is continuous.)

$$= \frac{p(\underline{x}|z=1)P(z=1)}{p(\underline{x}|z=1)P(z=1) + p(\underline{x}|z=0)P(z=0)}$$

Total Probability Theorem

$$= \frac{1}{1 + \frac{p(\underline{x}|z=0)}{p(\underline{x}|z=1)}\frac{P(z=0)}{P(z=1)}}$$

For $a>0, b>0$

$$\left(\frac{a}{b}\right)^d = e^{d\ln\left(\frac{a}{b}\right)} = e^{-d\ln\left(\frac{b}{a}\right)}$$

$$= \frac{1}{1+e^{-[\ln\frac{p(\underline{x}|z=1)}{p(\underline{x}|z=0)} + \ln\frac{P(z=1)}{P(z=0)}]}}$$

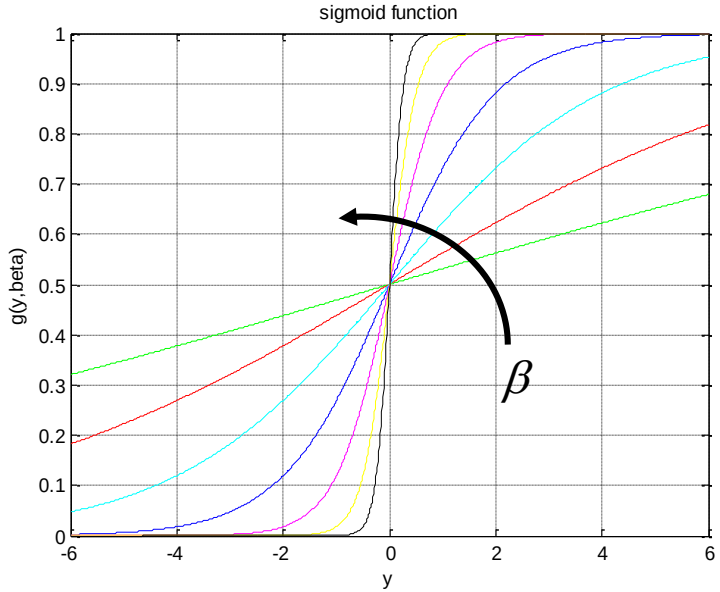
$\ln \frac{p(\underline{x}|z=1)}{p(\underline{x}|z=0)}$ - - Likelihood Ratio

$\ln \frac{P(z=1)}{P(z=0)}$ - - Prior Odds Ratio

Sigmoid Function

$$p(z=1|\underline{x}) = \frac{1}{1+e^{-y}} = g(y) \quad \textbf{where} \quad y = \ln \frac{p(\underline{x}|z=1)}{p(\underline{x}|z=0)} + \ln \frac{P(z=1)}{P(z=0)}$$

If $y = \underline{w}^T \underline{x}$, then this corresponds to a **simple neuron** with a **logistic sigmoid nonlinearity**. More generally,

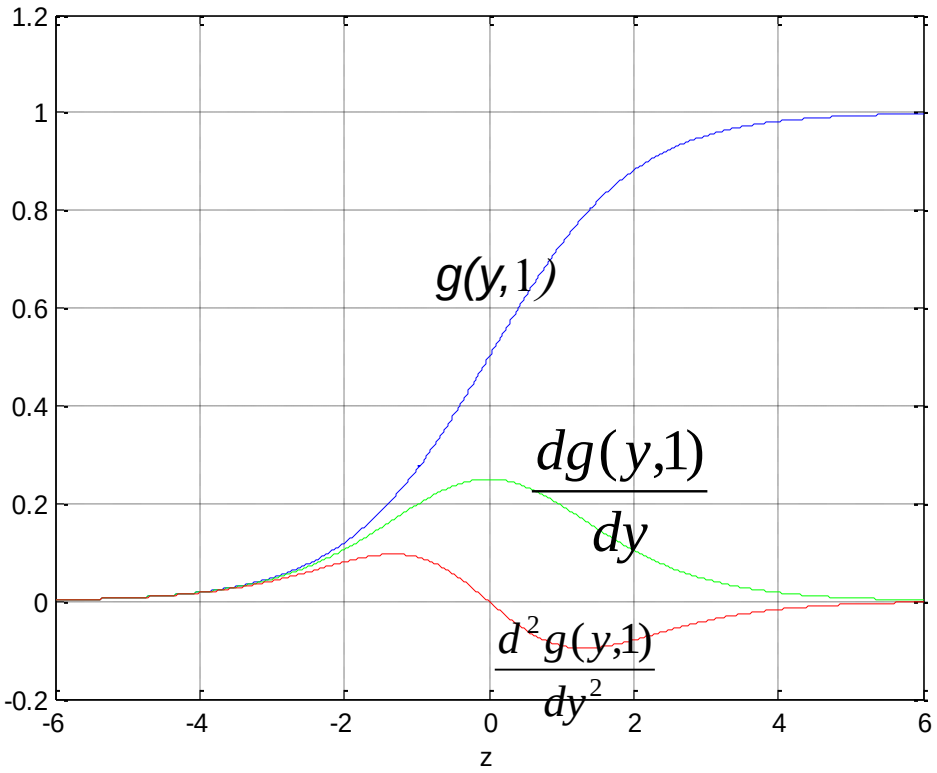


$$g(y,\beta) = \frac{1}{1+e^{-\beta y}}$$

- What happens as $\beta \to \infty$ or $\beta \to 0$?
- Is $g(y,\beta)$ convex? **NO.** Why?

We will show that for the so-called *exponential family* of densities, this form for $y = \underline{w}^T \underline{x}$ is valid.

First and Second Derivative of Sigmoid



$$g(y,\beta) = \frac{1}{1+e^{-\beta y}}$$

$$\frac{dg(y,\beta)}{dy} = \beta g(1-g)$$

$$\frac{d^2 g(y,\beta)}{dy^2} = \beta^2 g(1-g)(1-2g)$$

$\frac{dg(y,\beta)}{dy}$ nonnegative $\Rightarrow$ monotonic in y

$\frac{d^2 g(y,\beta)}{dy^2}$ can be positive or negative $\Rightarrow$ not convex

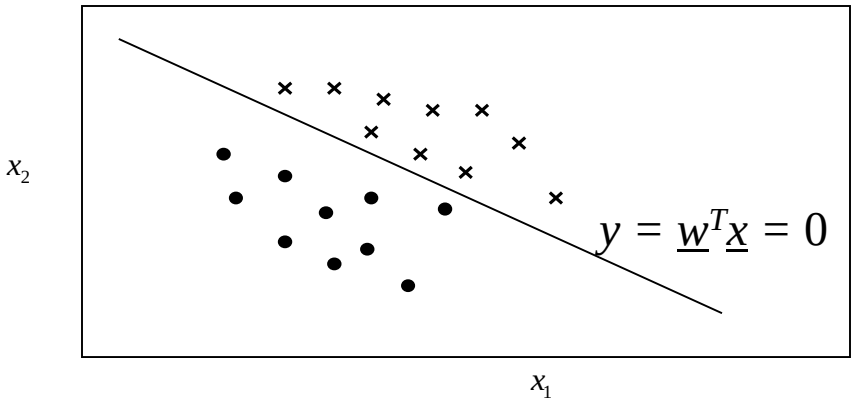
Discriminant Function

- y is also called the *discriminant function*.

$$p(z=1|\underline{x}) = \frac{1}{1+e^{-y}} = g(y)$$

Any function that can be used to decide a class membership

$$y = 0 \quad \Rightarrow \quad P(z=1|\underline{x}) = P(z=0|\underline{x}) \quad \Rightarrow \text{ decision boundary}$$



- Even for a non-exponential family, it is OK to use a logistic function, but they have a nonlinear form of y

$$y = g(\underline{x}, \underline{w}) \quad \Rightarrow \quad \begin{matrix}\text{Multi - layer Perceptrons (MLP)} \\ \text{Radial Basis Functions (RBF)}\end{matrix}$$

Regression

- Regression

$p(\underline{z} \mid \underline{x}) \Rightarrow$ MMSE estimate is the **conditional mean**.

$$E(\underline{z} \mid \underline{x}) = \int_{\underline{z}} \underline{z}\, p(\underline{z} \mid \underline{x})\, d\underline{z}$$

Discuss LS Problem from HW 1

- For Linear Regression: $E(\underline{z} \mid \underline{x}) = W\underline{x} = \underline{y} = \hat{\underline{z}}$

$$\hat{\underline{z}} = \underline{y} = \sum_{j=0}^{p} \underline{w}_j x_j \Rightarrow \hat{z}_i = y_i = \sum_{j=0}^{p} w_{ij} x_j ; i = 1,2,..,m$$

  - ADALINE: Adaptive Linear Element

- In general, we use nonlinear regression functions

$$\underline{y} = \sum_{j=0}^{p} \underline{w}_j \phi_j(\underline{x}) \Rightarrow y_i = \sum_{j=0}^{p} w_{ij} \phi_j(\underline{x}); i = 1,2,..,m$$

* How to select basis functions or kernels, $\phi_j(\underline{x})$?

RBF Networks & MLP

❑ RBF Networks

Radial basis function networks using Gaussian mixtures

$$\phi_j(\underline{x}) = \exp\{-\frac{1}{2}(\underline{x}-\underline{\mu})^T \Sigma^{-1}(\underline{x}-\underline{\mu})\}$$

- Select $\underline{\mu}_j$, Σ_j
- Optimize over $\{w_{ij}\}$

$\Rightarrow$ Local Approximation

❑ MLP: Generalize single layer perceptron

$$y_i(\underline{x}) = g\{\sum_{j=1}^{M} w_{ij}\ g(\underline{\tilde{w}}_j^{\ T}\ \underline{x})\} \qquad \underline{x} = [-1, x_1, x_2, \ldots . x_p]^T$$

$$= g\{w_i^{\ T}\ \underline{g}(\underline{x}, \underline{\tilde{w}})\}$$

Model of this form can approximate continuous functions to arbitrary accuracy -- universal function approximators.

Example Nonlinearities - 1

- Rectifier or rectified linear unit (ReLU). Popular in deep learning

$$g(y) = max(0, y) = (y)^+$$

- Hyperbolic tangent

$$g(y) = tanh(y) = (e^{y}-e^{-y}) / (e^{y}+e^{-y}) = (e^{2y}-1)/(e^{2y}+1) = \sigma(2y) - \sigma(-2y)$$

- Sigmoid

$$g(y) = 1/(1 + e^{-y}) = \sigma(y)$$

- Softmax (for multiple classes as an output unit)

$$g(y_i) = e^{y_i} / \sum_{k=1}^{C} e^{y_k} ; \sum_{i=1}^{C} g(y_i) = 1; g(y_i) > 0$$

- RBF unit: used with features in SVM, RBF,…

$$\phi_j(\underline{x}) = \exp\{-\frac{1}{2}(\underline{x} - \underline{\mu})^T \Sigma^{-1} (\underline{x} - \underline{\mu})\}$$

Example Nonlinearities - 2

- Softplus (function approximation and Restricted Boltzmann Machine (RBM))

$$g(y) = ln(1+e^{y}) = -\ln \sigma(-y) = -\ln(1- \sigma(y))$$

soft approximation to $y^{+} = \max(0,y)$

- Hard *tanh* or unit saturation block

$$g(y) = max(-1, min\ (1,y))$$

- Absolute value

$$g(y) = |y|$$

- Maxout (used with features)

$$g(\underline{x}) = \max_{i}(\underline{w}_i^T\ \underline{x} + w_{oi})$$

RBF Networks & MLP

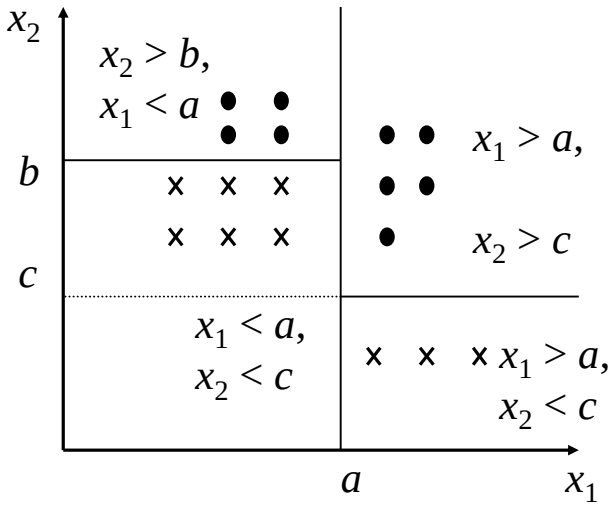
These can be extended to multiple layers → RBF: Mixture of exponentials; MLP

RBF and MLP belong to the category of *Feedforward Neural Networks* ⇒ Acyclic Graphs

⊗ Can we get insight into what is going on within MLP and RBF?

⊗ Can we extract rules from MLP and RBF?

⊗ Alternately, can we represent y_i as a function of non-overlapping basis functions



⇓

- Decision tree classifiers
- Instance-based classifiers (case-based AI reasoning)

Recurrent NNs & Density Estimation

- Recurrent Neural Networks:

  What if NNs have cycles? ($\Rightarrow$ recurrent Neural Networks)

  - Hopfield NNs
  - Boltzmann and Helmholtz Machines
  - Dynamic Neural Networks
  - Long Short-term Memory Networks (LSTMs)

- Density Estimation:

$$p(\underline{z} \mid \underline{x}) = \frac{p(\underline{x}, \underline{z})}{p(\underline{x})} = \frac{p(\underline{x} \mid \underline{z})\, p(\underline{z})}{p(\underline{x})}$$

$$p(\underline{x}) = \int_{\underline{z}} p(\underline{x}, \underline{z}) d\,\underline{z} \text{ for continuous } \underline{z}; \text{ (or) } \sum_{\underline{z}} p(\underline{x}, \underline{z}) \text{ for discrete } \underline{z}$$

Two key questions:

- Should we model $p(\underline{x},z)$ *or* $p(z|\underline{x})$ *directly?*
- What are the weights? How do you learn them?

Generative versus Discriminative

Generative

- Models $p(\underline{x}, \underline{z})$ from which $p(\underline{x})$, $p(\underline{z})$ and $p(\underline{x} \mid \underline{z})$ follow
- Easy to fit
- Easy to add new classes
- Handles missing data easily
- Handles unlabeled training data
- Can infer $p(\underline{x} \mid \underline{z})$ … inverse problem
- Hard to preprocess features
- Can give extreme predictions
- Less accurate in practice

Discriminative

- Models $p(\underline{z} \mid \underline{x})$ directly … flexibility
- Needs optimization
- Retrain to add new classes
- Not easy to handle missing data
- Harder to handle unlabeled training data
- Not possible to infer $p(\underline{x} \mid \underline{z})$
- Can preprocess features (e.g., kernels)
- Provides calibrated predictions
- More accurate in practice

Generative & Discriminative Methods

Generative

- Classifiers
  - Discriminant Analysis (DA)
  - Naïve Bayes (NB)
  - Tree-augmented NB
  - K-nearest neighbor
  - Infinite Mixture DA

Discriminative

- Classifiers (**C**) & Regression (**R**)
  - Logistic Regression (LR).. **C** only
  - Sparse Linear/Logistic Regression
  - Mixture of experts
  - MLP/Deep Neural Networks/RBF
  - Conditional Random Fields (CRF) ..**C**
  - Decision & Regression Trees
  - Boosting
  - Sparse kernelized linear/logistic
  - Relevance vector machine (RVM)
  - Support vector machine (SVM)
  - Gaussian processes (GP)
  - Smoothing splines … **R** only

Popular Models of $p(\underline{x}|\underline{z})$ and $p(\underline{z})$

- $p(\underline{x})$ is a Gaussian mixture

  $p(\underline{x}\,|\,\underline{z})$ is multivariate Gaussian and $p(\underline{z})$ is discrete

- $p(\underline{x})$ is a mixture of multinomials

  $p(\underline{x}\,|\,\underline{z})$ is a product of discrete distributions and $p(\underline{z})$ is discrete

- Factor analysis

  $p(\underline{x}\,|\,\underline{z})$ is a product of Gaussians and $p(\underline{z})$ is a product of Gaussians

- Probabilistic independent component analysis (ICA)

  $p(\underline{x}\,|\,\underline{z})$ is a product of Gaussians and $p(\underline{z})$ is a product of Laplace

- Multinomial PCA

  $p(\underline{x}\,|\,\underline{z})$ is a product of discrete and $p(\underline{z})$ is a product of Gaussian

- Sigmoid belief net

  $p(\underline{x}\,|\,\underline{z})$ is a product of Bernoulli and $p(\underline{z})$ is a product of Bernoulli

MAP & ML Learning

So, among the learning processes, the difference is only in the data used for learning.

D = data

MAP: $\underline{w} = \arg \max_{\underline{w}} p(\underline{w} \mid D)$

ML: $\underline{w} = \arg \max_{\underline{w}} p(D \mid \underline{w})$

$$p(\underline{w} \mid D) = \frac{p(D \mid \underline{w})\, p(\underline{w})}{p(D)}$$

$$J(\underline{w}) = -\ln p(\underline{w} \mid D)$$

$$= -\ln p(D \mid \underline{w}) - \ln p(\underline{w}) + \underbrace{\ln p(D)}_{\text{Independent of } \underline{w}}$$

$$J(\underline{w}) = -\ln p(D \mid \underline{w}) - \ln p(\underline{w})$$

$p(\underline{w})$ *uniform* $\Rightarrow MAP = ML$

Supervised Learning

For Supervised Learning (assuming independent data sets)

$$p(D \mid \underline{w}) = \prod_{n=1}^{N} p(z_n, \underline{x}_n \mid \underline{w})$$

$$= \prod_{n=1}^{N} p(z_n \mid \underline{x}_n, \underline{w})\, p(\underline{x}_n \mid \underline{w})$$

$$= \prod_{n=1}^{N} p(z_n \mid \underline{x}_n, \underline{w})\, p(\underline{x}_n)$$

$$-\ln[\, p(D \mid \underline{w})] = -\sum_{n=1}^{N} \{\ln\, p(z_n \mid \underline{x}_n, \underline{w}) + \ln\, p(\underline{x}_n)\}$$

$$J(\underline{w}) = -\ln\, p(D \mid \underline{w}) = \frac{1}{2} \sum_{n=1}^{N} \| z_n - g(\underline{x}_n, \underline{w}) \|^2 \Leftarrow \text{Gaussian Unity Covariance}$$

Thus, determining weights in supervised learning is a function minimization problem!

Classification Problem

For classification problems:

$$-\ln P(z_n \mid \underline{x}_n, \underline{w}) = -z_n \ln \underbrace{P(z_n = 1 \mid \underline{x}_n, \underline{w})}_{g(y_n)} - (1 - z_n) \ln \underbrace{P(z_n = 0 \mid \underline{x}_n, \underline{w})}_{1-g(y_n)}$$

$$= -z_n \ln g(y_n) - (1 - z_n) \ln(1 - g(y_n))$$

Cross-entropy error function

$$g(y_n) = P(z_n = 1 \mid \underline{x}_n, \underline{w}) = \frac{1}{1 + e^{-\beta y_n}}; y_n = \underline{w}^T \underline{x}_n$$

Cross-entropy is related to KL-divergence:

Data Distribution: $p = z_n \in \{0,1\}$; Classifier output: $q = \{g(y_n), 1 - g(y_n)\}$

Cross-entropy $H_n(p,q) = -z_n \ln g(y_n) - (1 - z_n) \ln(1 - g(y_n))$

$$= -z_n \ln z_n - (1 - z_n) \ln(1 - z_n)$$

$$+ [z_n \ln \frac{z_n}{g(y_n)} + (1 - z_n) \ln \frac{(1 - z_n)}{(1 - g(y_n))}]$$

$$= H_n(p) + KL(p \| q)$$

Gradient Computation -1

Cross-entropy function over all samples:

$$J(\underline{w}) = \sum_{n=1}^{N} H_n(p,q) = -\sum_{n=1}^{N} [z_n \ln g(y_n) + (1-z_n)\ln(1-g(y_n))]$$

$$g(y_n) = P(z_n = 1 \mid \underline{x}_n, \underline{w}) = \frac{1}{1+e^{-\beta y_n}}; y_n = \underline{w}^T \underline{x}_n$$

$$\frac{\partial g(y_n)}{\partial y_n} = \frac{\partial}{\partial y_n}\left(\frac{1}{1+e^{-\beta y_n}}\right) = \frac{\beta e^{-\beta y_n}}{(1+e^{-\beta y_n})^2} = \beta g(y_n)(1-g(y_n)) = \beta g_n(1-g_n)$$

Gradient has a nice (and unified) form: for classification

$$\nabla_{\underline{w}} J = -\sum_{n=1}^{N} \left[\frac{z_n}{g_n}\frac{\partial g_n}{\partial y_n}\nabla_{\underline{w}} y_n - \frac{(1-z_n)}{(1-g_n)}\frac{\partial g_n}{\partial y_n}\nabla_{\underline{w}} y_n\right]$$

$$= -\sum_{n=1}^{N} \frac{z_n - g_n}{g_n(1-g_n)}\frac{\partial g_n}{\partial y_n}\nabla_{\underline{w}} y_n = -\beta\sum_{n=1}^{N}(z_n - g_n)\nabla_{\underline{w}} y_n$$

Gradient Computation -2

So, $\nabla_{\underline{w}} J = -\beta \sum_{n=1}^{N} \{z_n - g_n)\nabla_{\underline{w}} y_n$

If $y_n = \underline{w}^T \underline{x}_n$ $\quad \nabla_{\underline{w}} J = -\beta \sum_{n=1}^{N} (z_n - g_n)\underline{x}_n$

$\Rightarrow$ Calculation of gradient is simple

$\Rightarrow$ Note the summation

$\Rightarrow$ With minor variations, leads to back propagation algorithm

Gradient has a nice (and unified) form: for regression

$$\nabla_{\underline{w}} J = -\sum_{n=1}^{N} \{z_n - g(\underline{x}_n, \underline{w})\}\nabla_{\underline{w}} g = -\beta \sum_{n=1}^{N} (z_n - g_n) g_n (1 - g_n)\underline{x}_n$$

Minor difference with classification case. *Derivative of sigmoid* is the extra term in the regression case and that is a problem! Why?

Hessian

- Classification

$$\nabla_{\underline{w}} J = -\beta \sum_{n=1}^{N} \{z_n - g_n)\underline{x}_n \Rightarrow \nabla^2_{\underline{w}} J = \beta^2 \sum_{n=1}^{N} g_n(1-g_n)\underline{x}_n \underline{x}_n^T$$

Cross-entropy error function is convex. Unique minimum

- Regression

$$\nabla_{\underline{w}} J = -\beta \sum_{n=1}^{N} (z_n - g_n) g_n (1-g_n)\underline{x}_n$$

$$\nabla^2_{\underline{w}} J = \beta^2 \sum_{n=1}^{N} \left[(z_n - g_n) g_n (1-g_n)(2g_n - 1) + \left(g_n(1-g_n) \right)^2 \right] \underline{x}_n \underline{x}_n^T$$

This is not necessarily convex!

Optimization Algorithms - 1

- Optimization Algorithms

$$\underline{w}_0 \rightarrow \underline{w}_1 \rightarrow \ldots\ldots\ldots\ldots \rightarrow \underline{w}^*$$

$$\underline{w}_{k+1} = \underline{w}_k - \eta H \nabla_w J$$

$$H = \begin{cases} I & \Rightarrow \text{SD or Gradient method} \\ [\nabla_w^2 J]^{-1} & \Rightarrow \text{Newton} \end{cases}$$

H PD →
- QN
- CG
- Levenberg-Marquardt $(\nabla_w J \nabla_w J^T + \nu I)^{-1}$

$$\approx (\nabla_w^2 J + \nu I)^{-1}$$

Optimization Algorithms - 2

- Batch versus incremental algorithms
- Would like to estimate $Var(y_i)$
- Prune connections to simplify the architecture

⊗ How long to converge, how many training patterns,…

- Optimization for ill-conditioned problems (ridge regression)

  Use regularization

$$\tilde{J} = J + \frac{1}{2}\nu \underline{w}^T \underline{w}$$

Regularization combats overfitting

$$\Rightarrow \nabla^2 \tilde{J} = \nabla_w^2 J + \nu I \quad \ldots\ldots \text{(Levenberg-Marquardt)}$$

- L_1-regularization (good for feature selection)

$$\tilde{J} = J + \nu \| \underline{w} \|_1$$

- Basis pursuit denoising (BPDN)
- Compressed sensing
- Lasso (Least Absolute Shrinkage & Selection Operator)

Optimization Algorithms - 3

- Combined L_1- and L_2-regularization

$$\tilde{J} = J + \nu_1 \| \underline{w} \|_1 + \frac{1}{2} \nu_2 \underline{w}^T \underline{w}$$

Elastic Net

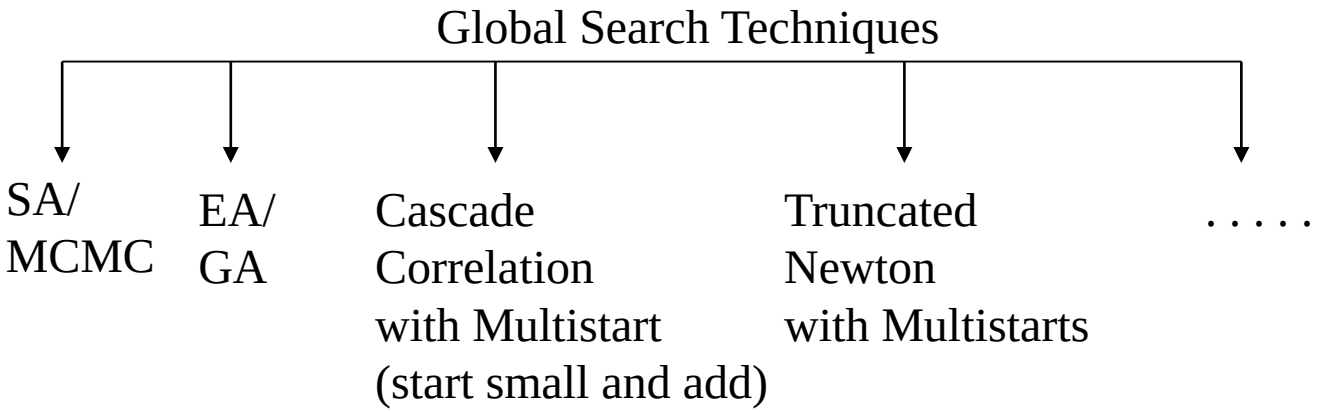
- Bridge regression

$$\tilde{J} = J + \nu_1 \sum_i | w_i |^b ; b > 0$$

- Coordinate descent or probabilistic relaxation methods (EM, variational Bayes, belief propagation, empirical Bayes (Type II ML,…))

Optimization Algorithms - 4

- Numerous local minima ⇒ use global search techniques



Unsupervised Learning	Reinforcement Learning
Density Estimation Non-parametric techniques Kohonen SOFM, LVQ, ART Clustering (K-means, GMM, Hierarchical,…)	Application of Approximate Dynamic Programming

Summary

- Provide a systematic account of the major topics in ML based *on fundamental mathematical principles*
- Focus on pattern classification and optimization applications
- Treat classification, regression, density estimation problems in a unified way
- Generative & Discriminative classification & regression
- Learning from data: Supervised, unsupervised, semi-supervised, reinforcement, active
- Use of a variety of optimization techniques for learning weights