IS-ZC464: MACHINE LEARNING

Lecture-01: Introduction to ML

Dr. Kamlesh Tiwari
Assistant Professor
Department of Computer Science and Information Systems,
BITS Pilani, Pilani, Jhunjhunu-333031, Rajasthan, INDIA

July 28, 2018 (WILP @ BITS-Pilani Jul-Nov 2018)

Introduction: The Machine

In 1936 **Alan Turing** proposed an abstract model of computation called as Turing Machine.[1]

First version of most hyped machine the **computer** became available in 1945. Currently we have its forth generation.

What for this machine is good?

Computation Computation Computation

Computation Computation

[1] ChurchTuring thesis hypothesizes that anything that can be "computed" can be computed by some Turing machine

Computational Problems

We have problems (the computational ones)

- Sorting: Arranging numbers in ascending/descending order
- Searching: Finding whether an item has specified key
- Determining the existence of Hamiltonian circuit (traversing every vertex once) or Euler walk (through every edge) in a graph

If we know how to solve the problem, then we could write a program

But, for some problems we don't precisely know

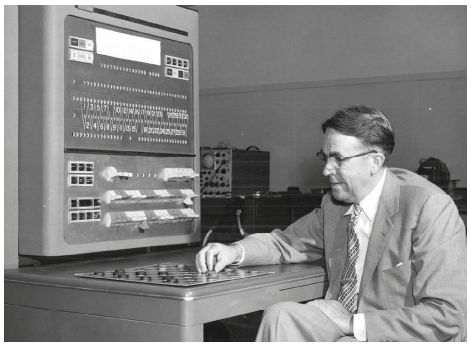
- Is there a cat in figure? which cat?
- What is written on the board? Which language it is in?
- How to ask for a help from foreigner *etc*.

Either 1) we don't know how to solve, or 2) difficult to specify solution procedure

Then we go for **Machine Learning** (ML)

Similar problem was faced by Arthur Samuel

- When in 1956 he wanted to develop a Checkers playing program that could beat him
- Idea was to let the computer play a lot of games against itself and learn
- In 1962 the computer won over human player
- Father of ML



Defined ML as a field of study that gives computers the ability to learn without being explicitly programmed.

Machine Learning: Definition

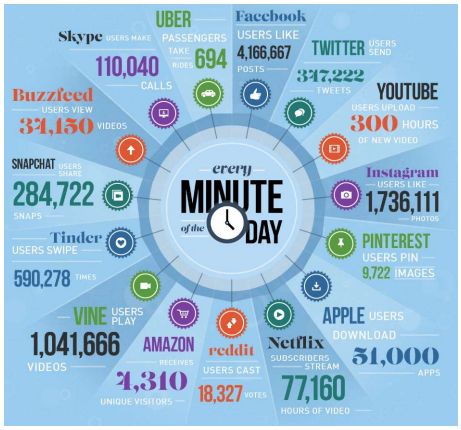
Definition: Tom Mitchell (1998)

a computer program is said to *learn* from experience **E** with respect to some *task* **T** and some performance measure **P**, if its performance on **T**, as measured by **P**, improves with experience **E**.

Performance of an algorithm does not solely depends on itself instead it takes into account of 1) training data, and 2) training methodology.

Data + Compute-Power

We are drowning in data.



What we do with the data

- Classification
- Regression
- Clustering
- Association Rule Mining
- Sequence Discovery

90% generated in last 2 year and 80% of it is unstructured [2].

[2] http://wersm.com/how-much-data-is-generated-every-minute-on-social-media/

Machine Learning: Tasks

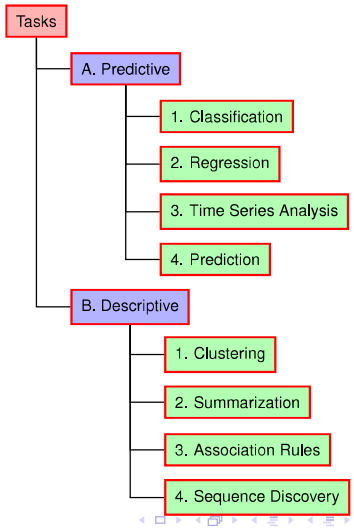
Two broad categories of machine learning models are *Predictive* and *Descriptive*. Some of the related tasks are

Predictive

focuses towards new data item or expanding beyond known facts

Descriptive

focus to understand the available data



Classification

Classification maps data into *predefined* labels.

Example: Lots of mails are there in my mail box. Can you tell me which are SPAM?

- Task of supervised learning
- Often based on some patterns or characteristics
- We can use the frequency of words
- Assumption is that some words appears more or less frequently in a SPAM mail

Regression

Regression is used to map data into *real valued* variable.

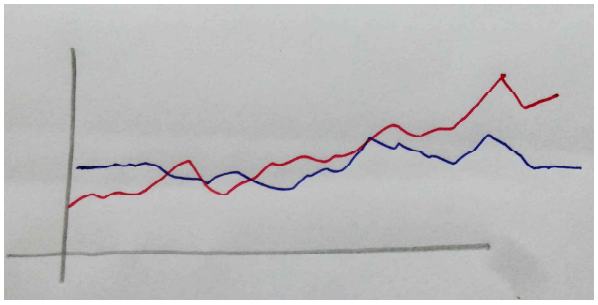
Example: What is the cost of my house?

- Task of supervised learning
- We have data about the cost of house based on features such as
  - location
  - Plot area
  - number of rooms
  - garden available or not
  - how old it is
- Current economical conditions can also matter
- Dimensionality is high

Time Series Analysis

In time series analysis the value of attribute is examined over time.

Example: Which stock is more profitable?



- The values are obtained as evenly spaced time points (daily, weekly, hourly, etc.)
- Distance measures are used to find similarity
- Structural analysis is done

Prediction

Predicting future data states based on current or historical data.

Example: What comes at the place of ?

2, 4, 6, 8, 10, ?

2, 3, 5, 7, ?, 13

2, 9, 16, 25, ?, 49

2, 6, 7, 33, 65, ?

(10 july, rain), (11 july, rain), (12 july, no-rain), (13 july, ?)

- Predication can sometimes be seen as classification
- Application includes weather, flood, pattern recognition.

Clustering

Clustering is similar to classification except the groups are not pre-defined.

Example: How many kind of files are there in my directory?

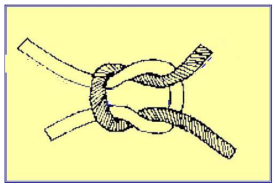
- Unsupervised learning setting
- We can use file name
- Words it has

Example: Who would take my offer?

- The database has information about age, gender, income, location, .. etc.

Summarization

Summarization maps data into subsets with associated simple descriptions (important details of main idea). It is also called characterization or generalization.



Example: How to compare two universities?

- Average JEE rank
- Average number of publication
- Student/Faculty ratio
- Combination

Association Rules

Association rules tries to do linked analysis.

Example: Whether sames products are selling together?

- $I = \{i_1, i_2, i_3, ...i_m\}$, $T = \{t_1, t_2, t_3, ..., t_n\}$ and $t_i \subseteq I$
- Minimum support count should be maintained
- Can you see: Subset of frequent items is also frequent
- Let $t_1 = (1, 3, 4)$, $t_2 = (2, 3, 5)$, $t_3 = (1, 2, 3, 5)$, $t_4 = (2, 5)$, $t_5 = (1, 4, 5)$
- What about $(2, 5)$?
- Apriori analysis can be applied

Sequence Discovery

Sequence Discovery is used to discover sequential patterns is the data.

Example: what is my website access pattern?

- Pattern is based on a time sequence of a action
- It is pattern discovery problem

Types of Learning

- **Supervised:** "right answers" are provided for sufficient training examples. Computer tells "right answers" for new input. Performance measure. (Classification and regression)
- **Unsupervised:** "right answers" are NOT provided and the computer tries to make sense of the data. How good the spread of items is. (clustering and association rule)
- **Semi-supervised:** "right answers" are provided for few training examples only
- **Active:** computer can ask questions. Needs less training. Opposite is passive learning
- **Lazy:** learner do not consolidate the findings.
- **Reinforced:** hit and trial method to minimize cost. (game playing)
- **Transfer:** Learning a task B to do A. (cycle riding for bike riding)
- **Deep:** processing like human brain

Challenges

- How good is the model
- How do I choose a model
- Do I have enough data
- Is data of sufficient quality - error in data, noise in data, missing value
- How confidence the result is
- Am I describing data correctly - whether features are correct

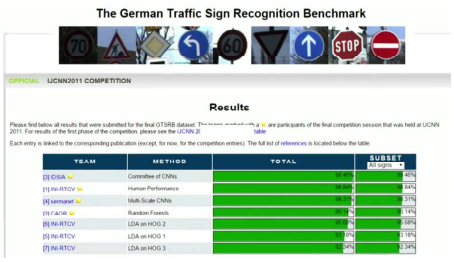
Applications of ML

In many domains including finance, robotics, bioinformatics, vision, natural language, *etc*.

- Spam filtering
- Speech/handwriting recognition
- Object detection/recognition
- Weather prediction
- Stock market analysis
- Search engines (e.g, Google)
- Ad placement on websites
- Adaptive website design
- Credit-card fraud detection
- Webpage clustering (e.g.,Google News)
- Machine Translation (e.g., Google
- Translate)
- Recommendation systems (e.g., Netflix, Amazon)
- Classifying DNA sequences
- Automatic vehicle navigation
- Performance tuning of computer systems
- Predicting good compilation flags for programs
- .. and many more

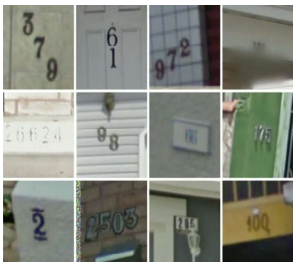
Success Stories

- Waymo: A safer driver that is always alert and never distracted
- First driverless ride on public roads in 2015 giving a ride to a sole blind
- In public: 2020

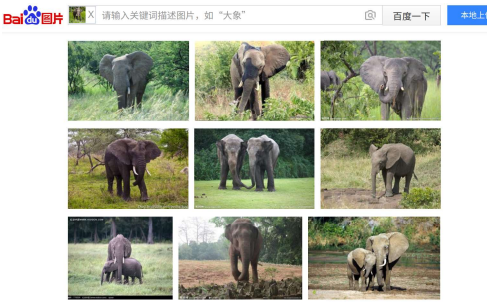


- German Traffic Sign Recognition Benchmark (GTSRB)
- 99.46% against 98.84% of human

Success Stories



- Google mapped every single location in France in two hour
- Images acquired from Google street view

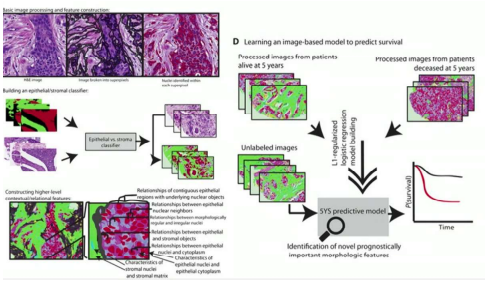


- Example of an image search
- That can taking care of color and pose of the object in the image

Success Stories

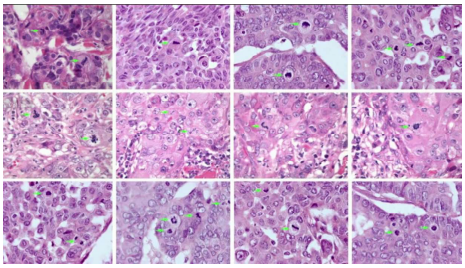
Computers can write

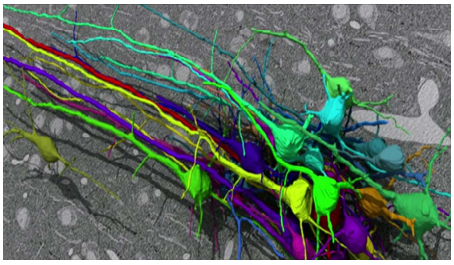
- Man in black shirt is playing guitar
- Two young girls are playing with Lego toy
- Black and white dog jumps over bar



- Tissues in magnification
- Stanford developed a ML algorithm that is better than human pathologist
- In predicting survival rate of cancer suffering

Success Stories

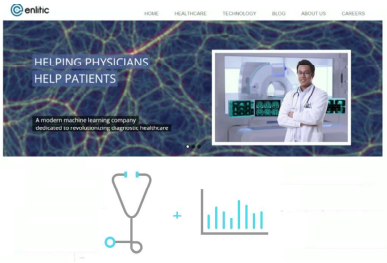


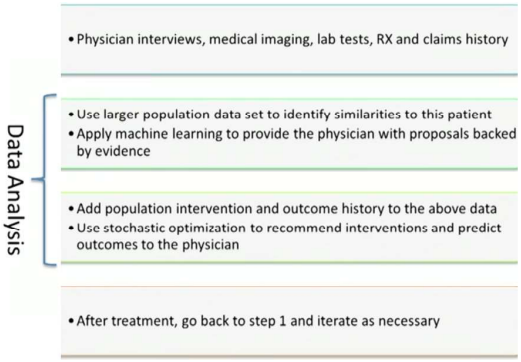


- System were developed by ML experts alone
- Without having any background in chemistry, biology or life sciences
- To identify cancerous tissues under microscope
- To perform neuron segmentation

Success Stories

It is possible to suggest very useful medicines by using just the data analytics techniques



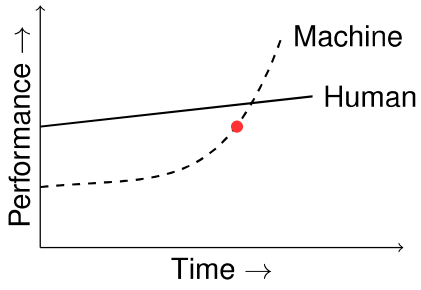


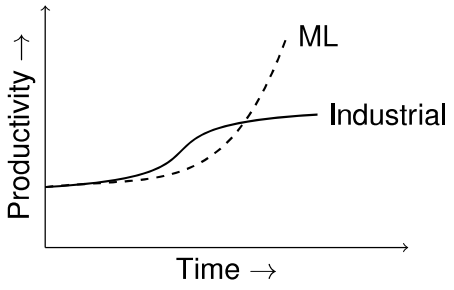
- `Enlitic` uses deep learning to make doctors faster and more accurate
- One have to use the middle path

Success Stories

By using following **four capabilities**, humans can do most of the work ($\sim$ 80%) like driving cars, preparing food, diagnosing diseases, Finding legal precedents, .. *etc*.

1. Reading and writing
2. Speaking and listening
3. Looking at things
4. Integrating knowledge

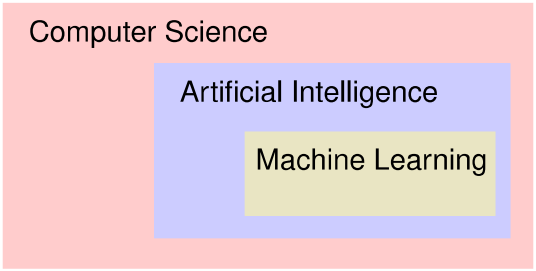




Perspective: Artificial Intelligence

Primary Question: How to make computers do things which at the moment, people do better [3]

AI attempts to build such intelligent entities



ML is a tool to enable AI

[3] There could be some tasks even humans are not good at.

Goal and Evaluation

Goal: We expect, at the end, you should be able to appreciate how various ML algorithms work, implement them on your own. Identify problems where ML can be applied.

Evaluation Scheme

1.	Quiz-01	5% Marks
2.	Quiz-02	5% Marks
3.	Assignments	10% Marks
4.	Mid-Semester Test (2H Close Book)	30% Marks
5.	Comprehensive Exam (3H Open Book)	50% Marks
	Total:	100% Marks

Thank You!

Thank you very much for your attention!

Queries ?

IS-ZC464: MACHINE LEARNING

Lecture-02: Basic ML

Dr. Kamlesh Tiwari
Assistant Professor
Department of Computer Science and Information Systems,
BITS Pilani, Pilani, Jhunjhunu-333031, Rajasthan, INDIA

July 29, 2018 (WILP @ BITS-Pilani Jul-Nov 2018)

Introduction

ML depends upon **Pattern Recognition** which corresponds to finding regularities in the data.

- There should be a pattern.
- No issues if we are unable to mathematically describe it.
- Sufficient examples or data is required.

Consider e-mail filtering SPAM/Not-SPAM

Assumption is that there are some words whose frequency is correlated to this filtering.

Netflix Prize (2009)

Open competition to predict user ratings for films. Prize of USD 1 million was given to the BellKor's Pragmatic Chaos team which improved previous prediction by $\sim$10.06% (uses matrix factorization)

Building Blocks

- Input: x
- Output: y
- Training data: $(x^{(1)}, y^{(1)})$, $(x^{(2)}, y^{(2)})$, ..., $(x^{(m)}, y^{(m)})$
- $x^{(i)}$ could be a multivariate say $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)})$
- Target function: true function

$$f : x \rightarrow y$$

- Hypothesis

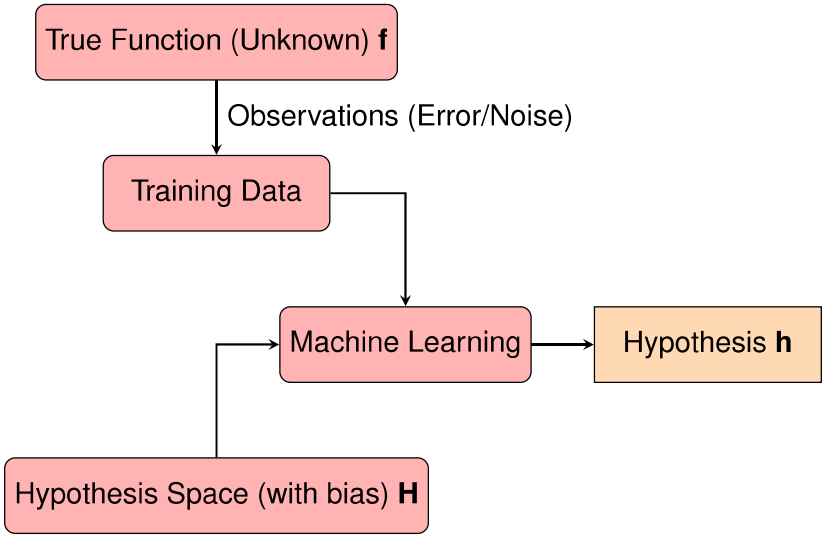
$$h : x \rightarrow y$$

- Accuracy: agreement b/w f and h

Issue is

True function is not known.

The Flow of ML



A Toy model

- **The Problem:** credit approval.
- Input: $x = (x_1, x_2, ..., x_n)$
- Let x_1=accountBal, x_2=Salary, x_3=age ...
- What weight we should give w_1=0.6, x_2=0.3, x_3=-0.1 ...
- The Model

$$\sum_{i=1}^{n} w_i \times x_i = \begin{cases} > Threshold & \text{Then APPROVE} \\ otherwise & \text{DENY/REJECT} \end{cases}$$

- Simplified:

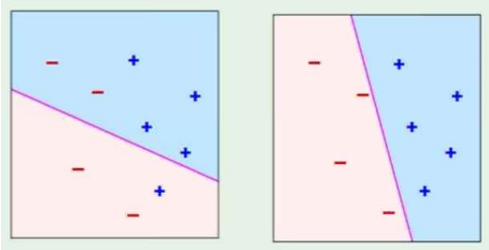
$$h(x) = sign(\sum_{i=1}^{n} w_i \times x_i - Threshold)$$

- Add an extra term x_0 then

$$h(x) = sign(\sum_{i=0}^{n} w_i \times x_i)$$

A Toy model (Contd..)

- Can you recognize $h(x) = sign(\sum_{i=0}^{n} w_i \times x_i)$
- It is a linear equation (in two dimension) or hyper plane



- Vector $(w_1, w_2, ..., w_m)$ would be normal on the plane. (why? because dot product is $\cos\theta$)
- What could change this plane? w_i's
- Learning: Use misclassified examples to update $w_i = w_i + y_i x_i$

Loss function

- Performance is the closeness of hypothesis function with target function
- For example
  - Classification

$$loss(y, h(x)) = \begin{cases} 1 & \text{if } h(x) \neq y \\ 0 & \text{otherwise} \end{cases}$$

  - Regression

$$loss(y, h(x)) = \begin{cases} (h(x) - y)^2 & \text{if } h(x) \neq y \\ 0 & \text{otherwise} \end{cases}$$

Performance

Error rates include the chance of accepting an intruder (False Acceptance Rate (FAR)) and that of rejecting a genuine individual (False Rejection Rate (FRR))[1]. FRR decreases when FAR increases

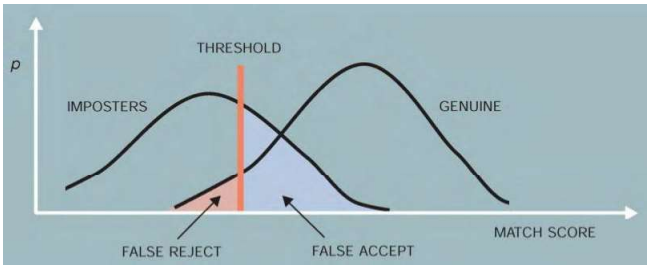
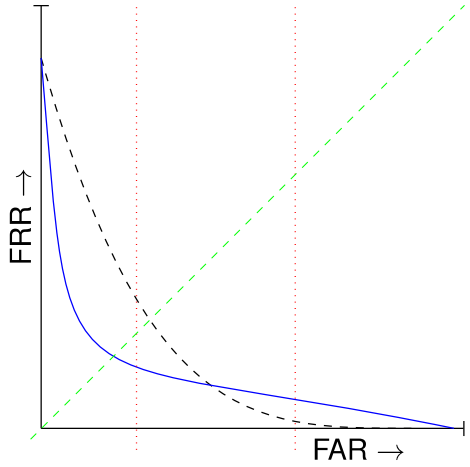


Figure: Probabilities of Genuine/ Imposter score (similarity measure)

Equal error rate (EER) corresponds to a point where FAR and FRR are equal. Threshold is used to take decision on accept/reject.

[1] Matching scores are of two kind similarity and dissimilarity.

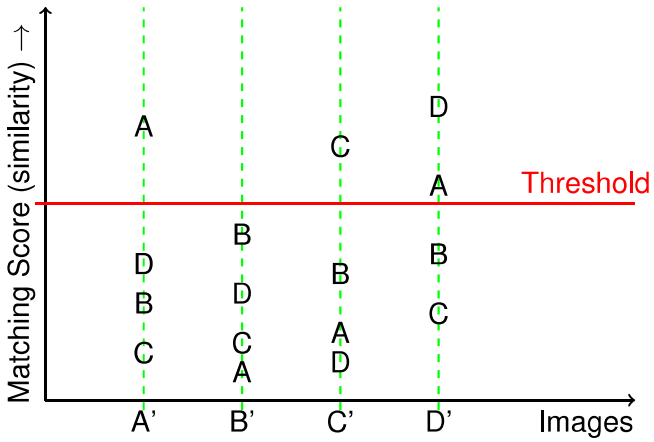
Receiver Operating Curve (ROC)



- Operating point is specified in terms of Threshold

Area under ROC represents error.

Error Happens

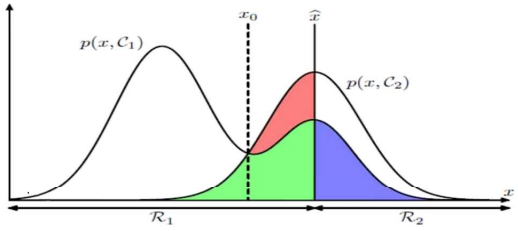


CRR is 100% but EER is ∼12%.

Minimize Misclassification

Goal is to minimize misclassification rate (risk)

$$p(mistake) = p(x \in R_1, C_2) + p(x \in R_2, C_1)$$

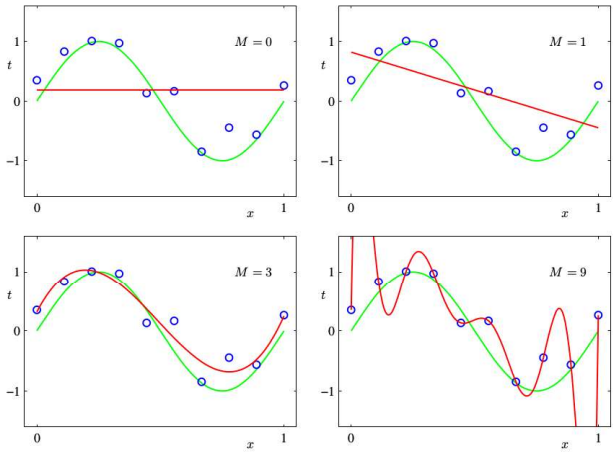


$$p(mistake) = \int_{R_1} p(x, C_2)\, dx + \int_{R_2} p(x, C_1)\, dx$$

Polynomial Curve Fitting (Towards Overfitting)

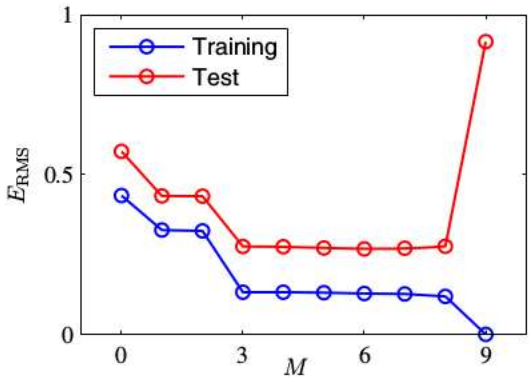
Is this a better curve?

$$h(x) = w_0 + w_1 x_1 + w_2 x_2^2 + w_3 x_3^3 + ...$$



Training and Test Error

Data is split in 1) Training 2) Testing 3) Validation [2]



Training error decreases with more complex model.
What happened at the end? (overfitting?)

[2] Generally 70, 20 and 10 %

Thank You!

Thank you very much for your attention!

Queries ?

IS-ZC464: MACHINE LEARNING

Lecture-03: Bayesian Learning (MAP and ML)

Dr. Kamlesh Tiwari
Assistant Professor
Department of Computer Science and Information Systems,
BITS Pilani, Pilani, Jhunjhunu-333031, Rajasthan, INDIA

August 04, 2018 (WILP @ BITS-Pilani Jul-Nov 2018)

Let's Predict

Consider following table. Can you determine the value at the place of ???

x1	x2	x3	x4	x5	Y
32	45	13	39	92	0
82	70	77	35	93	1
14	50	95	98	93	0
37	23	92	39	82	0
22	18	96	47	36	1
13	70	0	31	45	1
18	87	56	49	35	0
34	2	7	41	76	1
82	4	98	20	87	0
50	14	94	22	32	0
10	39	74	69	58	0
75	53	80	6	64	1
61	30	47	37	59	1
43	67	55	7	59	0
32	87	16	8	92	1
93	63	38	1	60	0
64	22	41	15	75	1
41	51	16	11	8	???

To do this

- You first need to determine how y depends upon $x1$, $x2$, $x3$, $x4$, $x5$

  $y = f(x1, x2, x3, x4, x5)$
- What about

  $f = mod(\sum w_i \times x_i, 2)$

  with $(w1, w2, w3, w4, w5) = (0,1,0,1,0)$

Recap: Building Blocks

- Input: x
- Output: y
- Training data: $(x^{(1)}, y^{(1)})$, $(x^{(2)}, y^{(2)})$, ..., $(x^{(m)}, y^{(m)})$
- $x^{(i)}$ could be a multivariate say $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)})$
- Target function: true function

$$f : x \rightarrow y$$

- Hypothesis

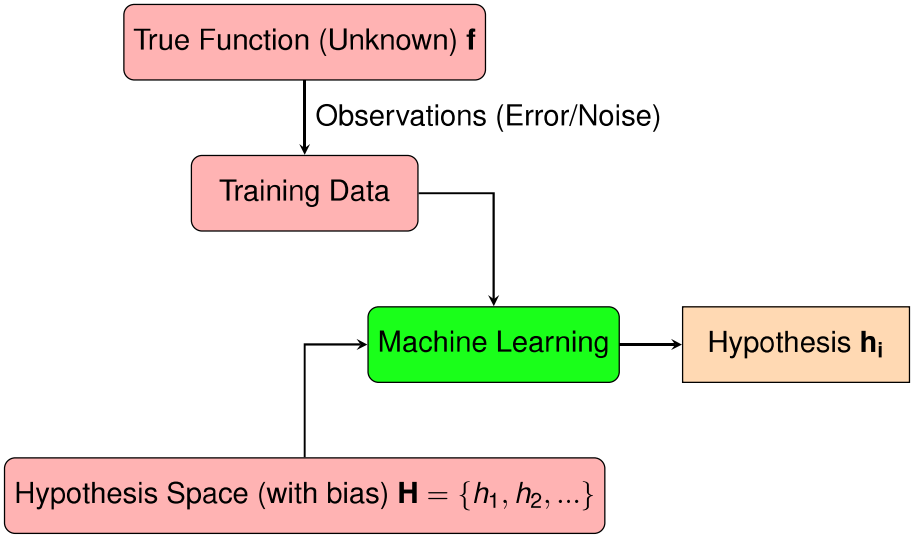
$$h : x \rightarrow y$$

- Accuracy: agreement b/w f and h

Issue is

True function is not known.

Recap: The Flow of ML



Recap: A Toy model

- **The Problem:** credit approval.
- Input: $x = (x_1, x_2, ..., x_n)$
- Let x_1=accountBal, x_2=Salary, x_3=age ...
- What weight we should give w_1=0.6, x_2=0.3, x_3=-0.1 ...
- The Model

$$\sum_{i=1}^{n} w_i \times x_i = \begin{cases} > \mathit{Threshold} & \text{Then APPROVE} \\ \mathit{otherwise} & \text{DENY/REJECT} \end{cases}$$

- Simplified: $h(x) = \mathit{sign}(\sum_{i=1}^{n} w_i \times x_i - \mathit{Threshold})$
- Add an extra term x_0 then

$$h(x) = \mathit{sign}(\sum_{i=0}^{n} w_i \times x_i)$$

Learning uses misclassified examples to update $w_i = w_i + \alpha y_i x_i$

Let's change our focus a bit

- Input: x Output: y
- Training data: $(x^{(1)}, y^{(1)})$, $(x^{(2)}, y^{(2)})$, ..., $(x^{(m)}, y^{(m)})$
- Target function $f : x \rightarrow y$
- Hypothesis $h : x \rightarrow y$

Instead of reporting y let's report $P(y)$; probability of being y

- For a given input x, output is not True/False
- But,

$$P(\mathit{True}) = 0.3$$

$$P(\mathit{False}) = 0.7$$

Bias: How to come up with these values

- Observe the data
- Assume data for 10 coin tosses be HHTHHTTHHH
- Now again the coin is flipped, what is expected output?
- $P(H) = 0.7$ and $P(T) = 0.3$

You can incorporate your bias

Assume you know the coin was unbiased (and you have a high confidence on this)

- You hypothetically consider 100 more flips. Being a fair coin it would give 50 H and 50 T
- So your estimate for probability of head is as below $P(H) = (7 + 50)/(110) = 0.518$ and $P(T) = 0.482$
- Why 100? why not 10000? OK; if you have more confidence on bias take 10000.

Bayes Theorem

Example: Three companies A, B and C makes 35%, 35% and 30% of all the lamps in market. Probability of their lamp being defective is 1.5%, 1% and 2% respectively. What is the probability that a randomly selected defective lamp was manufactured in factory C?

Bayes Theorem

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

Posterior, Prior, Likelihood and Evidence ($P(A|B)$, $P(A)$, $P(B|A)$, $P(B)$)

- What we want is $P(C|D)$ it is $\frac{P(D|C)P(C)}{P(D)}$
- $P(D) = P(A)P(D|A) + P(B)P(D|B) + P(C)P(D|C) = .35 * 0.015 + .35 * 0.01 + .3 * 0.02 = 0.01475$

$$P(C|D) = \frac{P(D|C)P(C)}{P(D)} = \frac{0.02 * 0.3}{0.01475} = 0.407$$

Hypothesis

X	Y
10	0
11	0
12	0
13	1
14	0
15	1
16	0
17	1
18	1

h_1	h_2	...
0	1	...
0	0	...
0	1	...
1	0	...
1	1	...
1	0	...
1	1	...
1	0	...
1	1	...

- In this example h_1, h_2, ... are hypothesis.
- **Hypothesis** is a function that aims to provide value of the Y
- Can you identify h_1 and h_2
- Represent H as candidate set of hypothesis, $i.e. h_i \in H$
- Size of H is at least 2^m

Bayesian Learning

It is based on assumption that quantities of interest are governed by probability distribution

- Notation
  - $P(h)$: initial probability that hypothesis h holds
  - $P(D)$: probability that data D will be observed
  - $P(D|h)$: probability of observing data D given some world in which hypothesis h holds
  - $P(h|D)$: probability of holding hypothesis h when data D is observed

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

Maximum a posteriori (MAP)

- Choose a hypothesis that maximizes $P(h|D)$

$$\begin{aligned} h_{MAP} &= \underset{h \in H}{\operatorname{argmax}} P(h|D) \\ &= \underset{h \in H}{\operatorname{argmax}} \frac{P(D|h)P(h)}{P(D)} \\ &= \underset{h \in H}{\operatorname{argmax}} P(D|h)P(h) \end{aligned} \tag{1}$$

- Because $P(D)$ is independent of h
- If all the hypothesis are equally probable, we may further simplify called *maximum likelihood (ML)*

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} P(D|h) \tag{2}$$

An Example

Let an illness affects 0.8% of population. There is a test which is 98% accurate for positive and 97% for negative. Consider following two hypothesis

- h_1: person is suffering some illness
- h_2: person is not suffering illness

A **randomly picked** person is tested for illness and is **found positive.** Which is MAP hypothesis out of h_1 and h_2.

- $P(D|h_1)P(h_1) = 0.98 \times 0.008 = 0.0078$ (normalized 0.21)
- $P(D|h_2)P(h_2) = 0.03 \times 0.992 = 0.0298$ (normalized 0.79)

Hypothesis h_2, that is the person is not suffering with illness is most probable.

Let $n = 100000 = (99200 + 800) = (\{96224+2976\} + \{16+784\})$
$P(h1) =\sim 0.21$ $P(h2) =\sim 0.79$

For our current example

X	Y
10	0
11	0
12	0
13	1
14	0
15	1
16	0
17	1
18	1

h_1	h_2	...
0	1	...
0	0	...
0	1	...
1	1	...
1	1	...
1	0	...
1	1	...
1	0	...
1	1	...

- Let bias for h_1 and h_2 be 2/50 and 6/50
- Since h_1 and h_2 are correct with probability 7/9 and 3/9 respectively
- Posterior is (7/9)*(2/50) and (3/9)*(6/50)
- Normalized probabilities are 0.4375 and 0.5625 respectively
- So MAP hypothesis corresponds to h_2
- Can you guess ML hypothesis? it is h_1

- **Brute-force MAP learning algorithm:** Evaluates posterior probability for all and returns the one with maximum
- **Consistent Learner:** learning algorithm is consistent learner if it provides a hypothesis that commits zero error

Next Class

Sum of Squared Differences (SSD) gives Maximum likelihood hypothesis

Some optimization methods

IS-ZC464: MACHINE LEARNING

Lecture-04: Bayesian Learning (ML is related to SSD), Lagrange

Dr. Kamlesh Tiwari
Assistant Professor
Department of Computer Science and Information Systems,
BITS Pilani, Pilani, Jhunjhunu-333031, Rajasthan, INDIA

August 11, 2018 (WILP @ BITS-Pilani Jul-Nov 2018)

Recap: Bayesian Learning

Bayes Theorem

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

- **Bayesian Learning** assumes that quantities of interest are governed by probability distribution
- **Maximum a posteriori (MAP)**

$$h_{MAP} = \underset{h \in H}{\text{argmax}}\, P(D|h)P(h)$$

$$h_{ML} = \underset{h \in H}{\text{argmax}}\, P(D|h)$$

Probability of getting a dataset

Assume you are flipping a biased coin where $p(H) = 0.4$. What is the probability that you see this dataset $D =< H, H, T, T, H, H >$

- $p(H) = 0.4$
- $p(T) = 1 - p(H) = 1 - 0.4 = 0.6$
- If **all the trails are independent** then $p(D|\theta)$

$$= p(H) \times p(H) \times p(T) \times p(T) \times p(H) \times p(H)$$

$$= 0.4^4 \times 0.6^2 = 0.009216$$

Note: Order do not matter in the trail. So $p(< H, H, H, H, T, T >)$ is same (in fact any other permutation)

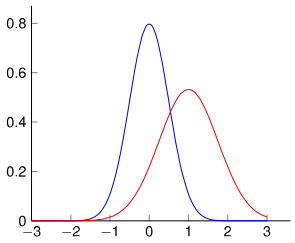
What is θ

It is the parameter. For our case it represents $p(H) = 0.4$

Normal Distribution

Many natural phenomenas are assumed to have Normal Distribution

1. Marks obtained by students in a test
2. Wight of a people in a population
3. Sum on dice, tossed 10 times
4. Number of heads in 1000 toss

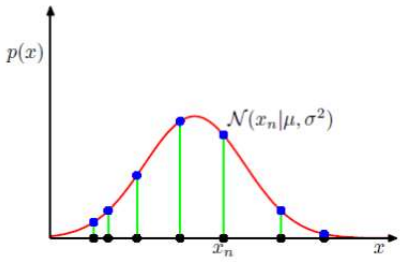


$$N(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Normal Distribution

Normal Distribution

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



- $E[x] = \int_{-\infty}^{\infty} N(x|\mu,\sigma^2) \times x \; dx = \mu$
- $E[x^2] = \int_{-\infty}^{\infty} N(x|\mu,\sigma^2) \times x^2 \; dx = \mu^2 + \sigma^2$
- $var[x] = E[x^2] - E[x]^2 = \sigma^2$
- Let the data set D has m data points that are is i.i.d.[1] drawn from normal distribution where true estimates for mean and variance are μ and σ^2
- Likelihood of black points is given by red curve

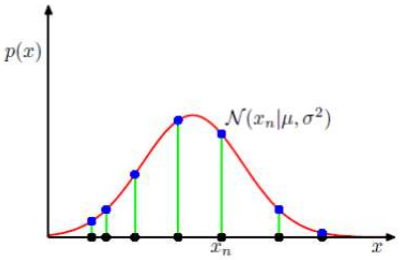
[1] Independent and identically distributed (68, 95, 99.7)

ML Estimate of Normal Distribution

Assume m data points that are i.i.d. (Independent and identically distributed) in given training data set D

- Probability of the data set D is

  $p(D|\mu,\sigma^2) = \Pi_{i=1}^{m} N(x_i \in D|\mu,\sigma^2)$
- Changing μ and σ gives different hypothesis h
- $h_{ML} = \text{argmax}_{h \in H} P(D|h)$



Our interest is to determine value of μ and σ that maximizes $p(D|\mu,\sigma^2)$

Maximum Likelihood Estimator of Normal Distribution

MLE would maximize the probability $p(D|\mu, \sigma^2)$ using appropriate μ and σ^2

- What if we optimize log of this probability?(it would be same)

$$\begin{aligned} \log p(D|\mu, \sigma^2) &= \sum_{i=1}^{m} \log N(x_i \in D|\mu, \sigma^2) \\ &= \frac{-1}{2\sigma^2} \sum_{i=1}^{m} (x_i - \mu)^2 - \frac{1}{2} m. \log(2\pi\sigma^2) \end{aligned}$$

- What parameters would optimize this function? differentiate and set to zero (with respect to μ and σ^2). For μ it is

$$\frac{d}{d\mu} \log p(D|\mu, \sigma^2) = \frac{-1}{2\sigma^2} \sum_{i=1}^{m} 2(x_i - \mu)(-1) - 0$$

contd ...

$$\frac{-1}{2\sigma^2} \sum_{i=1}^{m} 2(x_i - \mu)(-1) = 0$$

$$\sum_{i=1}^{m} (x_i - \mu) = 0$$

$$\mu = \frac{1}{m} \sum_{i=1}^{m} x_i$$

- Similarly for σ^2

$$\frac{d}{d\sigma^2} \log p(D|\mu, \sigma^2) = \frac{1}{2\sigma^4} \sum_{i=1}^{m} (x_i - \mu)^2 + 0 - \frac{m}{2\sigma^2}$$

contd ...

Equating to zero

$$\frac{1}{2\sigma^4}\sum_{i=1}^{m}(x_i-\mu)^2-\frac{m}{2\sigma^2}=0$$

Gives

$$\sigma^2=\frac{1}{m}\sum_{i=1}^{m}(x_i-\mu)^2$$

For given data set $D=\{x_1,x_2,...,x_m\}$ that are drawn i.i.d from normal distribution; one can find ML estimate μ and σ^2 as

$$\mu_{ML}=\frac{1}{m}\sum_{i=1}^{m}x_i$$

$$\sigma^2_{ML}=\frac{1}{m}\sum_{i=1}^{m}(x_i-\mu_{ML})^2$$

Note that

Expected value of μ_{ML} is true value

$$\begin{aligned}E[\mu_{ML}] &= E[\frac{1}{m}\sum_{i=1}^{m}x_i]=\frac{1}{m}\sum_{i=1}^{m}E[x_i]\\ &= \frac{1}{m}\times m\times E[x_i]=E[x_i]=\mu\end{aligned}$$

Where as expected value of σ^2_{ML} is as follows

$$E[\sigma^2_{ML}]=\frac{m-1}{m}\sigma^2$$

MLE is biased in case of variance but not for mean

How?

For notational convenience let us use $\bar{x} = \mu_{ML}$

$$\begin{aligned} E[\sigma^2_{ML}] &= E[\frac{1}{m}\sum_{i=1}^{m}(x_i - \bar{x})^2] = E[\frac{1}{m}\sum_{i=1}^{m}(x_i^2 - 2x_i\bar{x} + \bar{x}^2)] \\ &= \frac{1}{m}E[\sum_{i=1}^{m}x_i^2 - 2\bar{x}\sum_{i=1}^{m}x_i + \sum_{i=1}^{m}\bar{x}^2] = \frac{1}{m}E[\sum_{i=1}^{m}x_i^2 - 2\bar{x}(m.\bar{x}) + m.\bar{x}^2] \\ &= \frac{1}{m}E[\sum_{i=1}^{m}x_i^2 - m.\bar{x}^2] = \frac{1}{m}[m.E[x^2] - m.E[\bar{x}^2]] = E[x^2] - E[\bar{x}^2] \end{aligned}$$

Since $\sigma_x^2 = E[x^2] - E[x]^2 \qquad \sigma_{\bar{x}}^2 = E[\bar{x}^2] - E[\bar{x}]^2 \qquad E[x] = E[\bar{x}] = \mu$

$$E[\sigma^2_{ML}] = (\sigma_x^2 + E[x]^2) - (\sigma_{\bar{x}}^2 + E[\bar{x}]^2) = \sigma_x^2 - \sigma_{\bar{x}}^2$$

Since $\sigma_{\bar{x}}^2 = var[\bar{x}] = var[\frac{1}{m}\sum_{i=1}^{m}x_i] = \frac{1}{m^2}var[\sum_{i=1}^{m}x_i] = \frac{1}{m^2}\sum_{i=1}^{m}var[x_i] = \frac{1}{m^2}m.var[x_i] = \frac{1}{m}var[x_i] = \frac{1}{m}\sigma_x^2$

$$E[\sigma^2_{ML}] = \sigma_x^2 - \frac{1}{m}\sigma_x^2 = \frac{m-1}{m}\sigma_x^2$$

To overcome this, we could multiply sample variance with $m/(m-1)$

Maximum Likelihood and Least Squared Error (LSE)

Under certain assumptions, a learning algorithm minimizing Least Squared Error, will output Maximum Likelihood hypothesis h_{ML}

- Assume *noise* is random and obeys normal distribution with zero mean and variance σ^2 which is independent for every data point

$$h_{ML} = \underset{h \in H}{\text{argmax}}\, P(D|h)$$

- $D =< (x_1, y_1), (x_2, y_2), ..., (x_m, y_m) >$ with $y_i = f(x_i) + e_i$. Assuming training data being mutually independent given h

$$h_{ML} = \underset{h \in H}{\text{argmax}}\, \Pi_{i=1}^{m} p(y_i|h)$$

- Data has variance σ^2 and mean around true target value $\mu = f(x_i)$. Therefore, $p(y_i|h)$ being a normal distribution would have a variance σ^2 and mean μ

Maximum Likelihood and LSE (contd...)

$$h_{ML} = \underset{h \in H}{\text{argmax}} \Pi_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(y_i - \mu)^2}$$

As we assume h to be correct description of f so $\mu = f(x_i) = h(x_i)$

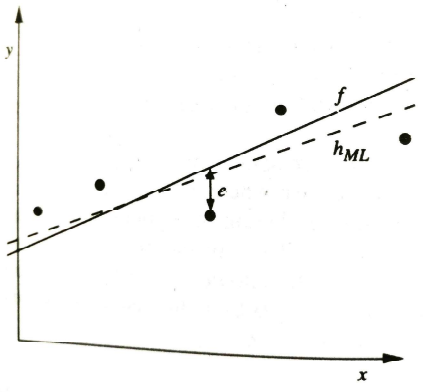
$$h_{ML} = \underset{h \in H}{\text{argmax}} \Pi_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(y_i - h(x_i))^2}$$

Being continuous function, we may choose its log to optimize

$$\begin{aligned} h_{ML} &= \underset{h \in H}{\text{argmax}} \sum_{i=1}^{m} \ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}(y_i - h(x_i))^2 \\ &= \underset{h \in H}{\text{argmax}} \sum_{i=1}^{m} -\frac{1}{2\sigma^2}(y_i - h(x_i))^2 = \underset{h \in H}{\text{argmin}} \sum_{i=1}^{m} (y_i - h(x_i))^2 \end{aligned}$$

As the first term being constant.

SSD gives Maximum Likelihood hypothesis



$$h_{ML} = \underset{h \in H}{\text{argmin}} \sum_{i=1}^{m} (y_i - h(x_i))^2$$

Multivariate Optimization

- **Lagrange Multiplier:** Used to find local optima (maxima or minima) of some objective function f, subject to equality constraint on g. *Example:* maximize $f(x, y)$ subject to $g(x, y) = c$
- Essentially we want a point where gradient of f and g are scaler multiple $\bigtriangledown f = \lambda \bigtriangledown g$ that is $\bigtriangledown f - \lambda \bigtriangledown g = 0$

Example: Let cost of producing x amount of product A and y amount of product B is $6x^2 + 12y^2$ and we want at least 90 items to be produced. Find optimal value of x and y

Lagrange equation for the problem is $F = 6x^2 + 12y^2 - \lambda(x + y - 90)$

$$\frac{\partial}{\partial x}(F) = 12x - \lambda \qquad \frac{\partial}{\partial y}(F) = 24y - \lambda \qquad \frac{\partial}{\partial \lambda}(F) = x + y - 90$$

Solving above equations by equating to zero gives $x = 60$ and $y = 30$

Example2: Lagrange Multiplier

Example

Optimize $f(x, y, z) = x^2 + x + 2y^2 + 3z^2$ on constraint $x^2 + y^2 + z^2 = 1$

- $F = x^2 + x + 2y^2 + 3z^2 - \lambda(x^2 + y^2 + z^2 - 1)$
- Determine

$$\frac{\partial}{\partial x}(F) \qquad \frac{\partial}{\partial y}(F) \qquad \frac{\partial}{\partial z}(F) \qquad \frac{\partial}{\partial \lambda}(F)$$

- Case-1: $y = z = 0 \rightarrow x = \pm 1$
- Case-2: $y = 0, \lambda = 3 \rightarrow x = 1/4, z = \pm\frac{\sqrt{15}}{4}$
- Case-3: $\lambda = 2, z = 0 \rightarrow x = 1/2, y = \pm\frac{\sqrt{3}}{2}$
- Max is 25/8 and min is 0 at $(\frac{1}{4}, 0, \frac{\sqrt{15}}{4})$ and $(-1, 0, 0)$

Thank You!

Thank you very much for your attention!

Queries ?

IS-ZC464: MACHINE LEARNING

Lecture-05: EM, Singular Value Decomposition (SVD)

Dr. Kamlesh Tiwari
Assistant Professor
Department of Computer Science and Information Systems,
BITS Pilani, Pilani, Jhunjhunu-333031, Rajasthan, INDIA

August 18, 2018 (WILP @ BITS-Pilani Jul-Nov 2018)

Basics

- Vector: amplitude, addition, scaler multiplication, dot product and angle between two vectors

$$\cos\theta = \frac{\vec{a}.\vec{b}}{|\vec{a}|.|\vec{b}|}$$

- Matrix, transpose, multiplication and inverse

$$A^{-1} = \frac{adjoint(A)}{|A|}$$

- Adjoint is transpose of co-factor matrix of A
- Non singular matrix has $|A| \neq 0$

Expectation Maximization (EM)

Mixture of Gaussian

Data may come from multiple distributions. How to separate?

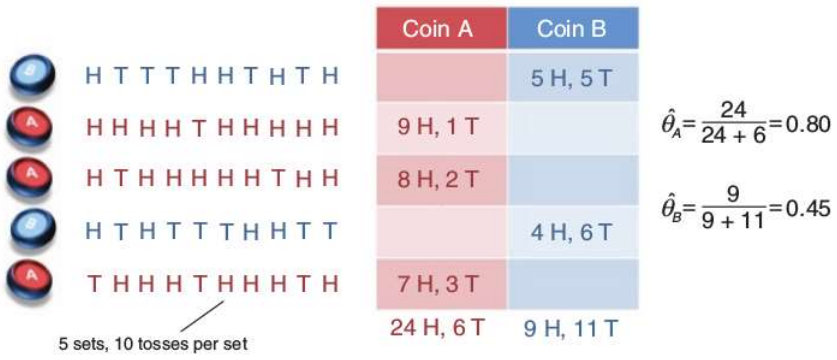
Example: Consider two coins A and B with biases (Probabilities of getting head) θ_A, θ_B.
Let we have two vectors $x = (x_1, x_2, .., x_m)$ and $z = (z_1, z_2, .., z_m)$ where $z_i \in \{A, B\}$ specifying number of heads in 10 flips and identity of coin. Then

$$\hat{\theta}_A = \frac{\text{Number of heads using A}}{\text{Total coin flips using A}}$$

$$\hat{\theta}_B = \frac{\text{Number of heads using B}}{\text{Total coin flips using B}}$$

Let $x = (5, 8, 8, 4, 7)$ and $z = (B, A, A, B, A)$

Expectation Maximization (EM)



What if z is not provided ? Refer z as latent (hidden) variable.

Expectation Maximization (EM)

1. Start with initial guess of parameters $\theta^{(t)} = (\hat{\theta}_A^{(t)}, \hat{\theta}_B^{(t)})$
2. Determine for each of the m sets whether coin A or B has generated this observation (estimating z)
3. Assume this coin assignment to be correct and apply maximum likely hood estimation to get $\theta^{(t+1)} = (\hat{\theta}_A^{(t+1)}, \hat{\theta}_B^{(t+1)})$
4. Repeat step 2-3 until converge

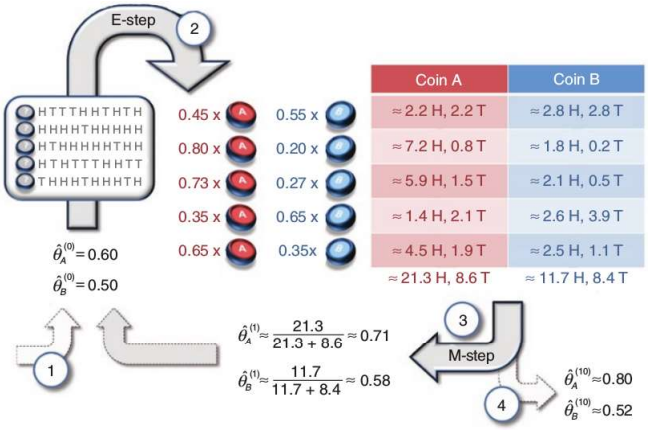
Recall (Example)

If you have a coin with $p(H) = 0.3$ what is probability of getting 4 heads in 12 trials?

$$^{12}C_4 \times p(H)^4 \times (1 - p(H))^8$$

Example

1. Compute likelihood it was from coin A or B. Using the binomial distribution with probability θ of head on n trials with k success $p(k) = {}^n C_k \theta^k (1-\theta)^{n-k}$. Likelihood if A and B for first trial is 0.00079 and 0.00097 (prob 0.45, 0.55)



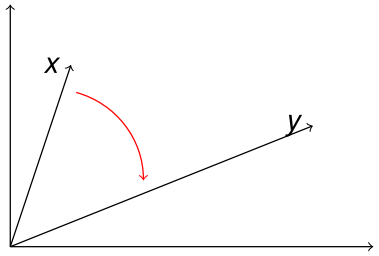
Vector and Matrix multiplication

Let

$$x = \begin{bmatrix} 1 \\ 3 \end{bmatrix} \qquad A = \begin{bmatrix} 2 & 1 \\ -1 & 1 \end{bmatrix}$$

Consider

$$y = Ax = \begin{bmatrix} 2 & 1 \\ -1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 3 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$$



Matrix multiplication has two effect

$$\textit{Rotation} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

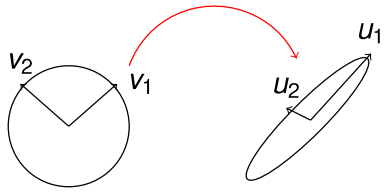
$$\textit{Scaling} \begin{bmatrix} \alpha & 0 \\ 0 & \alpha \end{bmatrix}$$

Eigen Vector of a Matrix

Special vectors

Can only be scaled on multiplication

What happens to a circle under Matrix multiplication



- Circle becomes ellipse
- If unit vectors along major and minor axis be u_1 and u_2 then orthogonal vectors v_1 and v_2 on the circle becomes $\sigma_1 u_1$ and $\sigma_2 u_2$
- Essentially it is a transformation from one **vector space** with v_1, v_2, ...v_n to new vector space u_1, u_2, ...u_n along with stretching factor σ_1, σ_2, ...σ_n. Such that $A \times v_j = \sigma_1 u_j \quad j \in \{1, ..., n\}$.

$$[A]\,[v_1, v_2, ..., v_n] = [u_1, u_2, ..., u_n] \begin{bmatrix} \sigma_1 & 0 & ... & 0 \\ 0 & \sigma_2 & ... & 0 \\ 0 & 0 & ... & 0 \\ 0 & 0 & ... & \sigma_n \end{bmatrix}$$

$$AV = U\Sigma \tag{1}$$

What happens to a circle under Matrix multiplication

$$AV = U\Sigma \tag{2}$$

When V being a *orthogonal matrix* (*i.e.* transpose is its inverse). Multiply V^T to both side of Equation.2
$AV \times V^T = U\Sigma V^T$

$$A = U\Sigma V^T$$

- One matrix is being represented as a product of three matrices
- V^T and U are orthogonal matrices that are related to rotation
- Σ is related to scaling or stretching

1. Every matrix $A_{m \times n}$ has SVD decomposition
2. Singular values $\{\sigma_i\}$ are positive and are uniquely determined. Also $\sigma_i \geq \sigma_j \geq 0 \quad \forall i \leq j$
3. $\{u_j\}$ and $\{v_j\}$ are also unique.

How to find SVD

$$A = U\Sigma V^T$$

$$\begin{aligned} A^TA &= (U\Sigma V^T)^T U\Sigma V^T \\ &= (V\Sigma^T U^T)U\Sigma V^T \\ &= V\Sigma^T(U^TU)\Sigma V^T \\ &= V\Sigma(I)\Sigma V^T \\ &= V\Sigma^2 V^T \\ (A^TA)V &= (V\Sigma^2 V^T)V \\ &= V\Sigma^2 \end{aligned}$$

$$\begin{aligned} AA^T &= U\Sigma V^T(U\Sigma V^T)^T \\ &= U\Sigma V^T(V\Sigma^T U^T) \\ &= U\Sigma(V^TV)\Sigma^T U^T \\ &= U\Sigma(I)\Sigma U^T \\ &= U\Sigma^2 U^T \\ (AA^T)U &= (U\Sigma^2 U^T)U \\ &= U\Sigma^2 \end{aligned}$$

1. Both the branches lead to a eigne value problem like $A \times x = \lambda \times x$
2. Therefore U and V are eigen vectors

Example

$$A = \begin{bmatrix} 5 & 5 \\ -1 & 7 \end{bmatrix}$$

$$A^TA = \begin{bmatrix} 5 & -1 \\ 5 & 7 \end{bmatrix} \times \begin{bmatrix} 5 & 5 \\ -1 & 7 \end{bmatrix}$$

$$A^TA = \begin{bmatrix} 26 & 18 \\ 18 & 74 \end{bmatrix}$$

$$det(A^TA - \lambda I) = \begin{vmatrix} 26-\lambda & 18 \\ 18 & 74-\lambda \end{vmatrix}$$

$$\lambda^2 - 100\lambda + 1600 = 0$$

$$(\lambda^2 - 20)(\lambda - 80) = 0$$

$$\lambda = 20, 80$$

Eigen vector for $\lambda = 80$

$$(A^TA - 80I) = 0$$

$$\begin{bmatrix} -54 & 18 \\ 18 & -6 \end{bmatrix} \times \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = 0$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{10}} \\ \frac{3}{\sqrt{10}} \end{bmatrix}$$

Example

Eigen vector for $\lambda = 20$

$$(A^T A - 20I) = 0$$

$$\begin{bmatrix} 6 & 18 \\ 18 & 54 \end{bmatrix} \times \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = 0$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \frac{-3}{\sqrt{10}} \\ \frac{1}{\sqrt{10}} \end{bmatrix}$$

Therefore,

$$V = \begin{bmatrix} \frac{1}{\sqrt{10}} & \frac{-3}{\sqrt{10}} \\ \frac{3}{\sqrt{10}} & \frac{1}{\sqrt{10}} \end{bmatrix}$$

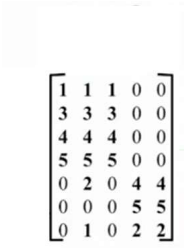
$$\Sigma = \begin{bmatrix} 4\sqrt{5} & 0 \\ 0 & 2\sqrt{5} \end{bmatrix}$$

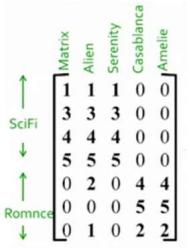
Using $AV = U\Sigma$

$$\begin{bmatrix} 5 & 5 \\ -1 & 7 \end{bmatrix} \times \begin{bmatrix} \frac{1}{\sqrt{10}} & \frac{-3}{\sqrt{10}} \\ \frac{3}{\sqrt{10}} & \frac{1}{\sqrt{10}} \end{bmatrix} = U \times \begin{bmatrix} 4\sqrt{5} & 0 \\ 0 & 2\sqrt{5} \end{bmatrix}$$

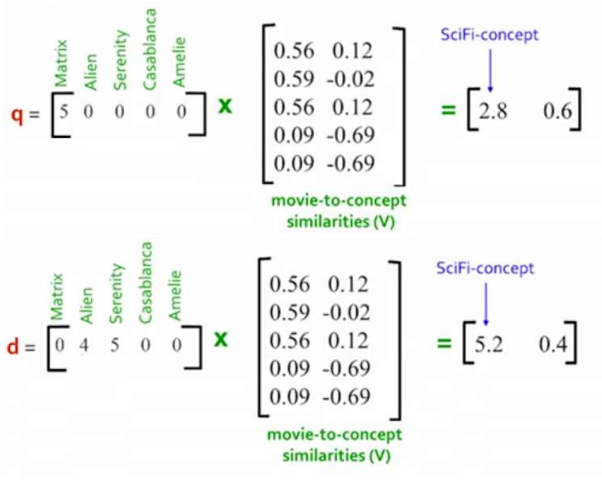
$$U = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

A Case Study





A Case Study



Conclusion

- SVD: every matrix can be decomposed in three components

$$\underset{m\times n}{[A]} = \underset{m\times r}{[U]} \times \underset{r\times r}{[\Sigma]} \times \underset{r\times n}{[V^T]}$$

- Time complexity is $O(m^2n)$ or $O(mn^2)$ as $AA^T = U\Sigma^2U^T$
- SVD provides best possible projection and is an optimal low rank approximation in terms of **Frobenius norm** $\sqrt{\sum(a_{ij} - b_{ij})^2}$
- For dimensionality reduction, retain 80-90% of energy ($\sum \sigma_i^2$)
- Interpretation is hard: a singular vector specifies a linear combination of all input columns or rows.
- Lack of sparsity: singular vectors are dense.

Next Class

Curse of Dimensionality, and PCA

MDL + HMM

Thank You!

Thank you very much for your attention!

Queries ?

IS-ZC464: MACHINE LEARNING

Lecture-06: Dimensionality, PCA and Eigenfaces, MDL + HMM

Dr. Kamlesh Tiwari
Assistant Professor
Department of Computer Science and Information Systems,
BITS Pilani, Pilani, Jhunjhunu-333031, Rajasthan, INDIA

August 25, 2018 (WILP @ BITS-Pilani Jul-Nov 2018)

Recap: Singular Value Decomposition (SVD)

SVD: every matrix can be decomposed in three components

$$A = U\Sigma V^T$$

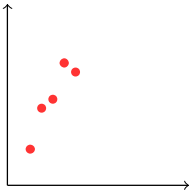
$$\underset{m\times n}{\begin{bmatrix} A \end{bmatrix}} = \underset{m\times r}{\begin{bmatrix} U \end{bmatrix}} \times \underset{r\times r}{\begin{bmatrix} \Sigma \end{bmatrix}} \times \underset{r\times n}{\begin{bmatrix} V^T \end{bmatrix}}$$

- Control the value of r for better understanding of the data.

Curse of Dimensionality

Observed vs True Dimensionality

- Consider following data and answer what is its dimensionality?
- Is it not 2?



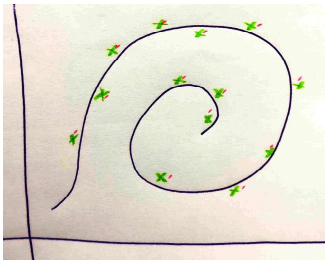
- May be not!
- Data may be observed across different sensors, so small variation maybe due to noise or ...
- It could also be possible that all the observations may be dependent on some quantity which is not being measured

Curse of Dimensionality

Databases are generally of high dimension

- Images contain lot of pixels and text may contain lot of words (250×250 pixels, or 10^6 words)

- True dimensionality may be lot lower than the observed one.
- Data may be in some low dimensional manifold (sheet) in high dimensional space



Curse of Dimensionality

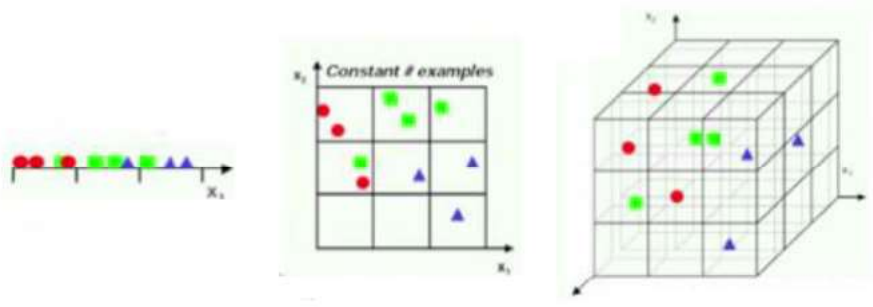
Consider handwritten digits

- Assume 20×20 bitmap (2^{400} observation)
- We would never see most of the events
- True dimensionality is something like possible pen strokes

Curse of Dimensionality

Why it is a problem

- Because most of the machine learning methods tends to **count** evidences
- Space grows very quickly but number example remains limited
- Density of data decreases sharply with the dimension. This lead to no observations for most of the cases



Mean, Variance and Covariance

- Conceptually **mean** represents the center and **variance** the spread of data points
- Let $a = [a_1 a_2 .. a_n]$ and $b = [b_1 b_2 .. b_n]$ be two set of data (assume their mean be zero). And we want to find out whether these two are statistically independent? **Covariance** comes into picture

$$\sigma_a^2 = \frac{1}{n-1} a \times a^T \quad \sigma_b^2 = \frac{1}{n-1} b \times b^T \quad \sigma_{ab}^2 = \frac{1}{n-1} a \times b^T$$

  Essentially an inner-product
- If a and b are of unit length then, the dot product $a \times b^T$ tells how much they are in same direction.
- If they are in same direction the value would be 1 and when they are orthogonal it would be 0.

Mean, Variance and Covariance

- Consider covariance among all pair of data vectors

$$C_X = \frac{1}{n-1} XX^T = \begin{bmatrix} \sigma_{a_1 a_1}^2 & \sigma_{a_1 a_2}^2 & \cdots & \sigma_{a_1 a_n}^2 \\ \sigma_{a_2 a_1}^2 & \sigma_{a_2 a_2}^2 & \cdots & \sigma_{a_2 a_n}^2 \\ \vdots & \vdots & \cdots & \vdots \\ \sigma_{a_n a_1}^2 & \sigma_{a_n a_2}^2 & \cdots & \sigma_{a_n a_n}^2 \end{bmatrix}$$

- Diagonal has variance and off diagonal covariance. It is a symmetric matrix
- If vectors are in same direction the covariance would be 1 and when they are orthogonal it would be 0.
- We want small covariance and large variance. *i.e.* large values at diagonal and small at rest of the places.
- If we could make it diagonal then there would be no redundancy

Diagonalization

- Let X be the data matrix, consider XX^T this is a symmetric matrix and self adjoint therefore one can always do **eigen value decomposition**.

$$XX^T = S\Lambda S^T$$

  where S is matrix of eigen vectors ($S^{-1} = S^T$) and Λ is diagonal matrix of eigen values
- Consider $Y = S^T X$, then

$$C_Y = \frac{1}{n-1} YY^T = \frac{1}{n-1} S^T X (S^T X)^T = \frac{1}{n-1} S^T X X^T S$$

$$C_Y = \frac{1}{n-1} S^T S \Lambda S^T S = \frac{1}{n-1} \Lambda$$

- Wow! covariance matrix of $S^T X$ is diagonal :)

PCA

- Similar operation can also be done by using SVD
- We know every matrix can be written as $X = U\Sigma V^T$
- Let $Y = U^T X$

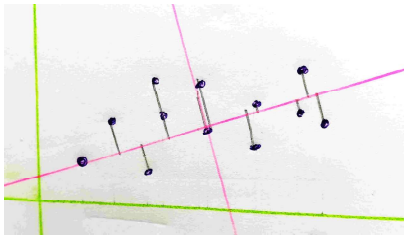
$$C_Y = \frac{1}{n-1} YY^T = \frac{1}{n-1} U^T X (U^T X)^T = \frac{1}{n-1} U^T X X^T U$$

$$C_Y = \frac{1}{n-1} U^T (U\Sigma^2 U^T) U = \frac{1}{n-1} \Sigma^2$$

- Note that $\Sigma^2 = \Lambda$, there is a connection between singular value and eigen value.

Data and dimensionality reduction

Effect of rotation and translation



- We can discover a better representation where there is more spread in the data therefore less chance of misclassification.

See some data [1]

Use *libraOffice* to create a .CSV file. First column is serial number and next three columns get random values from `FLOOR(RAND()*100)`.

1	47	21	22
2	32	80	90
3	35	39	53
4	8	63	73
5	34	10	79
6	1	14	53
7	75	58	39
8	45	81	53
9	71	42	65
10	15	97	70
11	66	84	36
:	:	:	:
:	:	:	:
19	36	27	8
20	7	36	49
21	38	96	93
22	87	71	2
23	52	96	35
24	15	49	34
25	26	98	36
26	81	17	17
27	67	41	70
28	38	91	34
29	11	36	39
30	71	85	91

```
import pandas as pd
from sklearn.decomposition import PCA

df = pd.read_csv(''./myData.csv'', names=['v1','v2','v3','v4'])

for i in range(1,5):
     pca = PCA(n_components=i)
     pca.fit(df)
     print sum(pca.explained_variance_ratio)
```

Result

```
0.389843872091
0.731978159735
0.96974472518
1.0
```

Four component take 100% of variation

See some more data [2]

```
import numpy.random as np
numPoints=15
np.seed(500)
v1 = [np.randint(low=1,high=80) for i in range(numPoints)]
v2 = [2*v1[i] for i in range(numPoints)]
v3 = [np.randint(low=1,high=80) for i in range(numPoints)]
v4 = np.permutation(v1)
v5 = [np.randint(low=0,high=2) for i in range(numPoints)]
aData = list(zip(v1,v2,v3,v4,v5))

print aData
```

(56, 112, 20, 18, 1), (66, 132, 14, 73, 0), (18, 36, 56, 56, 1),
(79, 158, 48, 62, 1), (62, 124, 57, 66, 0), (32, 64, 11, 32, 0),
(73, 146, 47, 79, 0), (72, 144, 14, 78, 1), (78, 156, 3, 18, 1),
(18, 36, 60, 19, 0), (18, 36, 61, 72, 0), (42, 84, 15, 18, 0),
(35, 70, 61, 35, 1), (43, 86, 76, 43, 0), (19, 38, 63, 42, 0)

```
df = pd.DataFrame(data = aData, columns=['v1', 'v2', 'v3', 'v4', 'v5'])
for i in range(1,6):
    pca = PCA(n_components=i)
    pca.fit(df)
    print sum(pca.explained_variance_ratio)
```

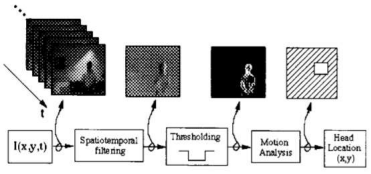
Result

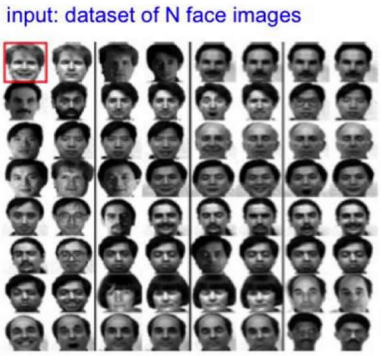
0.759164848209
0.932085855893
0.999943477329
1.0
1.0

Why only four components?
as v_2 is linearly dependent on v_1

PCA in action

Face recognition (Eigenfaces for recognition [1991] *Turk, Matthew*)

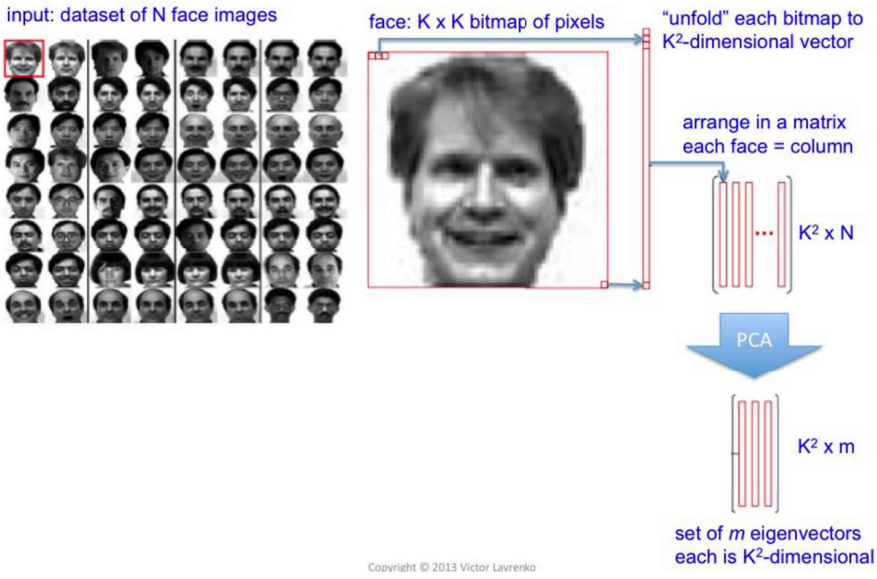




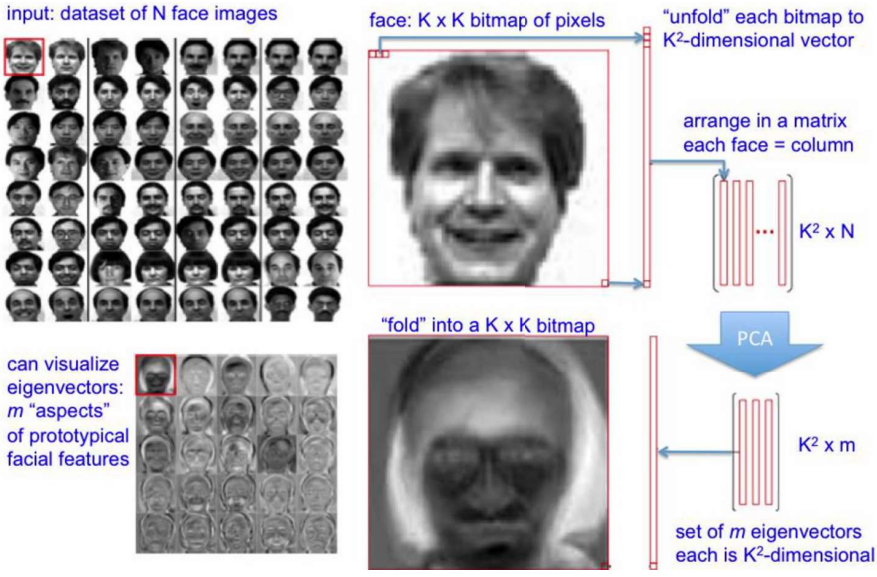
Database of 16 individuals (2500 images) achieved 96% correct classification [1]

[1] https://www.cs.ucsb.edu/ mturk/Papers/jcn.pdf

Eigenfaces



Eigenfaces



Minimum Description Length (MDL)

Kolmogorov complexity of data 1963

Kolmogorov complexity of a data is the length of the shortest computer program that produces it as output.

- Compare 1111111111111111 and 0000000000000000
- Compare 1111111111111111 and 1111111111101111
- Compare 1111111111111111 and 101100110

by Jorma Rissanen in 1978

Goal of statistical inference should be to find *regularity* in the data that can be identified by *ability to compress*

- Learning is viewed as data compression
- One should select a hypothesis that compresses the data most

Minimum Description Length (MDL)

Outlook x_1	Temperature x_2	Humidity x_3	Wind x_4	Play y
sunny	hot	high	weak	no
sunny	hot	high	strong	no
overcast	hot	high	weak	yes
rain	mild	high	weak	yes
rain	cool	normal	weak	yes
rain	cool	normal	strong	no
overcast	cool	normal	strong	yes
sunny	mild	high	weak	no
sunny	cool	normal	weak	yes
rain	mild	normal	weak	yes
sunny	mild	normal	strong	yes
overcast	mild	high	strong	yes
overcast	hot	normal	weak	yes
rain	mild	high	strong	no

- Each tuple can be represented as set of attribute pairs as E = { {x_1=summy, x_2=hot, x_3=high, x_4=weak}, {x_1=summy, x_2=hot, x_3=high, x_4=strong}, ... }
- Total number of different attribute values? are 10
- Each tuple have 4 attribute that could be chosen in at most ${}^{10}C_4$ ways =210
- Class attributes are two so total number of ways are 210×2

Minimum Description Length (MDL)

- Code length in bits $\log(210 \times 2) = 8.715$
- Code length of whole database $L(E) = 8.715 \times 14 = 122.01$

Consider two Hypothesis

H1: [play=yes]
If {*outlook*=overcast}
If {*humidity*=normal, *wind*=weak}

H2: [play=yes]
If {*outlook*=overcast}
If {*humidity*=normal, *wind*=weak}
If {*temperature*=mild, *humidity*=normal}

- Length of Hypothesis $L(H1) = \log({}^{10}C_1) + \log({}^{10}C_2) = 8.81$
- Length of Hypothesis
  $L(H2) = \log({}^{10}C_1) + \log({}^{10}C_2) + \log({}^{10}C_2) = 14.30$

Minimum Description Length (MDL)

Next we need to encode exceptions: (what is tp, tn, fp, fn)

Outlook x_1	Temperature x_2	Humidity x_3	Wind x_4	Play y	H1	H2
sunny	hot	high	weak	no		
sunny	hot	high	strong	no		
overcast	hot	high	weak	yes	y	y
rain	mild	high	weak	yes		
rain	cool	normal	weak	yes	y	y
rain	cool	normal	strong	no		
overcast	cool	normal	strong	yes	y	y
sunny	mild	high	weak	no		
sunny	cool	normal	weak	yes	y	y
rain	mild	normal	weak	yes	y	y
sunny	mild	normal	strong	yes		y
overcast	mild	high	strong	yes	y	y
overcast	hot	normal	weak	yes	y	y
rain	mild	high	strong	no		

H1

Actual/Predicted	Yes	No
Yes	7	2
No	0	5

H2

Actual/Predicted	Yes	No
Yes	8	1
No	0	5

Minimum Description Length (MDL)

H1

Actual/Predicted	Yes	No
Yes	7	2
No	0	5

H2

Actual/Predicted	Yes	No
Yes	8	1
No	0	5

- $L(E/H1) = \log(^7C_0) + \log(^7C_2) = 4.39$
- $L(E/H2) = \log(^8C_0) + \log(^6C_1) = 2.59$

- Compression of H1: $L(E) - L(H1) - L(E/H1) = 108.81$
- Compression of H2: $L(E) - L(H2) - L(E/H2) = 105.12$

Result shows that H1 has better compression than H2.

So, we should stop generating more rules and deliver H1 as good model

Markov Modal

- Andrev Markav: A canonical probabilistic model for temporal or sequential data. $X_0 \xrightarrow{A} X_1 \xrightarrow{A} ... \xrightarrow{A} X_n$
- Future is independent of past given the present. Assumption is that the present state encode all the history
- Order specifies how many evidences are important. Order three Markov Modal takes last three data
- iid[2] don't work.
- Temporal data, weather prediction, speech recognition, automatic music generation and handwriting recognition are some of the few applications

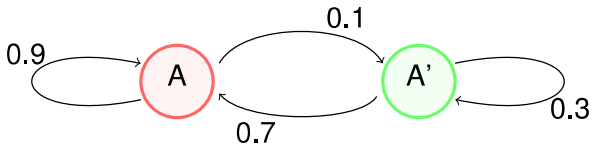
Example:

Suppose a company selling a product A (has market share of 20%), launches a advertise campaign that is expected to retain 90% old customers and attract 70% new. What maximum market share the product A can get?

[2] independent and identically distributed

Markov Modal

Transition diagram



Initial State

$$S_0 = \begin{bmatrix} 0.2 & 0.8 \end{bmatrix}$$

Transition matrix

$$A = \begin{bmatrix} 0.9 & 0.1 \\ 0.7 & 0.3 \end{bmatrix}$$

- $S_0 = \begin{bmatrix} 0.2 & 0.8 \end{bmatrix}$
- $S_1 = S_0 \times A = \begin{bmatrix} 0.74 & 0.26 \end{bmatrix}$
- $S_2 = S_1 \times A = \begin{bmatrix} 0.848 & 0.152 \end{bmatrix}$
- $S_3 = S_2 \times A = \begin{bmatrix} 0.8696 & 0.1304 \end{bmatrix}$

Is it going to saturated?

Stationary matrix

$$\begin{bmatrix} a & b \end{bmatrix} \times A = \begin{bmatrix} a & b \end{bmatrix}$$

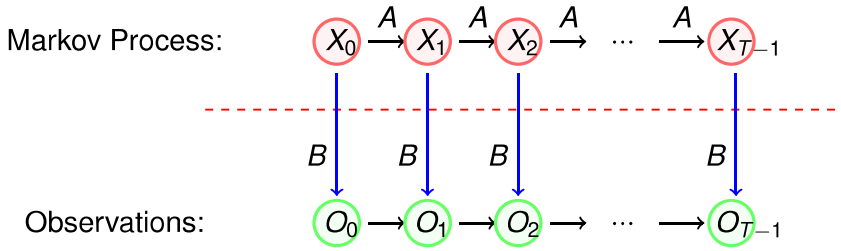
$$\begin{bmatrix} a & b \end{bmatrix} \times \begin{bmatrix} 0.9 & 0.1 \\ 0.7 & 0.3 \end{bmatrix} = \begin{bmatrix} a & b \end{bmatrix}$$

what are a and b? 0.875 and 0.125

- Does it always happen? No, only if matrix is **regular**
- When some power of the matrix has all positive values
- Which of these are regular?

$$\begin{bmatrix} 0.3 & 0.7 \\ 0.1 & 0.9 \end{bmatrix} \qquad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} 0.2 & 0.8 \\ 1 & 0 \end{bmatrix}$$

Hidden Markov Modal (HMM)



Assume we observe coverage (S/M/L) of some news article, to know whether a day was Hot or Cold?

$$B = \begin{array}{c} \\ \text{H} \\ \text{C} \end{array} \begin{array}{c} \begin{array}{ccc} \text{S} & \text{M} & \text{L} \end{array} \\ \begin{bmatrix} 0.1 & 0.4 & 0.5 \\ 0.7 & 0.2 & 0.1 \end{bmatrix} \end{array} \qquad A = \begin{array}{c} \\ \text{H} \\ \text{C} \end{array} \begin{array}{c} \begin{array}{cc} \text{H} & \text{C} \end{array} \\ \begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix} \end{array}$$

Hidden Markov Modal (HMM)

$$B = \begin{array}{c} \\ \text{H} \\ \text{C} \end{array} \begin{array}{c} \begin{array}{ccc} \text{S} & \text{M} & \text{L} \end{array} \\ \begin{bmatrix} 0.1 & 0.4 & 0.5 \\ 0.7 & 0.2 & 0.1 \end{bmatrix} \end{array} \qquad A = \begin{array}{c} \\ \text{H} \\ \text{C} \end{array} \begin{array}{c} \begin{array}{cc} \text{H} & \text{C} \end{array} \\ \begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix} \end{array}$$

- Assume initial configuration for H and C be $\pi = \begin{bmatrix} 0.6 & 0.4 \end{bmatrix}$
- And let observations be S, M, S, L
- Then what is $P(HHCC)$?
  $0.6 \times 0.1 \times (0.7 \times 0.4) \times (0.3 \times 0.7) \times (0.6 \times 0.1) = 0.000212$

Hidden Markov Modal (HMM)

state	probability	normalized probability
$HHHH$	.000412	.042787
$HHHC$	.000035	.003035
$HHCH$	.000706	.073320
$HHCC$	.000212	.022017
$HCHH$	.000050	.005193
$HCHC$	.000004	.000415
$HCCH$	.000302	.031364
$HCCC$	.000091	.009451
$CHHH$	.001098	.114031
$CHHC$	.000094	.009762
$CHCH$	.001882	.195451
$CHCC$	.000564	.058573
$CCHH$	.000470	.048811
$CCHC$	.000040	.004154
$CCCH$	.002822	.293073
$CCCC$	.000847	.087963

Optimum state sequence

- In dynamic programming is CCCH
- HMM choses most probable symbol at each position. (by summing)

	0	1	2	3
P(H)	0.188182	0.519576	0.228788	0.804029
P(C)	0.811818	0.480424	0.771212	0.195971

Optimum state sequence in HMM is ? CHCH

Thank You!

Thank you very much for your attention!

Queries ?

IS-ZC464: MACHINE LEARNING

Lecture-07: Concept Learning

Dr. Kamlesh Tiwari
Assistant Professor
Department of Computer Science and Information Systems,
BITS Pilani, Pilani, Jhunjhunu-333031, Rajasthan, INDIA

September 01, 2018 (WILP @ BITS-Pilani Jul-Nov 2018)

Concept learning

Concept[1] **learning.** Inferring a boolean-valued function (*hypothesis*) from training examples of its input and output.

Consider a dataset D

Attributes

- **Wind:** Strong/Weak
- **Water:** Warm/Cool
- **Forecast:** Same/Change
- **Sky:** Sunny/cloudy/Rainy
- **AirTemp:** Warm/Cold
- **Humidity:** Normal/High

SN	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

[1] Note that "concept" is true function. $h(x) = c(x)$

Concept learning

- For each attribute, the hypothesis will either
  - Indicate by a "?" that any value is acceptable for this attribute,
  - Specify a single required value (e.g., Warm) for the attribute, or
  - Indicate by a "ϕ" that no value is acceptable.
- If some instance x satisfies all the constraints of hypothesis h, then h classifies x as a positive example ($h(x) = 1$).
- A hypothesis that favorite sport is enjoyed only on cold days with high humidity (independent of the values of the other attributes) is represented by the expression $(?, \mathit{Cold}, \mathit{High}, ?, ?, ?)$
- The most general hypothesis that specifies every day is positive is represented by

$$(?, ?, ?, ?, ?, ?)$$

- Most specific hypothesis that no day is positive is given by

$$(\phi, \phi, \phi, \phi, \phi, \phi)$$

Inductive Learning Hypothesis

- The only information available about c is its value over the training examples
- Therefore, algorithms can at best guarantee that the output hypothesis fits the target concept over the training data.

Inductive learning hypothesis

Any hypothesis found to approximate the target function well over a sufficiently large set of training examples will also approximate the target function well over other unobserved examples.

Essentially we are focusing on lower side of efficiency/model-complexity plot.

Concept learning as search

- Concept learning can be viewed as the task of searching through a large space of hypotheses implicitly defined by the hypothesis representation.
- In EnjoySport learning task contains $3 \times 2 \times 2 \times 2 \times 2 \times 2 = 96$ distinct instances.
- So $5 \times 4 \times 4 \times 4 \times 4 \times 4 = 5120$ syntactically distinct hypotheses
- Hypothesis containing one or more ϕ represents the empty set of instances; that is, it classifies every instance as negative
- Although semantically distinct hypotheses are only $1 + (4 \times 3 \times 3 \times 3 \times 3 \times 3) = 973$

General-to-Specific Ordering

- A very useful structure that exists for any concept learning problem is a general-to-specific ordering
- Used for exhaustive search even with infinite hypothesis without explicitly enumerating every hypothesis.

$$h1 = (Sunny, ?, ?, Strong, ?, ?)$$

$$h2 = (Sunny, ?, ?, ?, ?, ?)$$

- As h_2 imposes fewer constraints, it classifies more instances as positive. Any instance classified positive by h_1 will also be classified positive by h_2. Therefore, h_2 is more general than h_1.

$h_j \geq_g h_k$

Let h_j and h_k be boolean-valued functions defined over X. Then h_j is more-general-than-or-equal-to h_k if

$$(\forall x \in X)[(h_k(x) = 1) \rightarrow (h_j(x) = 1)]$$

More-general-than Ordering

It is more useful to consider cases where one hypothesis is strictly more general than the other.

$h_j >_g h_k$

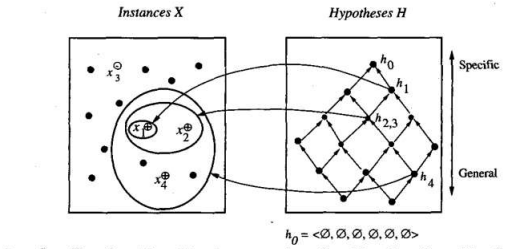
If $h_j \geq_g h_k$ and $h_k \not\geq_g h_j$

- Sometimes we also say h_j is **more-specific-than** h_k when h_k is more-general-than h_j.

Finding a maximally specific hypothesis

FIND-S

1. Initialize h to the most specific hypothesis in H
2. For each positive training instance x
   - For each attribute constraint a_i in h
     - If the constraint a_i is satisfied by x
       - Then do nothing
     - Else replace a_i in h by the next more general constraint that is satisfied by x
3. Output hypothesis h



x_1 = <Sunny Warm Normal Strong Warm Same>, +
x_2 = <Sunny Warm High Strong Warm Same>, +
x_3 = <Rainy Cold High Strong Warm Change>, -
x_4 = <Sunny Warm High Strong Cool Change>, +

h_0 = <∅, ∅, ∅, ∅, ∅, ∅>
h_1 = <Sunny Warm Normal Strong Warm Same>
h_2 = <Sunny Warm ? Strong Warm Same>
h_3 = <Sunny Warm ? Strong Warm Same>
h_4 = <Sunny Warm ? Strong ? ? >

FIND-S

- A way to use more-general-than partial ordering to organize the search.
- FIND-S is guaranteed to output the most specific hypothesis within H that is consistent with the positive training examples.
- No way to determine whether it has found the only hypothesis
- It is unclear whether we should prefer this hypothesis over, say, the most general, or some other hypothesis of intermediate generality.
- If training examples will contain some errors or noise then it can severely mislead FIND-S

LIST-THEN-ELIMINATE ALGORITHM

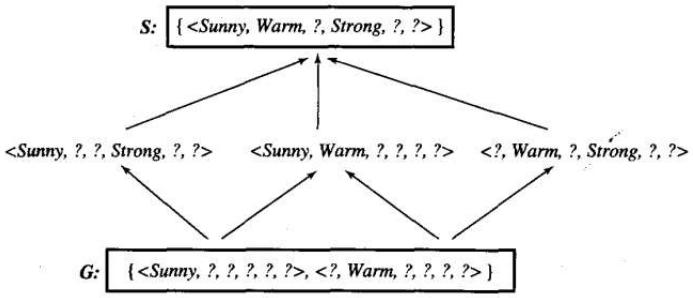
- Outputs a description of the set of all hypotheses consistent with the training examples.
- **Consistent hypothesis.** A hypothesis h is consistent with a set of training examples D if and only if $h(x) = c(x)$ for each example $(x, c(x))$ in D
- **Version space.** Subset of H containing only consistent hypothesis.

The LIST-THEN-ELIMINATE Algorithm

1. $VersionSpace \leftarrow$ a list containing every hypothesis in H
2. For each training example, $\langle x, c(x)\rangle$
   remove from $VersionSpace$ any hypothesis h for which $h(x) \neq c(x)$
3. Output the list of hypotheses in $VersionSpace$

CANDIDATE-ELIMINATION ALGORITHM

- Version space is represented by its most general and least general members.
- **General boundary G,** with respect to hypothesis space H and training data D, is the set of maximally general members of H consistent with D.
- **Specific boundary S,** with respect to hypothesis space H and training data D, is the set of minimally general (*i.e.*, maximally specific) members of H consistent with D.



Version space representation theorem

Version space representation theorem

Let X be an arbitrary set of instances and let H be a set of boolean-valued hypotheses defined over X. Let $c : X \rightarrow \{0, 1\}$ be an arbitrary target concept defined over X, and let D be an arbitrary set of training examples $\{(x, c(x))\}$. For all X, H, c, and D such that S and G are well defined,

$$VS_{H,D} = \{h \in H | (\exists g \in G)(\exists s \in S)(g \geq_g h \geq_g s)\}$$

Proof Sketch. It suffices to show that

1. Every h satisfying the right-hand side of the above expression is in $VS_{H,D}$, and
2. Every member of $VS_{H,D}$ satisfies the right-hand side of the expression.

Version space representation theorem

Every h satisfying the right-hand side; is in $VS_{H,D}$

Let $g \in G$ and $s \in S$ such that $g \geq_g h \geq_g s$. Then by definition s must satisfy all positive examples in D. Since $g \geq_g h$, h would also satisfy all positive examples in D. Similarly, by definition of g it cannot satisfy any negative example in D, and because of $h \geq_g s$ it also cannot satisfy any negative example in D. So this h satisfies all positive and no negative examples of D therefore it is in $VS_{H,D}$

Every member of $VS_{H,D}$ satisfies the right-hand side of the expression

It can be proven by assuming some $h \in VS_{H,D}$ that does not satisfy the right-hand side of the expression. Than showing that this leads to an inconsistency.

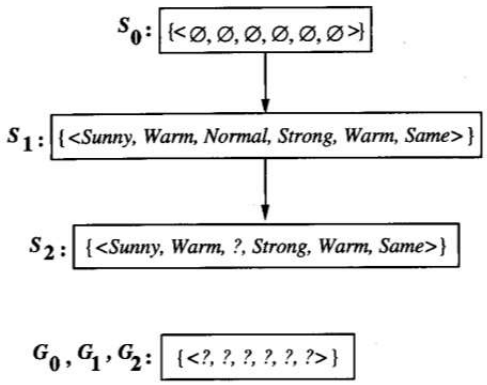
CANDIDATE-ELIMINATION ALGORITHM

Initialize G to the set of maximally general hypotheses in H
Initialize S to the set of maximally specific hypotheses in H
For each training example d, do

- If d is a positive example
  - Remove from G any hypothesis inconsistent with d
  - For each hypothesis s in S that is not consistent with d
    - Remove s from S
    - Add to S all minimal generalizations h of s such that
      - h is consistent with d, and some member of G is more general than h
    - Remove from S any hypothesis that is more general than another hypothesis in S
- If d is a negative example
  - Remove from S any hypothesis inconsistent with d
  - For each hypothesis g in G that is not consistent with d
    - Remove g from G
    - Add to G all minimal specializations h of g such that
      - h is consistent with d, and some member of S is more specific than h
    - Remove from G any hypothesis that is less general than another hypothesis in G

NOTE: $G_0 = \{?, ?, ?, ?, ?, ?\}$ and $S_0 = \{\phi, \phi, \phi, \phi, \phi, \phi\}$

Example

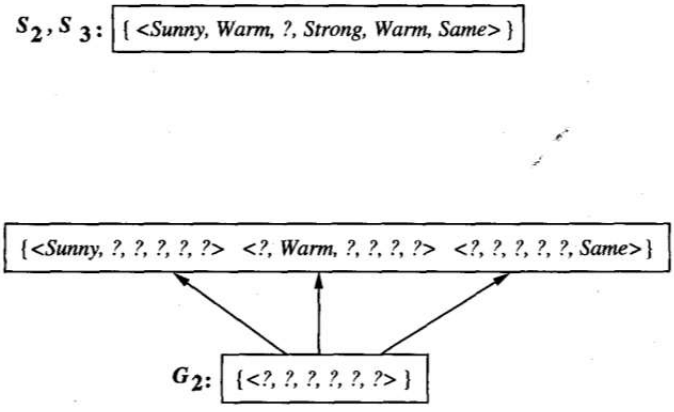


Training examples:

1. <*Sunny, Warm, Normal, Strong, Warm, Same*>, *Enjoy Sport = Yes*

2. <*Sunny, Warm, High, Strong, Warm, Same*>, *Enjoy Sport = Yes*

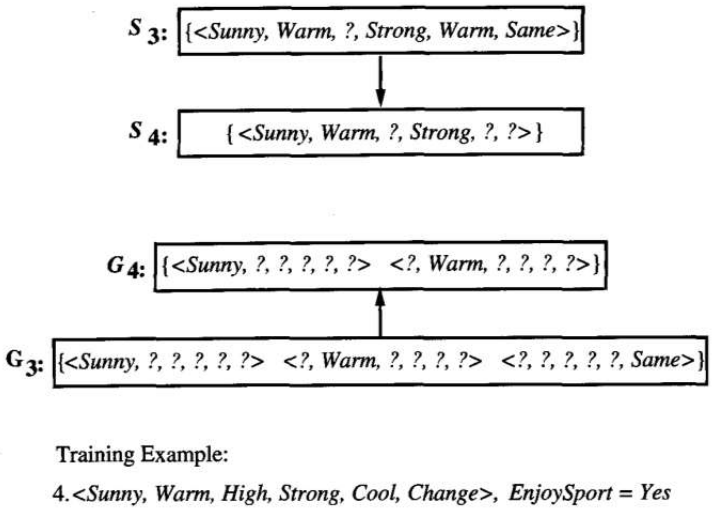
Example



Training Example:

3. <*Rainy, Cold, High, Strong, Warm, Change*>, *EnjoySport=No*

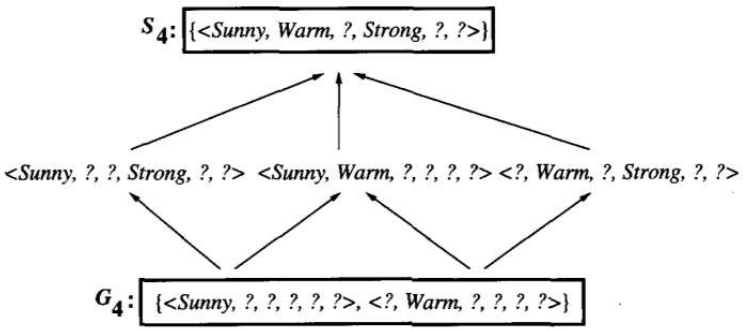
Example



Training Example:

4. <*Sunny, Warm, High, Strong, Cool, Change*>, *EnjoySport* = *Yes*

Example



Training is independent of data ordering.

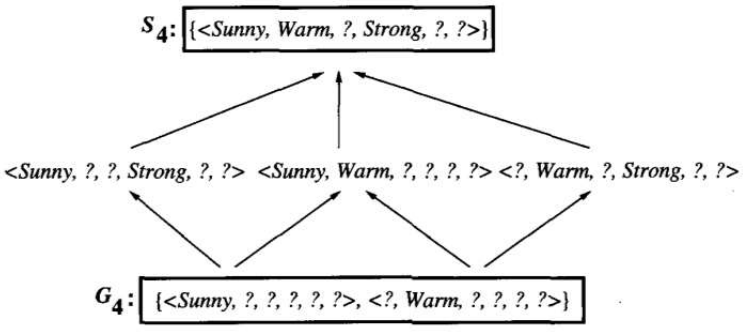
Convergence:

1. No errors in training data
2. There is some hypotheses that correctly describes the target concept

Active learning can help

- If learner is allowed to query (instead of teacher providing examples)
- It could choose contradicting instance that would be classified positive by some hypotheses and negative by others.
- Optimal query strategy is to generate instances that satisfy exactly half the hypotheses in the current version space.
- Size of the version space is reduced by half with each new example, and the correct target concept can therefore be found with only $\lceil \log(|VS|) \rceil$ experiments.

Partially Learned Concept



Classify

1. $A = (Sunny, Warm, Normal, Strong, Cool, Change)$ $+ve$
2. $B = (Rainy, Cold, Normal, Light, Warm, Same)$ $-ve$
3. $C = (Sunny, Warm, Normal, Light, Warm, Same)$?
4. $D = (Sunny, Cold, Normal, Strong, Warm, Same)$ $-ve$

Biased hypothesis space

When hypothesis space is restricted to include only conjunctions of attribute then we may have a problem.

$$\boxed{h = \{\phi, \phi, \phi, \phi, \phi, \phi\}}$$

Tr Example $(Sunny, Warm, Normal, Strong, Cool, Change)$ as $+ve$

$$\boxed{h = (Sunny, Warm, Normal, Strong, Cool, Change)}$$

Tr Example $(Cloudy, Warm, Normal, Strong, Cool, Change)$ as $+ve$

$$\boxed{h = (?, Warm, Normal, Strong, Cool, Change)}$$

Tr Example $(Rainy, Warm, Normal, Strong, Cool, Change)$ as $-ve$

$$\boxed{h = \phi}$$

OOPS: NO hypothesis for this training data!

Unbiased Learner

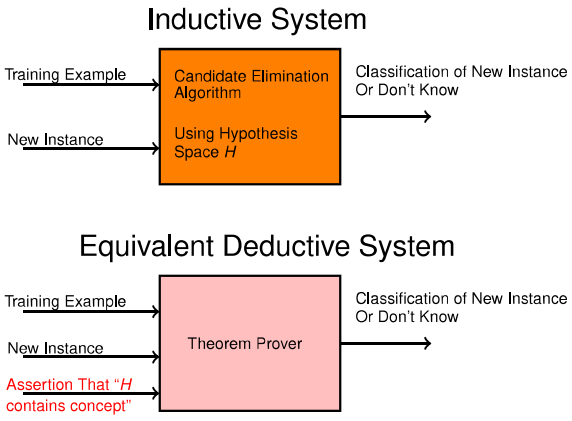
- Obvious solution is to make hypothesis space capable of representing every possible subset of the instances X.
- Use set of all subsets of a set X that is called the power set of X.
- In current example, *EnjoySport* learning task contains $3 \times 2 \times 2 \times 2 \times 2 \times 2 = 96$ distinct instances therefore hypothesis space becomes 2^{96} or 10^{28} as opposed to 973 in case of conjunctions.
- "Sky = Sunny or Sky = Cloudy" is represented as $(Sunny, ?, ?, ?, ?, ?) \vee (Cloudy, ?, ?, ?, ?, ?)$
- While eliminating problems of expressibility it introduce problem in learning algorithm that is now **unable to generalize**
- With three positive (x_1, x_2, x_3) and two negative (x_4, x_5) it looks like $S : \{(x_1 \vee x_2 \vee x_3)\}$ and $G : \{\neg(x_4 \vee x_5)\}$

Futility of Bias-Free Learning

- Is Bias-Free Learning useless?
- **Fundamental property of inductive inference:** a learner that makes no a priori assumptions regarding the identity of the target concept has no rational basis for classifying any unseen instances.
- Inductive learning requires inductive bias
- Let classification of x_i by algorithm L trained on D_c be $L(x_i, D_c)$
- **Inductive inference** is represented as $(D_c \wedge x_i) \succ L(x_i, D_c)$
- Let us represent **inductive bias** by B, then $(\forall x_i \in X)[(B \wedge D_c \wedge x_i) \vdash L(x_i, D_c)]$
- Inductive bias B, is the assumption that $c \in H$
- This assumption enables deduction (proof)

Inductive bias of a learner is defined as the set of additional assumptions B sufficient to justify its inductive inferences as deductive inferences.

Inductive and Deductive Learning



This assertion is the inductive bias. These two systems will produce identical outputs for every input set of training examples and every new instance in X.

Comparison of different learners based on bias

- **ROTE-LEARNER:** Learning corresponds to simply storing each observed training example in memory. New instances are classified by looking them up in memory. If the instance is found, the stored classification is returned. Otherwise, the system refuses to classify the new instance. (no inductive bias)
- **CANDIDATE-ELIMINATION:** New instances are classified only in the case where all members of the current version space agree on the classification. Otherwise, the system refuses to classify the new instance. (bias: target concept can be represented in its hypothesis space)
- **FIND-S:** Finds the most specific hypothesis consistent with the training examples. It then uses this hypothesis to classify all subsequent instances.(bias: target concept can be described in its hypothesis space + all instances are negative instances unless the opposite is entailed by its other knowledge)

Next Class

Decision Tree

Thank You!

Thank you very much for your attention!

Queries ?

(Reference[2])

[2] Book - *Machine Learning* by Tom Mitchell, ch-02

IS-ZC464: MACHINE LEARNING

Lecture-08: Decision Tree

Dr. Kamlesh Tiwari
Assistant Professor
Department of Computer Science and Information Systems,
BITS Pilani, Pilani, Jhunjhunu-333031, Rajasthan, INDIA

September 08, 2018 (WILP @ BITS-Pilani Jul-Nov 2018)

Classification

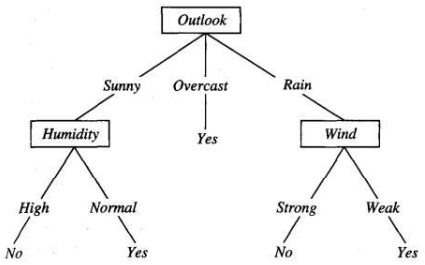
What feature (attributes) would you choose?

Color, texture, weight, density, hardness .....

Decision Tree

Decision Tree

is a method for approximating discrete-valued functions. It is robust to noisy data and capable of learning disjunctive expressions. Primarily useful for classification.



- Each node in the tree specifies a test for some attribute
- Each branch descending from the node corresponds to one of the possible value
- Decision trees represent a disjunction of conjunctions

$$(Outlook = Sunny \wedge Humidity = Normal) \vee (Outlook = Overcast) \vee (Outlook = Rain \wedge Wind = Weak)$$

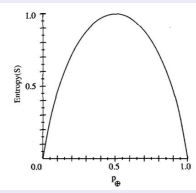
DT is Appropriate when

- Instances are represented by attribute-value pairs
- The target function has discrete output values
- Disjunctive descriptions may be required
- The training data may contain errors
- The training data may contain missing attribute values

Entropy

Characterizes the impurity of an arbitrary collection of examples

$$Entropy(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$



Value between 0 and 1.

0 – when all members are of same class.

1 – if equal number of positive and negative

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rainy	Mild	High	Weak	Yes
D5	Rainy	Cool	Normal	Weak	Yes
D6	Rainy	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rainy	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rainy	Mild	High	Strong	No

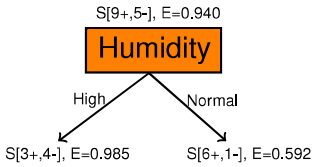
$$\begin{aligned} Entropy([9+,5-]) &= -(9/14)\log_2(9/14) \\ &\quad -(5/14)\log_2(5/14) \\ &= 0.94 \end{aligned}$$

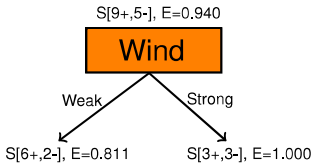
Information Gain

Information Gain of an attribute is the expected reduction in entropy caused by partitioning the examples according to that attribute

$$Gain(S,A) = Entropy(S) - \sum_{v \in Value(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

here S_v contains that data items of S where the value of attribute A is v



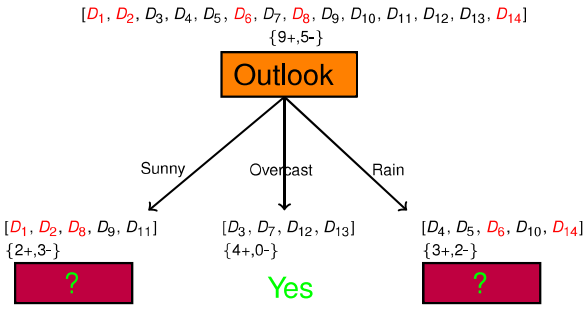


Gain(S,Humidity)$= 0.940 - (7/14)0.985 - (7/14)0.592 = 0.151$

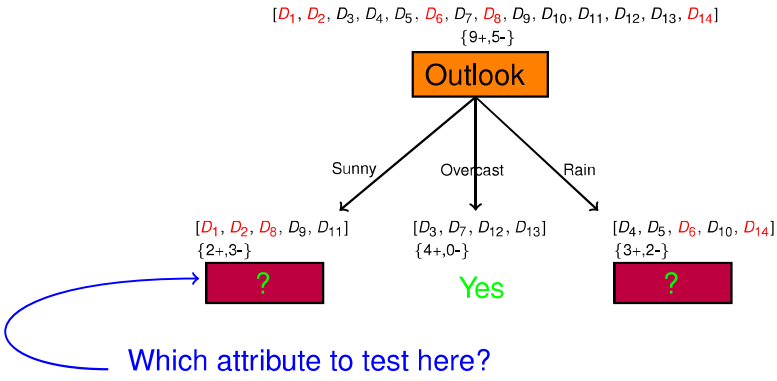
Gain(S,Wind) $= 0.940 - (8/14)0.811 - (6/14)1.000 = 0.048$

Information Gain and Decision Tree

$$
\begin{aligned}
Gain(S, Humidity) &= 0.151 \\
Gain(S, Wind) &= 0.048 \\
Gain(S, Outlook) &= 0.246 \\
Gain(S, Temperature) &= 0.029
\end{aligned}
$$



Recursively apply the same



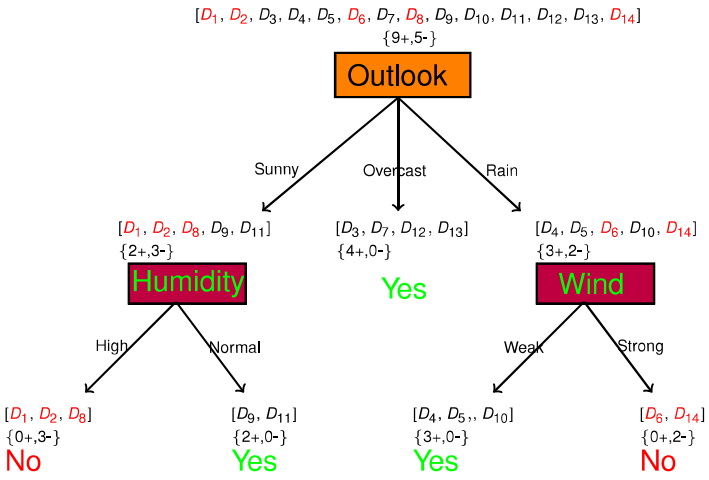
S_{sunny}=[D_1, D_2, D_8, D_9, D_{11}]

Gain(S_{sunny},Humidity) = 0.970 − (3/5)0.0 − (2/5)0.0 = 0.970

Gain(S_{sunny},Temperature)= 0.970 − (2/5)0.0 − (2/5)1.0 = 0.57

Gain(S_{sunny},Wind)= 0.970 − (2/5)1.0 − (3/5)1.0 = 0.019

Recursively apply the same



Decision Tree

A method for approximating discrete-valued functions that is robust to noisy data and capable of learning disjunctive expressions

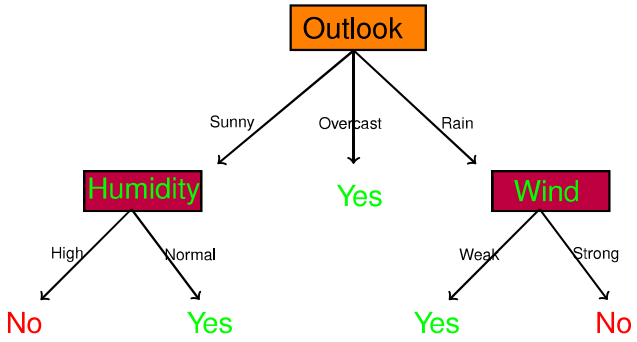
Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rainy	Mild	High	Weak	Yes
D5	Rainy	Cool	Normal	Weak	Yes
D6	Rainy	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rainy	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rainy	Mild	High	Strong	No

What is classification for

$(Overlook, Humidity, Wind) = (Rain, High, Weak)$

ALERT: (missing value) tell me about Temperature?

Example



Classification for $(Overlook, Humidity, Wind) = (Rain, High, Weak)$ is

YES

Iterative-Dichotomiser-3 (ID3) Algorithm By: John Ross Quinlan

Algorithm 1: ID3(Examples,Target_attribute,Attributes)

1 *Examples* are the training data, *Target_attribute* is the attribute whose value is to be predicted by the tree. *Attributes* is a list of other attributes that may be tested by the learned decision tree. Algorithm returns a decision tree that correctly classify the given example.

2 Create a single-node tree *Root*

3 **IF** *Examples* are all +ve **THEN return** *Root* with label +ve

4 **IF** *Examples* are all -ve **THEN return** *Root* with label -ve

5 **IF** *Attributes* $= \phi$ **THEN return** *Root* with most common *Target_attribute*

6 A $\leftarrow$ attribute from *Attributes* that best classifies *Examples*

7 Decision attribute for *Root* $\leftarrow$ A

8 **foreach** *value* v_i *of A* **do**

9 Add a new tree branch below Root, to test A=v_i

10 Examples$_{v_i}$ $\leftarrow$ subset of *Examples* having value v_i for A

11 **IF** Examples$_{v_i} = \phi$ **THEN** below this branch add a leaf with label = most common value of *Target_attribute* in *Examples*

12 **ELSE** below this branch add subtree ID3(Examples$_{v_i}$,*Target_attribute*,*Attributes*-{*A*})

13 **return** *Root*

Issues Decision Tree

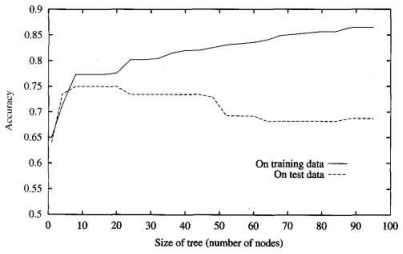
Given a collection of training examples, there could be many decision trees consistent with the examples

- ID3 search strategy
  - selects in favor of shorter trees over longer ones, and
  - selects trees that place the attributes with highest information gain closest to the root
- Issues in decision trees include
  1. how deeply to grow
  2. handling continuous attributes
  3. choosing an appropriate attribute selection measure
  4. missing attribute values
  5. attributes with differing costs, and
  6. improving computational efficiency

Issues in Decision Tree

Overfitting

Given a hypothesis space H, a hypothesis $h \in H$ is said to overfit the training data if there exists some alternative hypothesis $h' \in H$, such that h has smaller error than h' over the training examples, but h' has a smaller error than h over the entire distribution of instances.



- This can occur when training examples contain random errors or noise.

Approaches to avoid overfitting

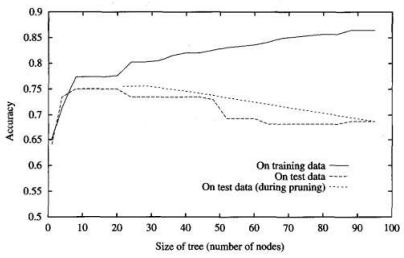
- Stop growing the tree earlier, before it reaches the point where it perfectly classifies the training data
- Allow the tree to overfit the data, and then post-prune the tree

Criterion to determine the correct final tree size include:

- Use a separate set of examples (called validation), distinct from the training examples, to evaluate the utility of post-pruning nodes from the tree
- Use all the available data for training, but apply a statistical test (such as chi-square test) to estimate whether expanding (or pruning) a particular node is likely to produce an improvement beyond the training set.
- Use an explicit measure of the complexity for encoding the training examples (such as Minimum Description Length) and the decision tree, halting growth of the tree when this encoding size is minimized.

Reduced-error pruning

- Pruning a decision node consists of removing the subtree rooted at that node, making it a leaf node, and assigning it the most common classification of the training affiliated with that node.
- Nodes are removed only if the resulting pruned tree performs no worse than-the original over the validation set
- Nodes are pruned iteratively, always choosing the node whose removal must increases the decision tree accuracy over the validation set

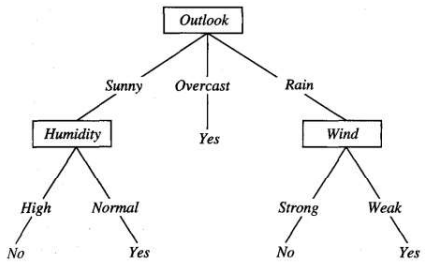


- Drawback: number of examples available for training is reduced.

Rule post-pruning

- Infer the decision tree from the training set, growing the tree until the training data is fit as well as possible and allowing overfitting to occur
- Convert the learned tree into an equivalent set of rules by creating one rule for each path from the root node to a leaf node
- Prune (generalize) each rule by removing any preconditions that result in improving its estimated accuracy
- Sort the pruned rules by their estimated accuracy, and consider them in this sequence when classifying subsequent instances

Rule post-pruning



- If $(Outlook = Sunny) \wedge (Humidity = High)$ Then $PlayTennis = No$

Each rule is pruned by removing any antecedent, or precondition, whose removal does not worsen its estimated accuracy

(on validation or test?)

Next Class

1. C4.5 and Random Forest
2. Clustering with K-Means

Syllabus of Mid-Sem (regular)

Till next class.

Thank You!

Thank you very much for your attention! (Reference[1])

Queries ?

[1] [1] Book - *Machine Learning*, ch-3, Tom M. Mitchell. [2] Decision Tree 1: how it works https://www.youtube.com/watch?v=eKD5gxPPeY0

IS-ZC464: MACHINE LEARNING

Lecture-09: D-Tree (contd..), Random-Forest, K-NN, K-means

Dr. Kamlesh Tiwari
Assistant Professor
Department of Computer Science and Information Systems,
BITS Pilani, Pilani, Jhunjhunu-333031, Rajasthan, INDIA

September 15, 2018 (WILP @ BITS-Pilani Jul-Nov 2018)

Recap: Decision Tree

- **Decision Tree** represents disjunction of conjunctions. Many consistent decision trees are possible for same dataset
- **ID3** search strategy partition according to the attribute with highest

$$Gain(S, A) = Entropy(S) - \sum_{v \in Value(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

  Thereby selecting in the favor of shorter trees, and ones that place the attributes with highest information gain closest to the root
- **Issues** in decision trees include
  1. Overfitting (how deep to grow?)
  2. Handling continuous attributes (information gain and heuristics?)
  3. Choosing an appropriate attribute selection measure (Gain Ratio?)
  4. Missing attribute values (most common or probability?)
  5. Attributes with differing costs, and
  6. Improving computational efficiency

Recap: Decision Tree

Given following data

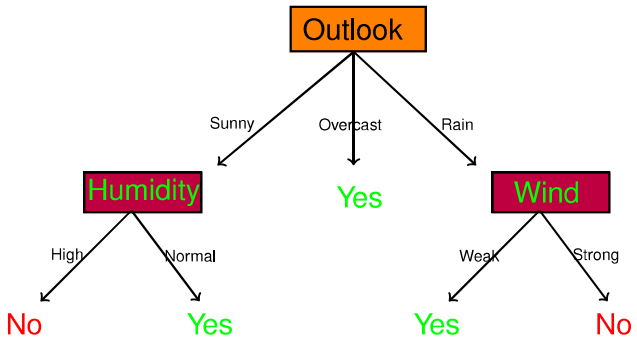
Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rainy	Mild	High	Weak	Yes
D5	Rainy	Cool	Normal	Weak	Yes
D6	Rainy	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rainy	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rainy	Mild	High	Strong	No

Find classification for

$(Outlook = Rain, Humidity = High, Wind = Weak)$

ALERT: (missing value) what is Temperature?

Recap: Iterative-Dichotomiser-3 (ID3) Algorithm



Classification for $(Outlook = Rain, Humidity = High, Wind = Weak)$ is

YES

C4.5 Algorithm

It is a statistical classifier that extends ID3 algorithm by dealing with both **continuous** and discrete attributes, **missing values** and **pruning** trees after construction.

- Determines pessimistic estimate by calculating the rule accuracy over the training example, then calculating the standard deviation in this estimated accuracy assuming a binomial distribution.
- For large data sets, the pessimistic estimate is very close to the observed accuracy (the standard deviation is very small), whereas it grows further from the observed accuracy as the size of the data set decreases.
- Although this heuristic method is not statistically valid, it has nevertheless been found useful in practice.

Incorporating Continuous-Valued Attributes

- Dynamically define new discrete-valued attributes that partition the continuous attribute value into a discrete intervals
- Dynamically create a new boolean attribute A_c that is true if $A < c$ and false otherwise
- The question is, how to select the best value for the threshold c
- Pick a threshold, c, that produces the greatest information gain
- By sorting the examples according to the continuous attribute A, then identifying adjacent examples that differ in their target classification, we can generate a set of candidate thresholds midway between the corresponding values of A.
- Continuous attribute can also be splits into multiple intervals

Example

$S =$

Temperature	40	48	60	72	80	90
playTennis	N	N	Y	Y	Y	N

$$Gain(S, A) = Entropy(S) - \sum_{v \in Value(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

- Entropy(S)=1
- Partitioned at 44 produces [0+,1-] and [3+,2-] having entropies 0 and $-\frac{3}{5}\log\frac{3}{5} - \frac{2}{5}\log\frac{2}{5} = 0.29$
  $Gain(S, A_{44}) = 1 - \frac{1}{6}0 - \frac{5}{6}0.29 = 0.243$

Similarly

- Determine $Gain(S, A_{54})$?
- Determine $Gain(S, A_{66})$?
- Determine $Gain(S, A_{76})$?
- Determine $Gain(S, A_{85})$?

Alternative Measures for Selecting Attributes

- There is a natural bias in the information gain measure that favors attributes with many values over those with few values
- Consider attribute *Date*, it would have the highest information gain of any of the attributes. Because *Date* alone perfectly predicts the target attribute over the training data.
- Thus, it would be selected as the decision attribute for the root node of the tree
- **Gain Ratio** is a measure that penalizes attributes such as *Date* by incorporating a term, called split information, that is sensitive to how broadly and uniformly the attribute splits the data:

$$splitInformation(S, A) = -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log \frac{|S_i|}{|S|}$$

- **Gain Ratio:** measure is defined as

$$GainRatio(S, A) = \frac{Gain(S, A)}{splitInformation(S, A)}$$

Alternative Measures for Selecting Attributes

- One practical issue arises using **GainRatio** when denominator is zero or very small ($|S_i| \lll |S|$)
- This makes **GainRatio** undefined or very large
- We can adopt some heuristic such as first calculating the Gain of each attribute, then applying the **GainRatio** test only considering those attributes with above average Gain
- An alternative to the **GainRatio**, is a distance-based measure introduced by Lopez de Mantaras (1991)[1]. Each attribute is evaluated based on the distance between the data partition it creates and the perfect partition (i.e., the partition that perfectly classifies the training data). The attribute whose partition is closest to the perfect partition is chosen.

[1] De Mántaras, R López, "A distance-based attribute selection measure for decision tree induction", Machine learning, 6(1), pp 81–92, Springer-1991

Missing Attribute Values

- In a medical domain, in which we wish to predict patient outcome based on various laboratory tests, it may be that the lab test Blood-Test-Result is available only for a subset of the patients.
- let for $< x, c(x) >$ we do not have value of $A(x)$
- One strategy for dealing with the missing attribute value is to assign it the value that is most common among training examples at node n
- A second, more complex procedure is to assign a probability to each of the possible values of A

Attributes with Differing Costs

- Instance attributes may have associated costs
- Consider attributes Temperature, BiopsyResult, Pulse, BloodTestResult
- We would prefer decision trees that use low-cost attributes where possible, relying on high-cost attributes only when needed to produce reliable classifications.
- We might divide the Gain by the cost of the attribute, so that lower-cost attributes would be preferred (without guarantee it puts a bias)
- Tan and Schlimmer for robots, demonstrated more efficient recognition strategies using $Gain^2(S, A)/Cost(A)$
- Nunez for medical diagnosis rules proposed (for $w \in [0, 1]$)

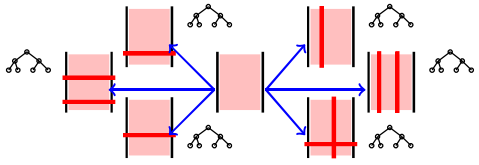
$$\frac{2^{Gain(S,A)} - 1}{(Cost(A) + 1)^w}$$

Random Forest

Combination of learning models (ensemble of classifiers) increases classification accuracy. Averaging compensates noise. Resulting model has low variance

Random Forest[2] is a large collection of decorrelated decision trees

- Training data is divided into a number of sub-sets (delete row or columns) that may have overlap (or subset of attributes)



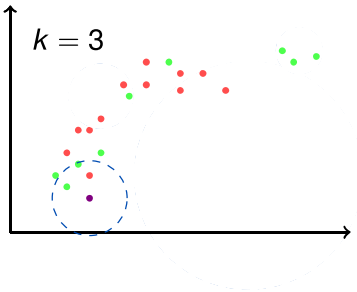
- For every subset, built a decision tree
- To classify new element: **apply voting**

[2] Leo Breiman, "Random Forests", ML 45, pp 5-32, 2001

K Nearest Neighbor (KNN)

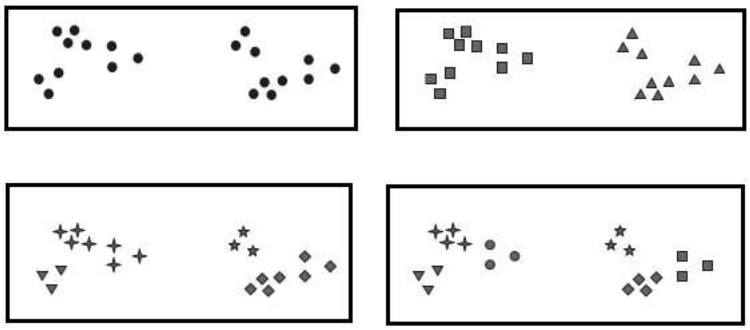
You are most likely as your friends (Bias)

- Two step algorithm
  1. Search k other datum points (most difficult part)
  2. Apply majority voting
- A lazy learner
- To avoid ties, k should NOT be a multiple of number of classes
- Small k is sensitive to noise and large one has high bias



Clustering

Grouping data based on their homogeneity (similarity or closeness).



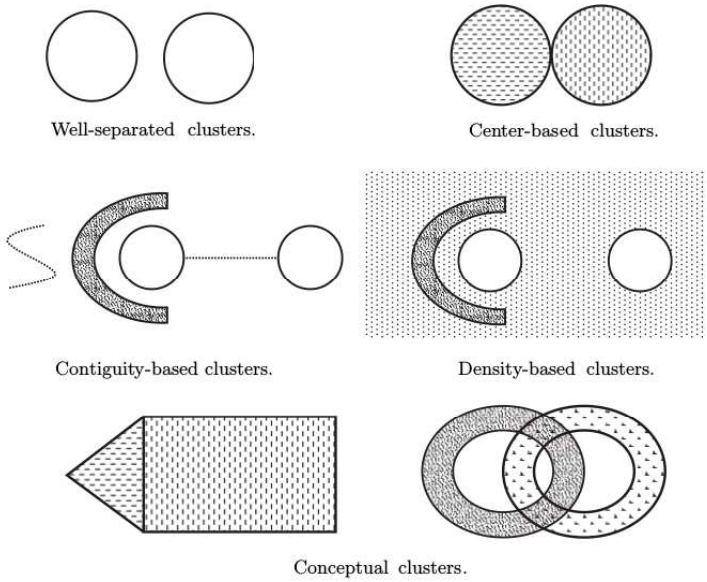
Objects within a group are similar (or related) and are different from the objects in other groups. When it is better?

Clustering

- **Unsupervised** in nature (i.e. right answers are not known)
- Clustering is useful to 1) Summarization, 2) Compression, and 3) Efficiently Finding Nearest Neighbors
- **Type:**
  - Hierarchical (nested) versus Partitional
  - Exclusive versus Overlapping versus Fuzzy
  - Complete versus Partial
- **K-means:** This is a prototype-based[3], partitional clustering technique that attempts to find a user-specified number of clusters (K), which are represented by their centroids.

[3] object is closer (more similar) to a prototype

Clustering Approaches



K-means Algorithm

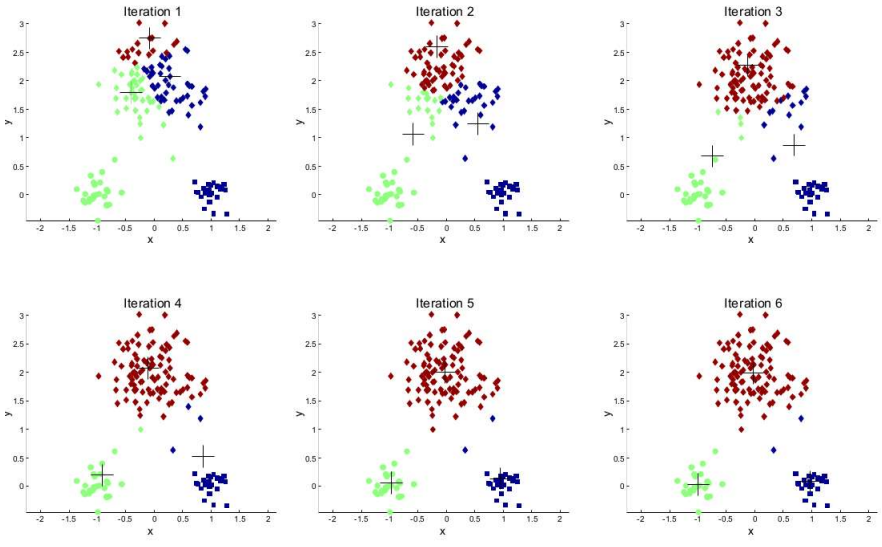
Number of clusters *i.e.* the value of K is provided by the user

Algorithm 3: K-means

1. Randomly select K points as centroids
2. **repeat**
3. | **foreach** *datum point* d_i **do**
4. | | Assign d_i to one of the closest centroids (thereby forming K clusters)
5. | Recompute centroid (mean) for each cluster
6. **until** *The centroids converge*;

Closeness is measured by **Euclidean distance**, cosine similarity, correlation, Bregman divergence *etc*

K-means in Action



Evaluation of K-means[4]

For a given data set $\{x_1, x_2, ..., x_n\}$, let K-means partitions it in $\{S_1, S_2, ..., S_K\}$ then the objective is

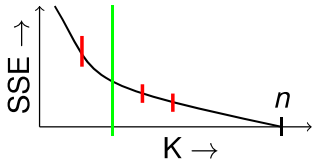
$$\underset{S}{\operatorname{argmin}} \sum_{i=1}^{K} \sum_{x \in S_i} dist^2(x, \mu_i)$$

where μ_i corresponds to i^{th} centroid. $\mu_i = \frac{1}{|S_i|} \sum_{x \in S_i} x$

- Typical choice for *dist* function is Euclidean Distance

How to proceed?

- Choose a K (How?)
  - Run K-means algorithm multiple times
  - Choose clusters corresponding to the one that minimized sum of squared error (SSE)
- If $K == n$, no error.
- Good clustering has smaller K

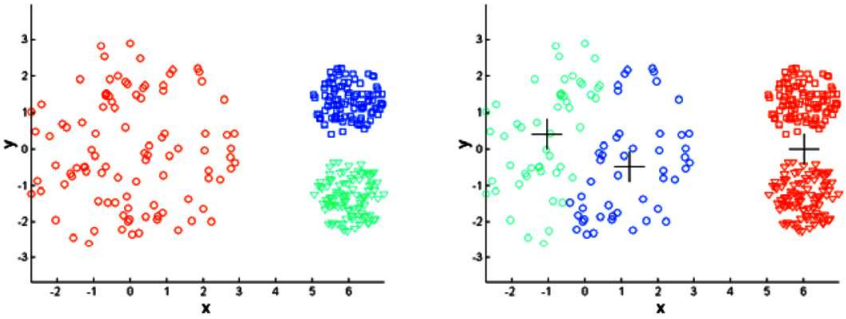


[4] Hamerly, Greg and Elkan, Charles, "Learning the k in k-means", pp 281–288, NIPS-2003

Evaluation of K-means

- **Choosing K:** 1) Domain Knowledge, 2) Preprocessing with another algorithm, 3) Iteration on K
- **Initialization of Centers:** 1) Random point in space, 2) Random point of data, 3) look for dense region, 4) Space uniformly in feature space
- **Cluster Quality:** 1) Diameter of cluster verses Inter-cluster distance, 2) Distance between members of a cluster and the cluster center, 3) Diameter of smallest sphere, 4) Ability to discover hidden patterns

Limitations of K-means



- Has problem when data has
  - Different size clusters
  - Different densities
  - Non-globular shape
- Handling Empty Clusters
- When there are outliers
- Updating Centroids Incrementally

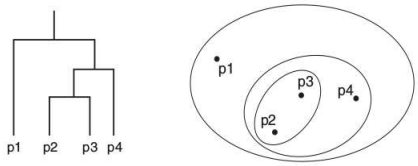
Important Note:

- K-Means and K-NN are different (K nearest neighbors)

K-NN is a **supervised** approach for **classification**

Other Clustering Approaches

- **K-Medoids:** chooses data point as center and minimizes a sum of pairwise dissimilarities. Resistance to noise and/or outliers
- **Agglomerative Hierarchical Clustering:** repeatedly merging the two closest clusters until a single (Single Link)

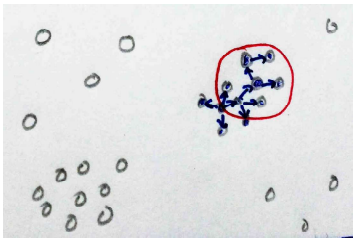


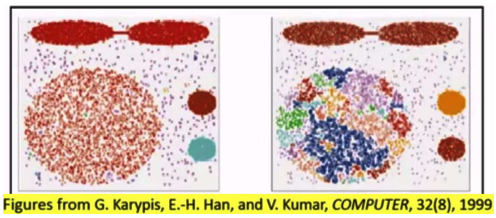
- **DBSCAN:** density-based clustering algorithm that produces a partitional clustering, in which the number of clusters is automatically determined by the algorithm.

DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a spatial clustering algorithm of KDD96

- Parameters (Eps/MinPts) and points (core/border/noise)
- Uses DFS

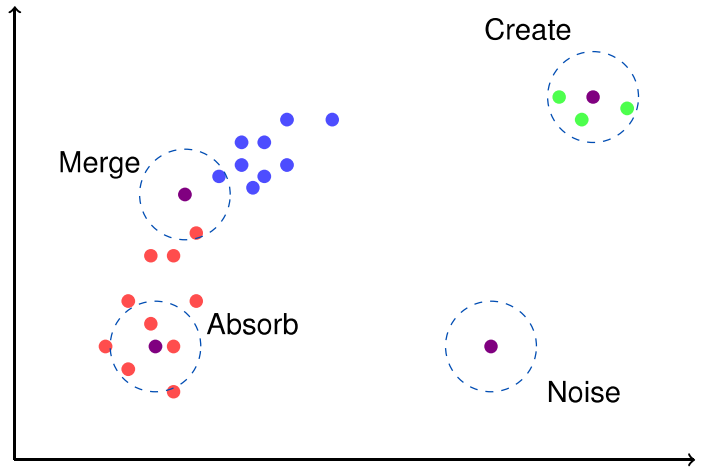




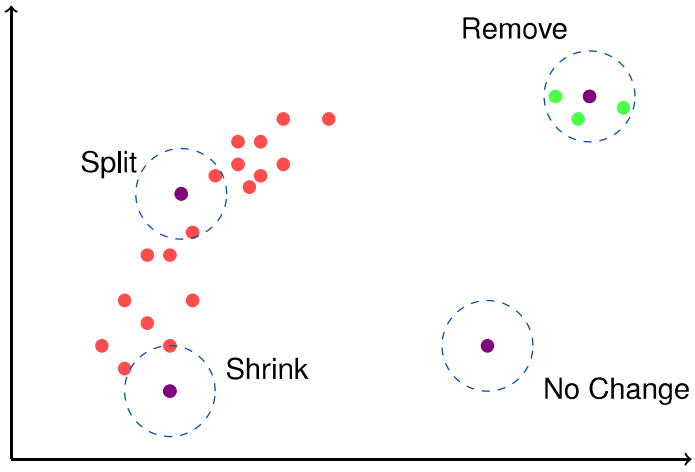
Figures from G. Karypis, E.-H. Han, and V. Kumar, *COMPUTER*, 32(8), 1999

- Disadvantage: Sensitive to parameters
- Advantage: 1) clusters of arbitrary shape, 2) Can handle dynamic databases

Incremental DBSCAN (Addition)



Incremental DBSCAN (Deletion)



Next Class

1. Naive Bayes Classifier
2. Linear Model for Regression

Thank You!

Thank you very much for your attention! (Reference[5])

Queries ?

[5] [1] Book - *Machine Learning*, ch-3, Tom M. Mitchell. [2] Decision Tree 1: how it works https://www.youtube.com/watch?v=eKD5gxPPeY0, [2] An efficient k-means clustering algorithm: Analysis and implementation, T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu, IEEE Transaction on Pattern Analysis and Machine Intelligence, pp 881–892, 24 (2002) [3] https://www-users.cs.umn.edu/∼kumar/dmbook/ch8.pdf