Classification

Yi-Shin Chen
Institute of Information Systems and Applications
Department of Computer Science
National Tsing Hua University
yishin@gmail.com

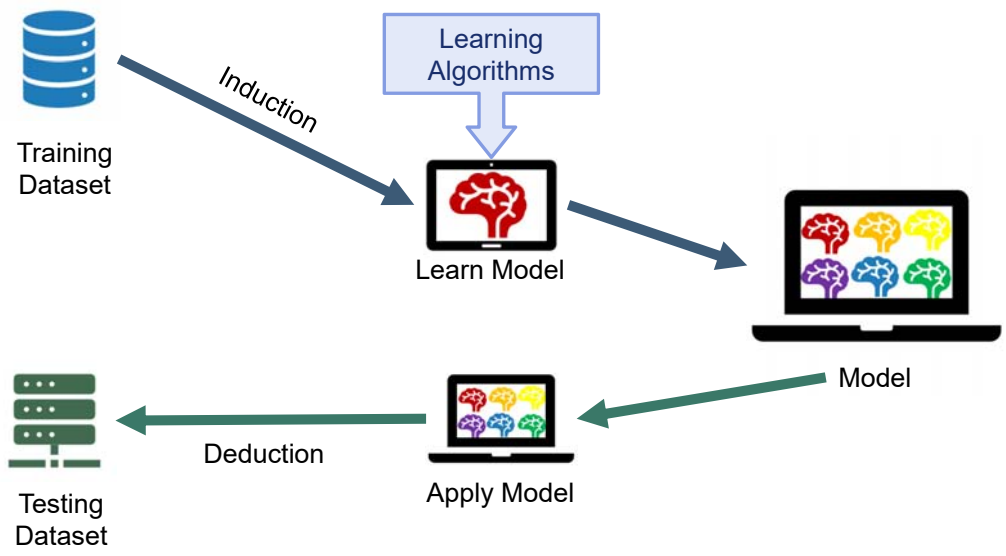
Classification: Definition

- Given a collection of records (*training set*)
  - Each record contains a set of *attributes*
  - One of the attributes is the *class*
- Find a *model* for class attribute:
  - The model forms a function of the values of other attributes
- Goal: previously unseen records should be assigned a class as accurately as possible.
  - A *test set* is needed
    - To determine the accuracy of the model

Supervised vs. Unsupervised Learning

- Supervised learning
  - Supervision: The training data with class labels
  - New data is classified based on the training set
- Unsupervised learning
  - The class labels of training data is unknown
  - Given a set of measurements with the aim of establishing the existence of clusters in the data

Illustrating Classification Task



Examples of Classification Task

- Predicting tumor cells as benign or malignant
- Classifying credit card transactions as legitimate or fraudulent
- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil
- Categorizing news stories

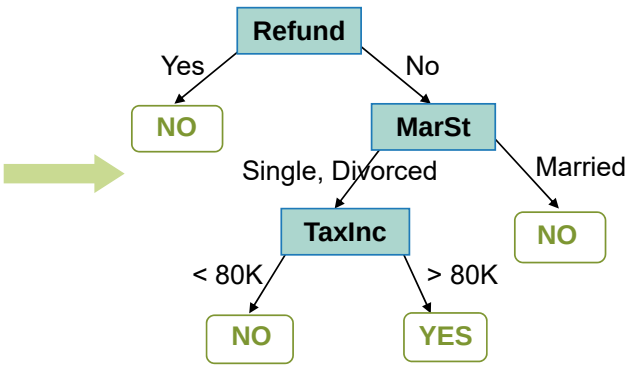
Classification Techniques

- Decision Tree based Methods
- Rule-based Methods
- Memory based reasoning
- Neural Networks
- Naïve Bayes and Bayesian Belief Networks
- Support Vector Machines

Examples of Decision Tree

	categorical	categorical	continuous	class
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data



Model: Decision Tree

Issues Regarding Classification: Data Preparation

- Data cleaning
  - Preprocess data in order to reduce noise and handle missing values
- Relevance analysis (feature selection)
  - Remove the irrelevant or redundant attributes
- Data transformation
  - Generalize and/or normalize data

Issues Regarding Classification: Evaluating Classification Methods

- Predictive accuracy
- Speed and scalability
- Robustness
  - Handling noise and missing values
- Scalability
- Interpretability:
  - Understanding and insight provided by the model
- Goodness of rules

Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical
    - If continuous-valued, they are discretized in advance
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)

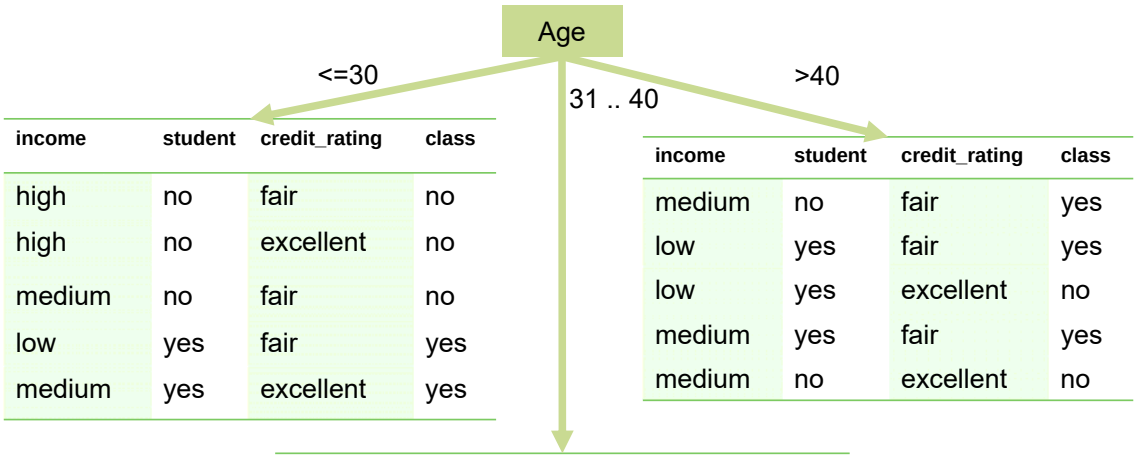
Algorithm for Decision Tree Induction (Contd)

- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no samples left

Training Dataset

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Partition Example



income	student	credit_rating	class
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	student	credit_rating	class
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	student	credit_rating	class
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

Tree Induction

- Greedy strategy.
  - Split the records based on an attribute test
    - That optimizes certain criterion
- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

How to Determine the Best Split

- Greedy approach:
  - Nodes with homogeneous class distribution are preferred
- Need a measure of node impurity
  - Entropy
  - Gini Index
  - Misclassification error

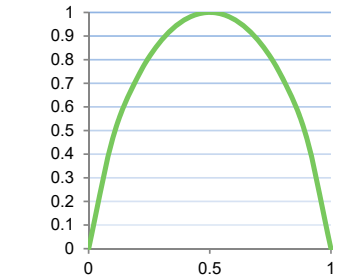
C0: 5 C1: 5	C0: 9 C1: 1
Non-homogeneous	Homogeneous
High degree of impurity	Low degree of impurity

Entropy

- Entropy measures the amount of randomness or surprise or uncertainty
- Entropy is defined as:

$$H(p_1,\cdots p_n)=\sum_{i=1}^{n}\left(p_i\times\log\frac{1}{p_i}\right)=-\sum_{i=1}^{n}(p_i\times\log p_i)$$

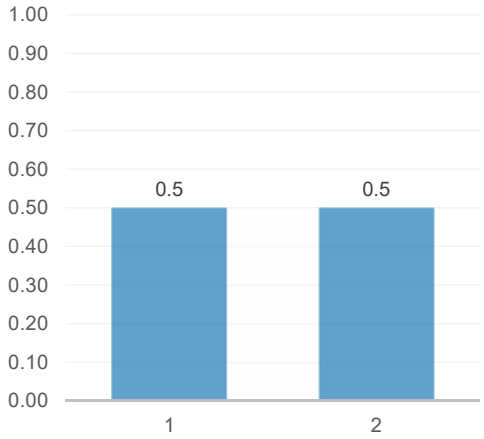
$$where\sum_{i=1}^{n}(p_i)=1$$



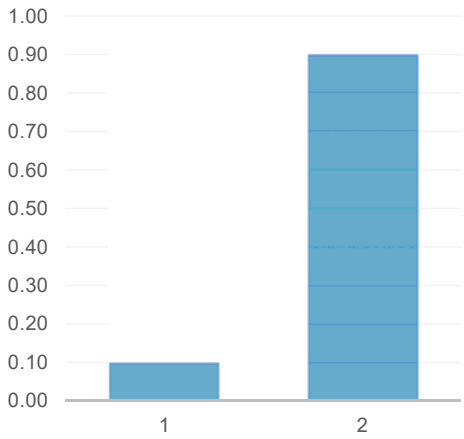
Entropy(p,1-p)

Example of Entropy

Entropy is a measure of '**uncertainty**' in a probability distribution.



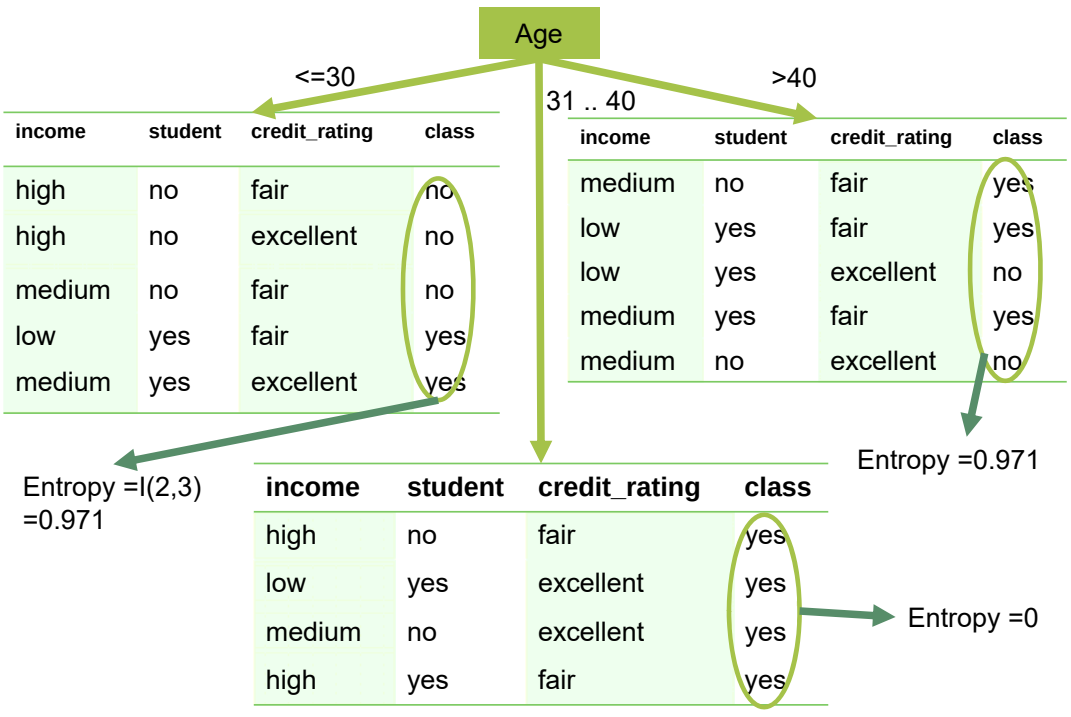
Probability(event 1) = 0.5
Probability(event 2) = 0.5
Entropy = 1.0



Probability(event 1) = 0.1
Probability(event 2) = 0.9
Entropy = 0.469

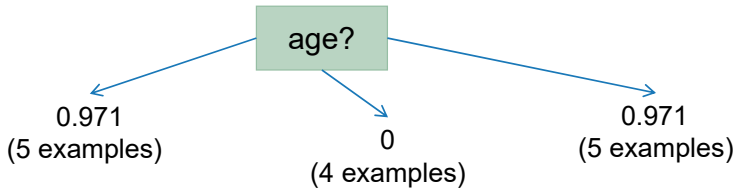
Attribute Selection Measure: Information Gain

- Select the attribute with the highest information gain (△)
  - $$\Delta = I(parent) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$



Attribute Selection Measure: Information Gain

- Select the attribute with the highest information gain (△)
  - $$\Delta = I(parent) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$



$$E(age) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0) + \frac{5}{14} I(3,2) = 0.694$$

$$Gain(age) = I(9,5) - E(age) = 0.246$$

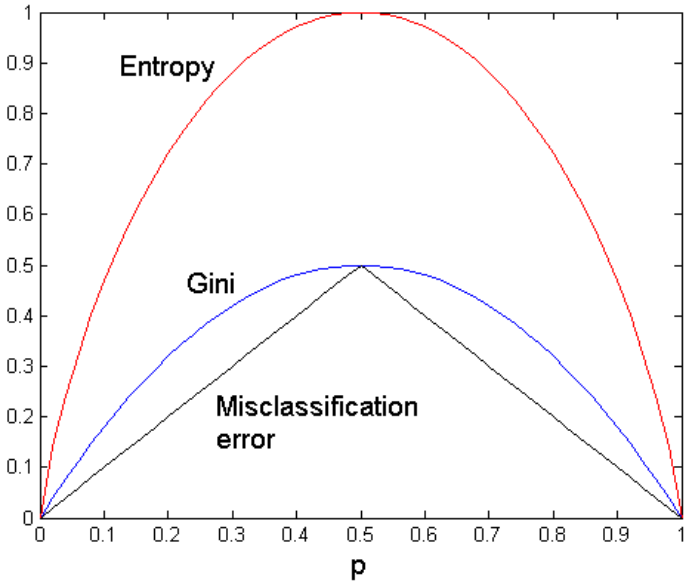
Selection Attribute

Age	Income	Student	Credit Rating	Buys Computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

$Gain(income) = 0.029$
$Gain(student) = 0.151$
$Gain(credit\_rating) = 0.048$
$Gain(age) = 0.246$

Comparison among Splitting Criteria

For a 2-class problem:



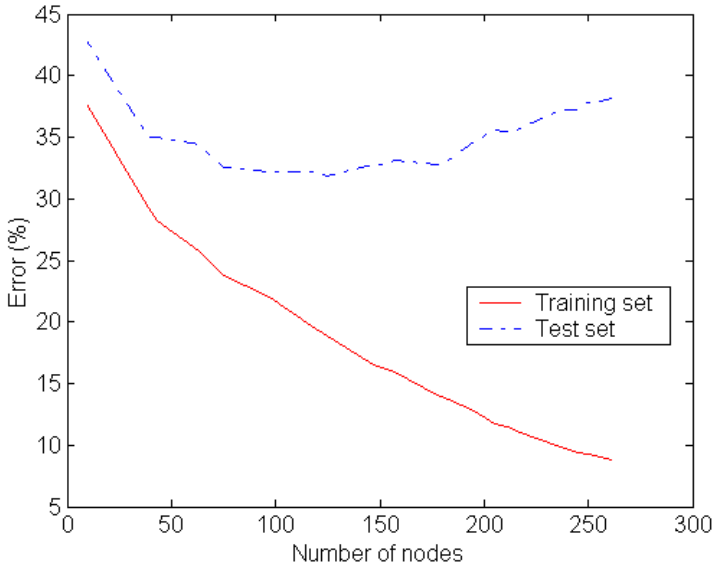
Decision Tree Based Classification

- Advantages:
  - Inexpensive to construct
  - Extremely fast at classifying unknown records
  - Easy to interpret for small-sized trees
  - Accuracy is comparable to other classification techniques for many simple data sets

Practical Issues of Classification

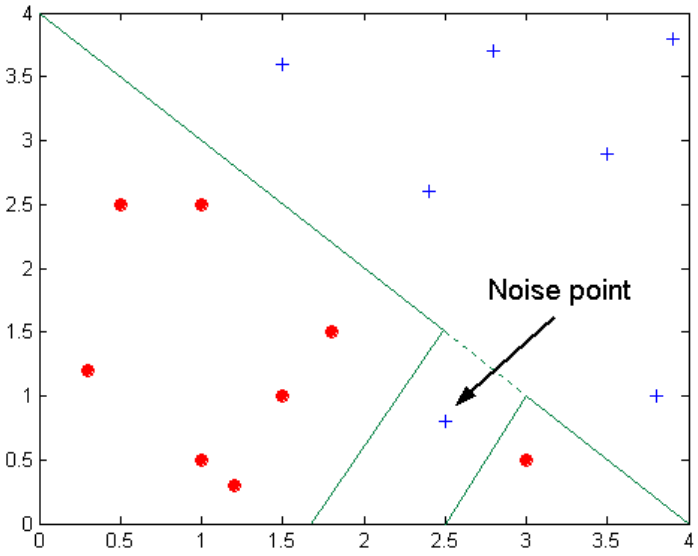
- Under fitting and Overfitting
- Missing Values
- Costs of Classification

Underfitting and Overfitting



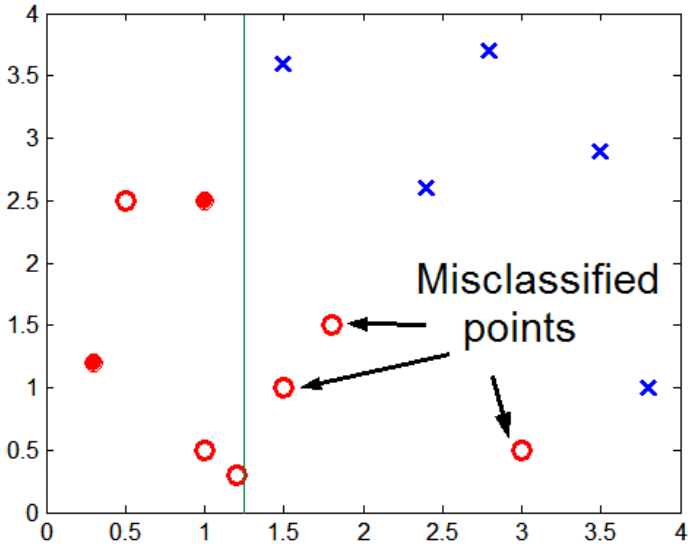
Underfitting: when model is too simple, both training and test errors are large

Overfitting due to Noise



Decision boundary is distorted by noise point

Overfitting due to Insufficient Examples



Avoid Overfitting in Classification

- Two approaches to avoid overfitting
  - Prepruning
    - Halt tree construction early
    - Difficult to choose an appropriate threshold
  - Postpruning
    - Remove branches from a “fully grown” tree
    - Use a set of data different from the training data to decide which is the “best pruned tree”

Metrics for Performance Evaluation

- Focus on the predictive capability of a model
  - Rather than how fast it takes to classify or build models, scalability, etc.
- Confusion Matrix:

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive) b: FN (false negative)

c: FP (false positive) d: TN (true negative)

Metrics for Performance Evaluation

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

- Most widely-used metric:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Limitation of Accuracy

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10
- If model predicts everything to be class 0, accuracy is 9990/10000 = 99.9 %
- Accuracy is misleading because model does not detect any class 1 example

Cost-Sensitive Measures

- Precision is biased towards C(Yes|Yes) & C(Yes|No)
- Recall is biased towards C(Yes|Yes) & C(No|Yes)
- F-measure is biased towards all except C(No|No)

$$\text{Precision (p)} = \frac{a}{a+c}$$

$$\text{Recall (r)} = \frac{a}{a+b}$$

$$\text{F-measure (F)} = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

$$\text{Weighted Accuracy} = \frac{w_1 a + w_4 d}{w_1 a + w_2 b + w_3 c + w_4 d}$$

Test of Significance

- Given two models:
  - Model M1: accuracy = 85%, tested on 30 instances
  - Model M2: accuracy = 75%, tested on 5000 instances
- Can we say M1 is better than M2?
  - How much confidence can we place on accuracy of M1 and M2?
  - Can the difference in performance measure be explained as a result of random fluctuations in the test set?

Confidence Interval for Accuracy

- Prediction can be regarded as a Bernoulli trial
  - A Bernoulli trial has 2 possible outcomes
    - Possible outcomes for prediction: correct or wrong
  - Collection of Bernoulli trials has a Binomial distribution:
    - x ≈ Bin(N, p) x: number of correct predictions
    - e.g: Toss a fair coin 50 times, how many heads would turn up?
      - Expected number of heads = N × p = 50 × 0.5 = 25

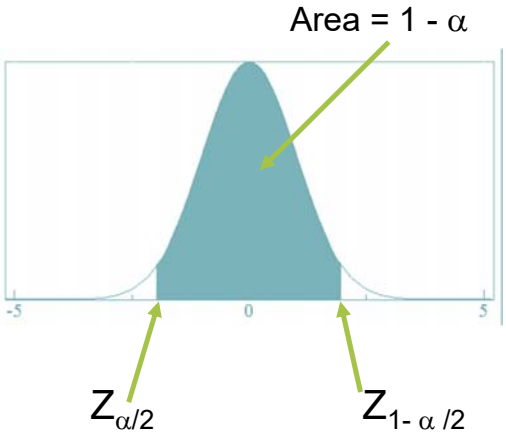
- Given x (# of correct predictions) or equivalently, accuracy (a_c)=x/N, and N (# of test instances)

  Can we predict p (true accuracy of model)?

Confidence Interval for Accuracy

- For large test sets (N > 30),
  - a_c has a normal distribution with mean p and variance p(1-p)/N

$$P(Z_{\alpha/2} < \frac{a_c - p}{\sqrt{p(1-p)/N}} < Z_{1-\alpha/2}) = 1-\alpha$$



  - Confidence Interval for p:

$$p = \frac{2\times N\times a_c + Z^2_{\alpha/2} \pm Z_{\alpha/2}\sqrt{Z^2_{\alpha/2} + 4\times N\times a_c - 4\times N\times a_c{}^2}}{2(N + Z^2_{\alpha/2})}$$

Example: Confidence Interval for Accuracy

- Consider a model that produces an accuracy of 80% when evaluated on 100 test instances:
  - N=100, acc = 0.8

$$p = \frac{2 \times N \times a_c + Z_{\alpha/2}^2 \pm Z_{\alpha/2}\sqrt{Z_{\alpha/2}^2 + 4 \times N \times a_c - 4 \times N \times a_c{}^2}}{2(N + Z_{\alpha/2}^2)}$$

1-α	Z
0.99	2.58
0.98	2.33
0.95	1.96
0.90	1.65

N	50	100	500	1000	5000
p(lower)	0.670		0.763	0.774	0.789
p(upper)	0.888		0.833	0.824	0.811

Comparing Performance of 2 Models

- Given two models, say M1 and M2, which is better?
  - M1 is tested on D1 (size=n1), found error rate = e1
  - M2 is tested on D2 (size=n2), found error rate = e2
  - Assume D1 and D2 are independent
  - The difference in the error rate is: d = e1 – e2
  - The variance of d is:

$$\sigma_t^2 = \sigma_1^2 + \sigma_2^2 \cong \frac{e1(1-e1)}{n1} + \frac{e2(1-e2)}{n2}$$

  - The confidence interval for the true difference is

$$d_t = d \pm Z_{\alpha/2}\hat{\sigma}_t$$

Example :Comparing Performance of 2 Models

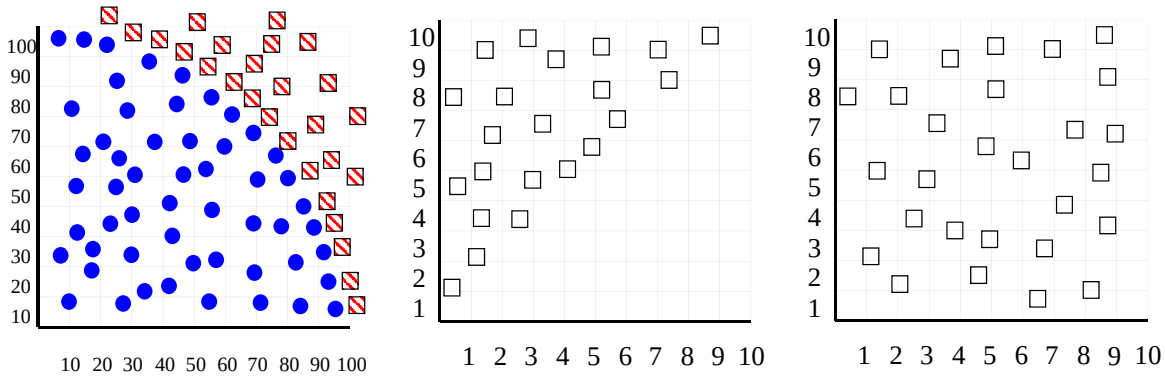
- Given: M1: n1 = 30, e1 = 0.15
  M2: n2 = 5000, e2 = 0.25
  - d = |e2 – e1| = 0.1 (2-sided test)

$$\hat{\sigma}_d = \frac{0.15(1-0.15)}{30} + \frac{0.25(1-0.25)}{5000} = 0.0043$$

- At 95% confidence level, $Z_{\alpha/2}=1.96$

$$d_t = 0.100 \pm 1.96 \times \sqrt{0.0043} = 0.100 \pm 0.128$$

The Problem Of Decision Tree

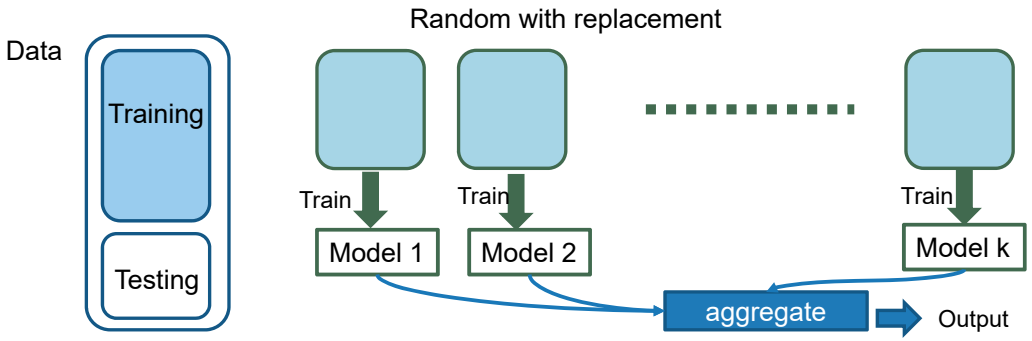


Advantages/Disadvantages of Decision Trees

- Advantages:
  - Easy to understand
  - Easy to generate rules
- Disadvantages:
  - May suffer from overfitting.
  - Classifies by rectangular partitioning (so does not handle correlated features very well).
  - Can be quite large – pruning is necessary.
  - Does not handle streaming data easily

Bagging/Bootstrap Aggregating

- To reduce the variance of an estimated prediction function
- Basic idea:
  - A *committee* of trees each casts a vote for the predicted class
  - Randomly draw datasets with replacement from the training data



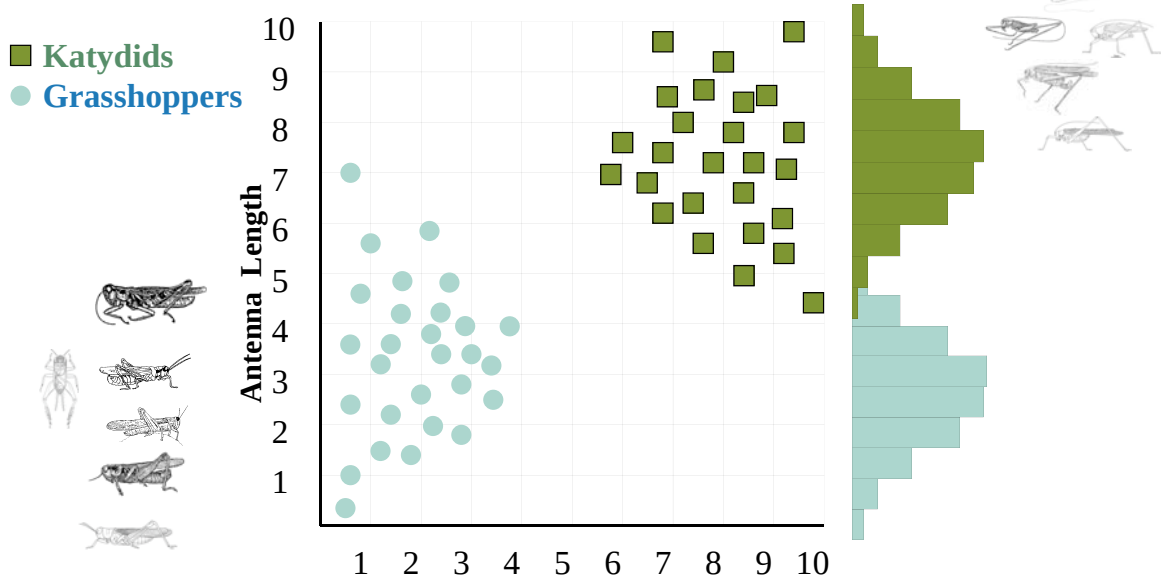
Random Forest Classifier

- An extension to bagging which uses de-correlated trees
- Create bootstrap samples from the training data
  - Each construct a decision tree
  - At each node in choosing the split feature choose only among m<M features
  - Take the majority vote

Naïve Bayes Classifier

Thomas Bayes
1702 - 1761

Example of Histogram



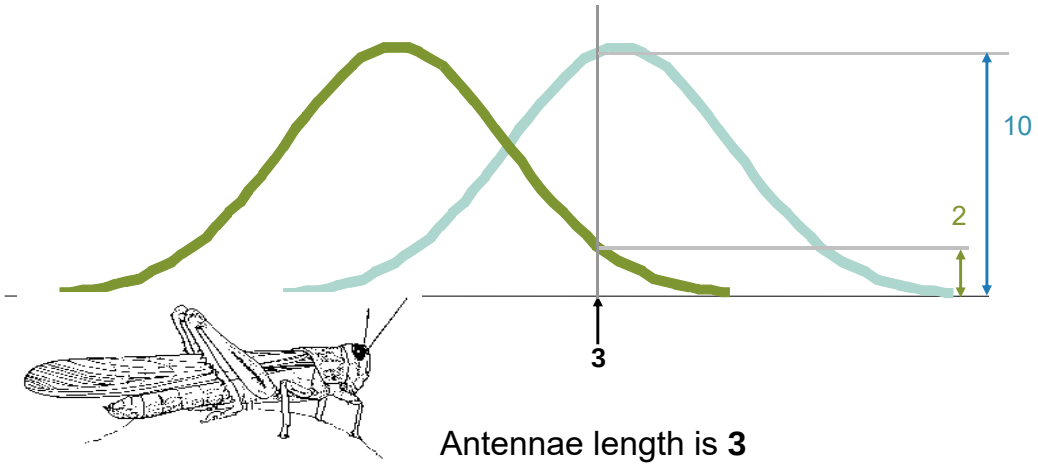
With a lot of data, we can build a histogram.
Let us just build one for "Antenna Length" for now…

Example of Histogram (Contd.)

$p(c_j \mid d)$ = probability of class c_j, given that we have observed d

P(**Grasshopper** | **3**) = 10 / (10 + 2) = 0.833

P(**Katydid** | **3**) = 2 / (10 + 2) = 0.166



Antennae length is **3**

Bayes Classifier

- A probabilistic framework for solving classification problems
- Conditional Probability:

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

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

- Bayes theorem:

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

Example of Bayes Theorem

- Given:
  - 1% of people have a certain genetic defect.
  - 90% of tests for the gene detect the defect.
  - 9.6% of the test are false positives.

X= positive test result

$P(G) = 1\%; P(\sim G) = 99\%$

$P(X|\mathrm{G}) = 90\%$

$P(\mathrm{X}|\sim\mathrm{G}) = 9.6\%$

- Question:
  - If a person gets a positive test results, what are the odds they actually have the genetic defect? $P(G|X) = ?$

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

$$P(G \mid X) = \frac{P(X \mid G)P(G)}{P(X)} = \frac{0.9 \times 0.01}{(0.9 \times 0.01 + 0.096 \times 0.99)} = 8.658\%$$

Bayesian Classifiers

- Consider each attribute and class label as random variables
- Given a record with attributes (A_1, A_2,…,A_n)
  - Goal is to predict class C
  - Specifically, we want to find the value of C that maximizes P(C| A_1, A_2,…,A_n)
- Can we estimate P(C| A_1, A_2,…,A_n) directly from data?

Bayesian Classifier Approach

- Compute the posterior probability P(C | A_1, A_2, …, A_n) for all values of C using the Bayes theorem

$$P(C \mid A_1 A_2 \ldots A_n) = \frac{P(A_1 A_2 \ldots A_n \mid C) P(C)}{P(A_1 A_2 \ldots A_n)}$$

- Choose value of C that maximizes
  P(C | A_1, A_2, …, A_n)
- Equivalent to choosing value of C that maximizes
  P(A_1, A_2, …, A_n|C) P(C)
- How to estimate P(A_1, A_2, …, A_n | C)?

Naïve Bayes Classifier

- A simplified assumption: attributes are conditionally independent and each data sample has n attributes

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$$

- No dependence relation between attributes
- By Bayes theorem,

$$P(Ci \mid X) = \frac{P(X \mid Ci) P(Ci)}{P(X)}$$

- As P(X) is constant for all classes, assign X to the class with maximum P(X|Ci)*P(Ci)

Take Home Example

Given a Test Record: $X = (\text{Refund} = \text{No}, \text{Married}, \text{Income} = 120\text{K})$

naive Bayes Classifier:

P(Refund=Yes|No) = 3/7
P(Refund=No|No) = 4/7
P(Refund=Yes|Yes) = 0
P(Refund=No|Yes) = 1
P(Marital Status=Single|No) = 2/7
P(Marital Status=Divorced|No)=1/7
P(Marital Status=Married|No) = 4/7
P(Marital Status=Single|Yes) = 2/7
P(Marital Status=Divorced|Yes)=1/7
P(Marital Status=Married|Yes) = 0

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

- P(X|Class=No) = P(Refund=No|Class=No)
  × P(Married| Class=No)
  × P(Income=120K| Class=No)
  = 4/7 × 4/7 × 0.0072 = 0.0024

- P(X|Class=Yes) = P(Refund=No| Class=Yes)
  × P(Married| Class=Yes)
  × P(Income=120K| Class=Yes)
  = 1 × 0 × 1.2 × 10^{-9} = 0

Since P(X|No)P(No) > P(X|Yes)P(Yes)

Therefore P(No|X) > P(Yes|X)

=> Class = No

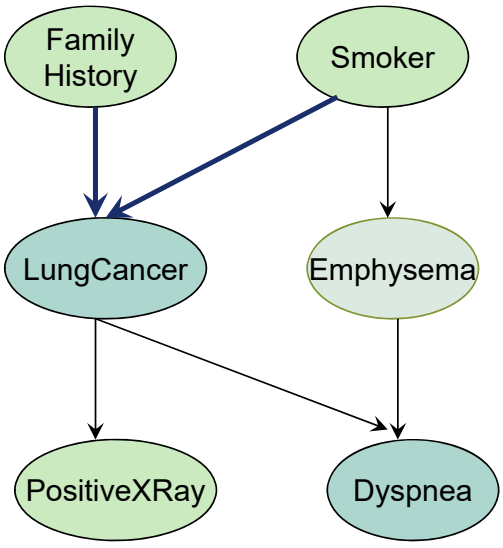
Naïve Bayesian Classifier: Comments

- Advantages :
  - Easy to implement
  - Good results obtained in most of the cases
- Disadvantages
  - Assumption: class conditional independence
  - Practically, dependencies exist among variables
    - E.g., hospitals: patients: Profile: age, family history etc
    - E.g., Symptoms: fever, cough etc., Disease: lung cancer, diabetes etc
  - Dependencies among these cannot be modeled by Naïve Bayesian Classifier
- How to deal with these dependencies?
  - Bayesian Belief Networks

Bayesian Networks

- Bayesian belief network allows a *subset* of the variables conditionally independent
- A graphical model of causal relationships
  - Represents dependency among the variables
  - Gives a specification of joint probability distribution

Bayesian Belief Network: An Example



Bayesian Belief Networks

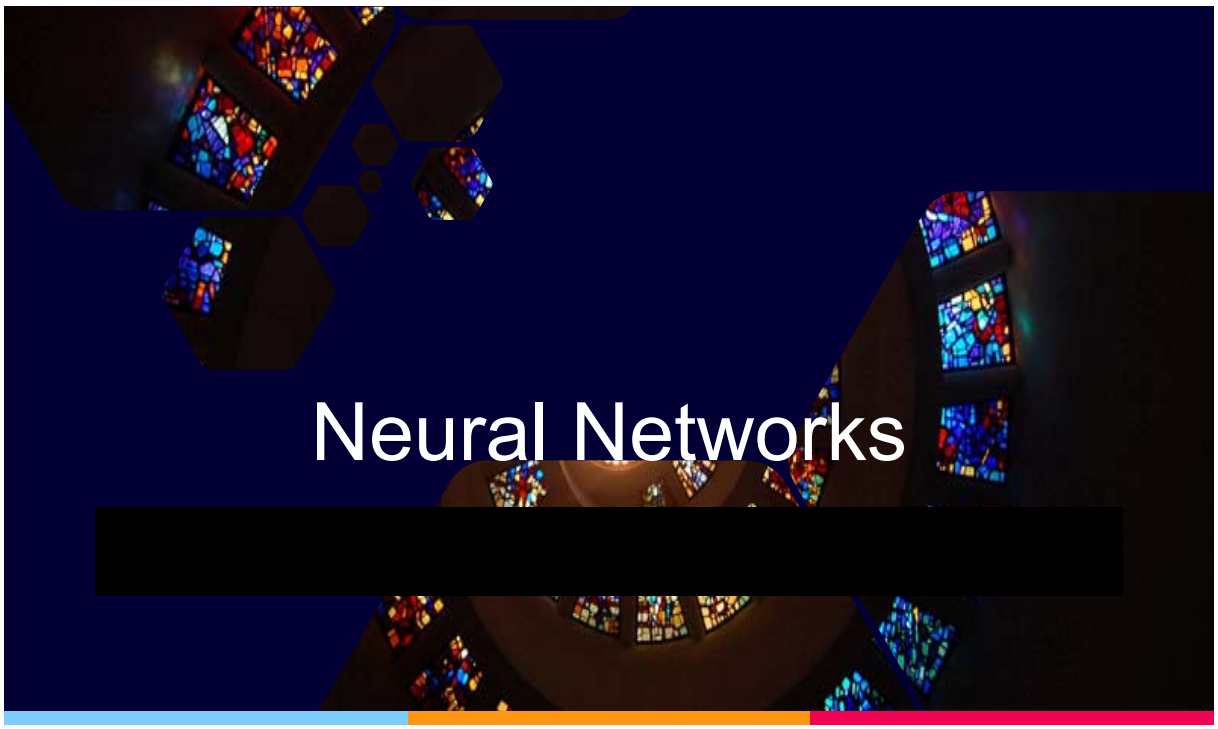
	(FH, S)	(FH, ~S)	(~FH, S)	(~FH, ~S)
LC	0.8	0.5	0.7	0.1
~LC	0.2	0.5	0.3	0.9

The conditional probability table for the variable LungCancer:
Shows the conditional probability for each possible combination of its parents

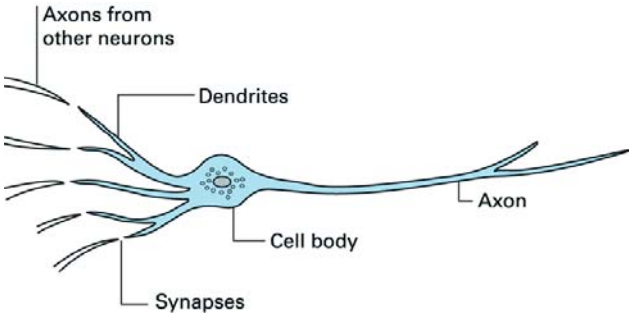
$$P(z1,..., zn) = \prod_{i=1}^{n} P(z_i \mid Parents(Z_i))$$

Constructing a Belief Network

- Select a set of variables describing the application domain
- Choose an ordering of variables
- Start with empty network
- Add variables to the network one by one based on the ordering
  - To add i-th variable X_i:
  - Determine $pa(X_i)$ of variables already in the network (X_1, …X_{i-1}) such that
    $P(X_i,X_1,…X_{i-1})=P(X_i|pa(X_i))$ (need domain knowledge)
  - Draw an arc from each variable in $pa(X_i)$ to X_i



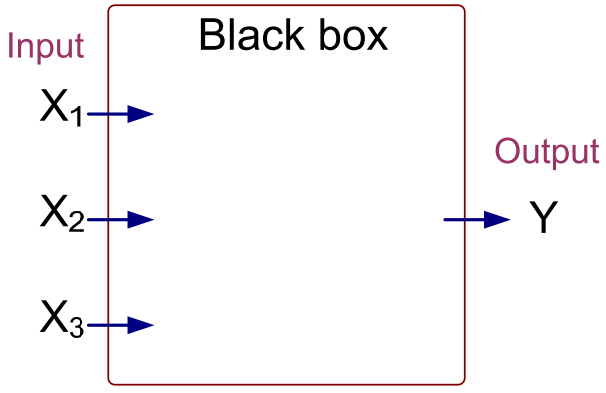
Neural Networks



- Artificial neuron
  - Each input is multiplied by a weighting factor.
  - Output is 1 if sum of weighted inputs exceeds a threshold value; 0 otherwise
- Network is programmed by adjusting weights using feedback from examples

Framework of Neural Networks

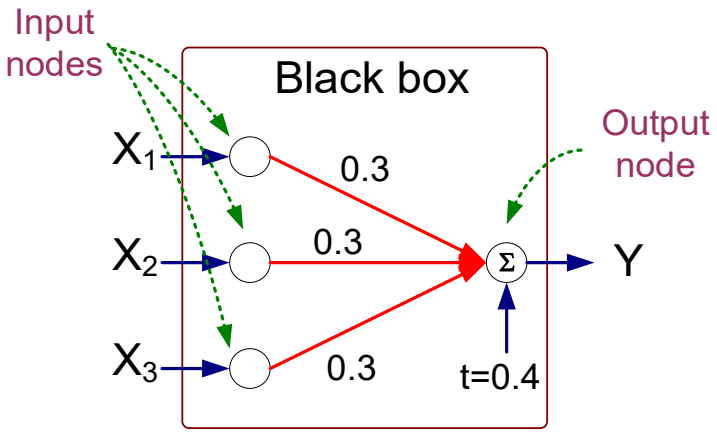
X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0



Output Y is 1 if at least two of the three inputs are equal to 1.

Training Examples

X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0

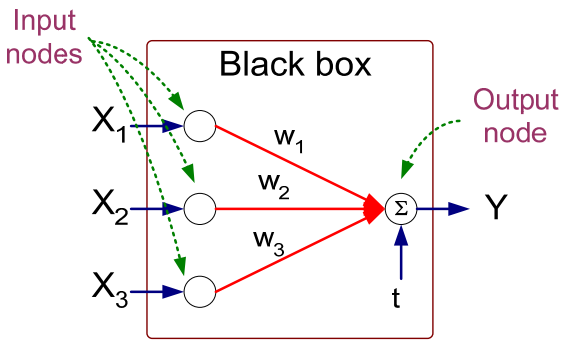


$$Y = I(0.3X_1 + 0.3X_2 + 0.3X_3 - 0.4 > 0)$$

$$\text{where } I(z) = \begin{cases} 1 & \text{if } z \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

Neural Network Model

- Model is an assembly of inter-connected nodes and weighted links
- Output node sums up each of its input value according to the weights of its links
- Compare output node against some threshold t



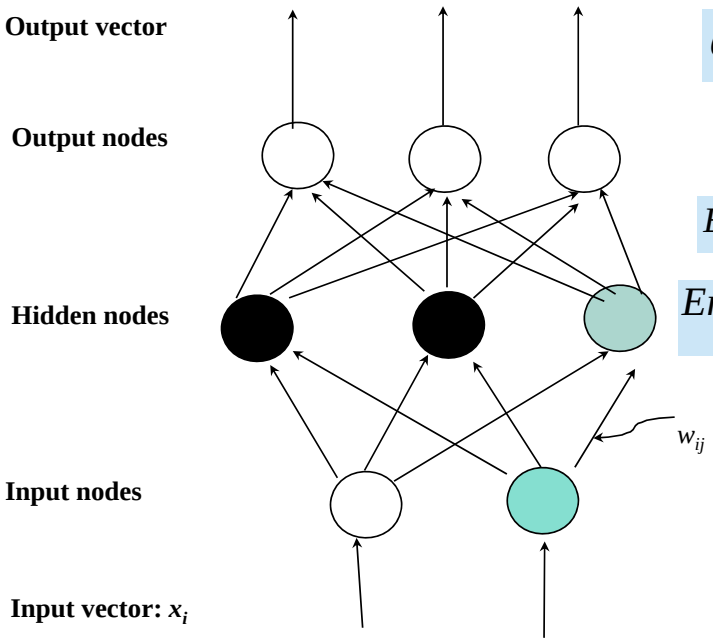
Perceptron Model

$$Y = I(\sum_i w_i X_i - t)$$

or

$$Y = sign(\sum_i w_i X_i - t)$$

General Structure



$$\theta_j = \theta_j + (l)Err_j$$

$$w_{ij} = w_{ij} + (l)Err_j O_i$$

$$Err_j = O_j(1-O_j)(T_j - O_j)$$

$$Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$$

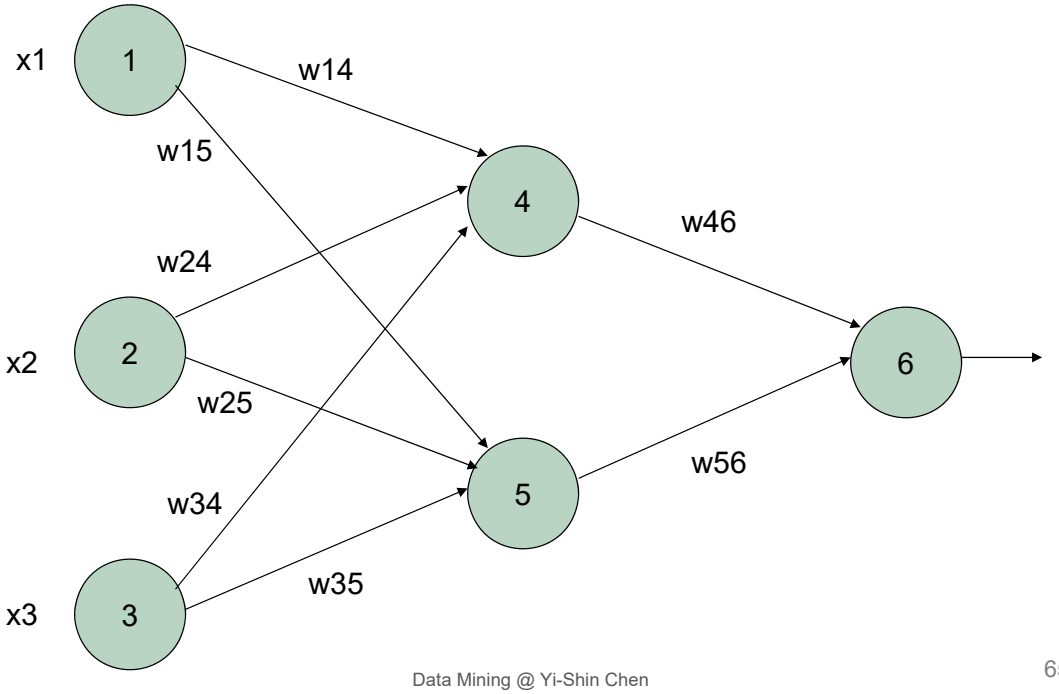
$$O_j = \frac{1}{1+e^{-I_j}}$$

$$I_j = \sum_i w_{ij} O_i + \theta_j$$

Network Training

- The ultimate objective of training
  - Obtain a set of weights that makes almost all the tuples in the training data classified correctly
- Steps
  - Initialize weights with random values
  - Feed the input tuples into the network one by one
  - For each unit
    - Compute the net input to the unit as a linear combination of all the inputs to the unit
    - Compute the output value using the activation function
    - Compute the error
    - Update the weights and the bias

Example of Neural Networks



Take Home Example

Initial input, weight, and bias values													
x1	x2	x3	w14	w15	w24	w25	w34	w35	w46	w56	Θ4	Θ5	Θ6
1	0	1	0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	-0.4	0.2	0.1

The net input and output calculations		
Unit j	Net input, I_j	Output, O_j
4	1*0.2 + 0*0.4 +1*(- 0.5) - 0.4 = -0.7	$1/(1 + e^{0.7}) = 0.332$
5	-0.3 + 0 + 0.2 + 0.2 = 0.1	$1/(1 + e^{-0.1}) = 0.525$
6	(-0.3)(0.332) – (0.2)(0.525) + 0.1 = -0.105	$1/(1 + e^{0.105}) = 0.474$

Take Home Example

$$Err_j = O_j(1-O_j)(T_j - O_j)$$

$$Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$$

Calculation of the error at each node	
Unit j	Err_j
6	(0.474)(1 – 0.474)(1 – 0.474) = **0.1311**
5	(0.525)(1 – 0.525)(**0.1311**)(-0.2) = -0.0065
4	(0.332)(1 – 0.332)(0.1311)(-0.3) = -0.0087

$$w_{ij} = w_{ij} + (l)Err_j O_i$$

$$\theta_j = \theta_j + (l)Err_j$$

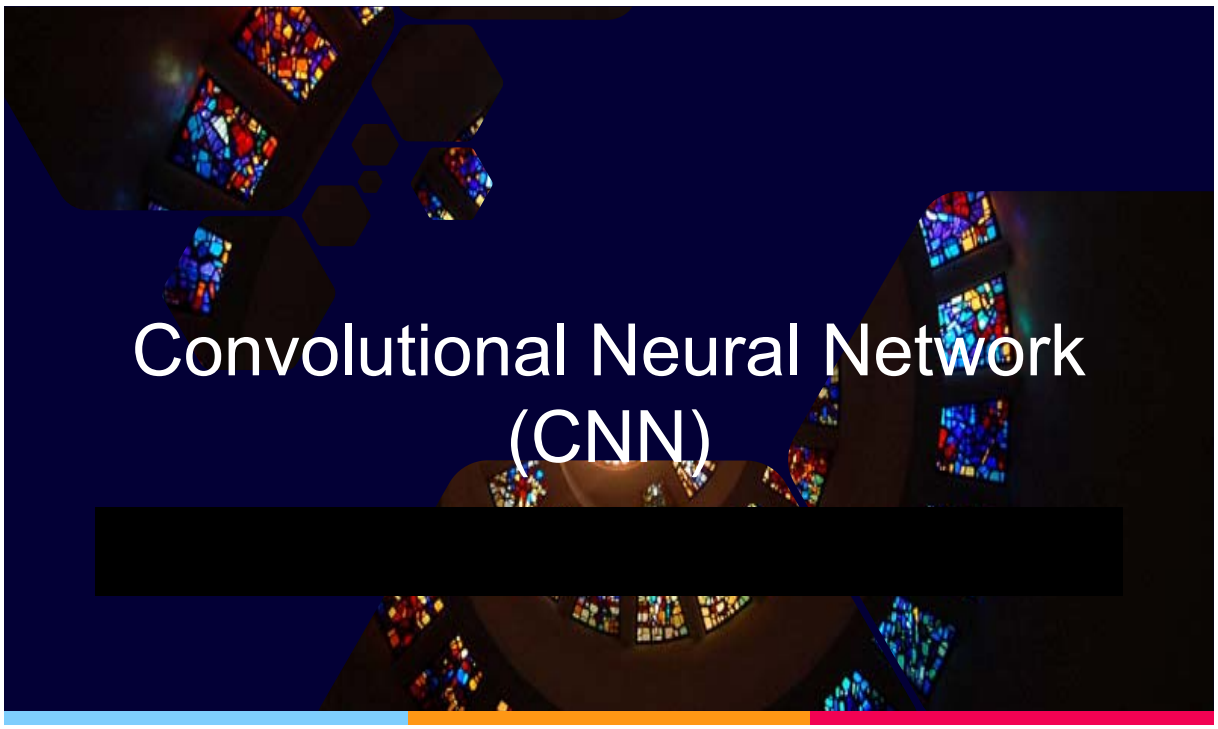
Calculation for weight and bias updating	
Weight or bias	New value
w46	-0.3+(0.9)(0.1311)(0.332)=-0.261
Θ4	-0.4+(0.9)(-0.0087)=-0.408

Network Pruning

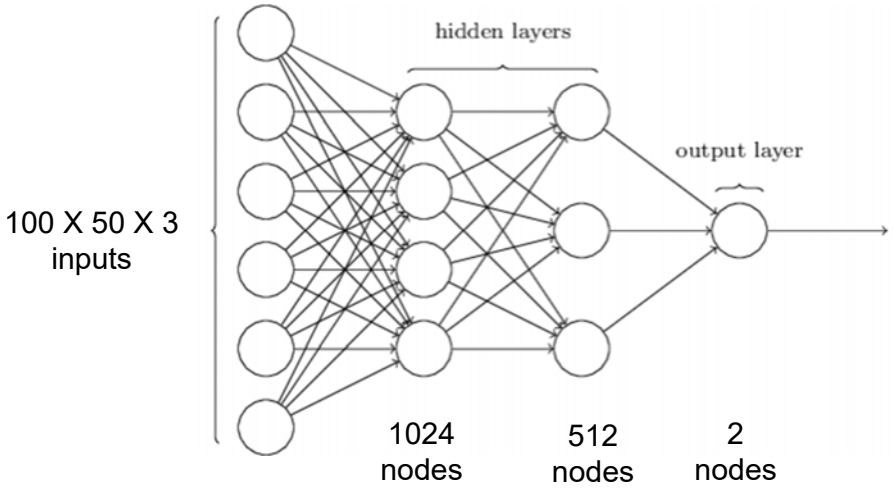
- Terminating conditions
  - All △wij were so small as to be below threshold
  - The percentage of samples misclassified is below threshold
  - A prespecified number of runs has expired
- Network pruning
  - Fully connected network will be hard to articulate
  - *N* input nodes, *h* hidden nodes and *m* output nodes lead to *h(m+N)* weights
  - Pruning: Remove some of the links without affecting classification accuracy of the network

Summary of Neural Networks

- Advantages
  - Prediction accuracy is generally high
  - Robust, works when training examples contain errors
  - Fast evaluation of the learned target function
- Criticism
  - Long training time
  - Difficult to understand the learned function (weights)
  - Not easy to incorporate domain knowledge



Size of Training Samples?

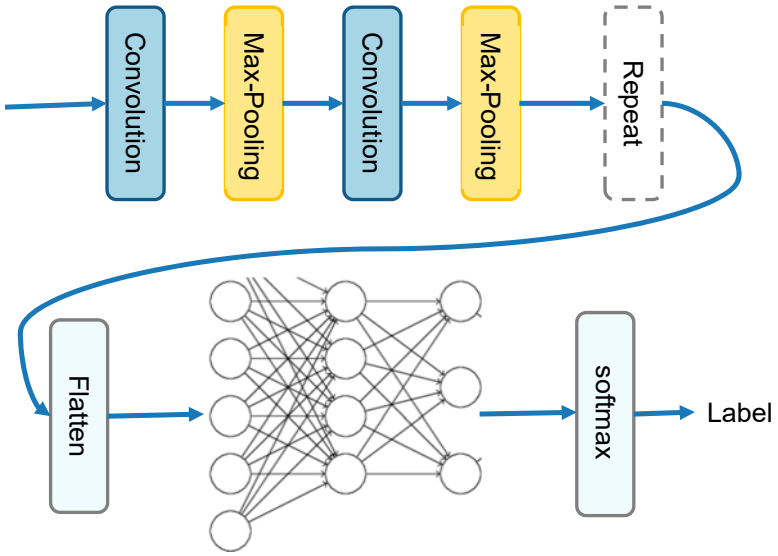


We need at least 10 * 15,728,640,000 images to well train a neural network!

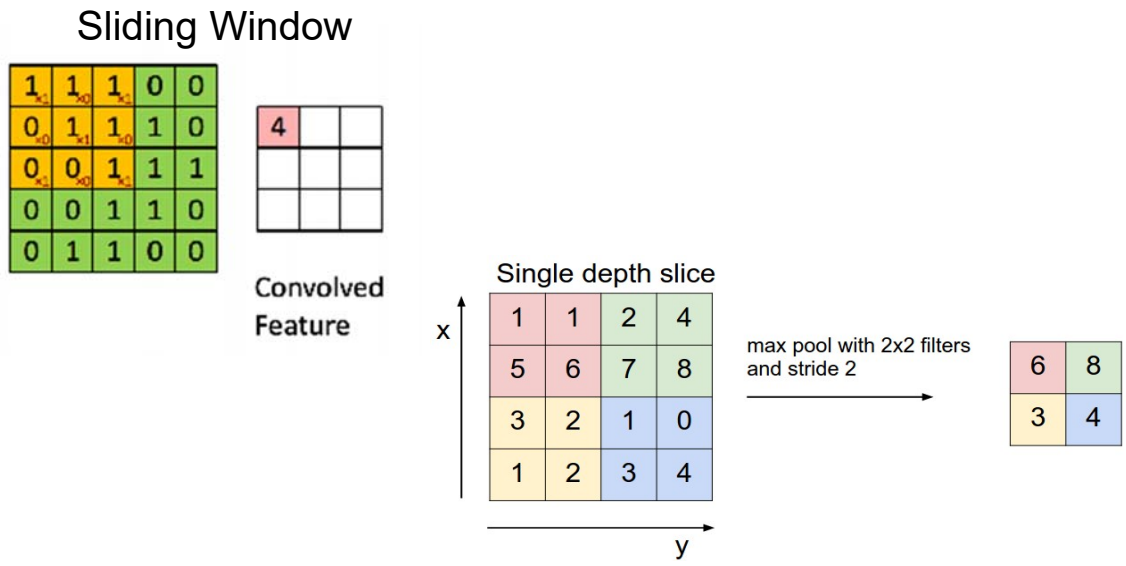
Intuition

- Convolution- Some patterns are small
  - No need to see the whole image
  - Similar pattern appears in different regions
  - Combining patterns to interpret the image
- Subsampling an image
  - Smaller image also has patterns
  - With less parameters

CNN Overview

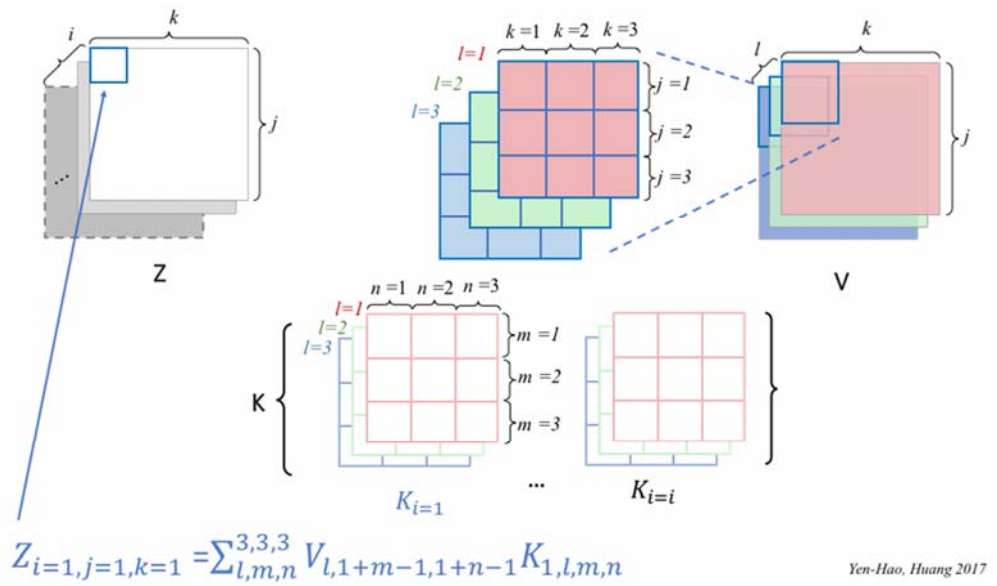


Examples



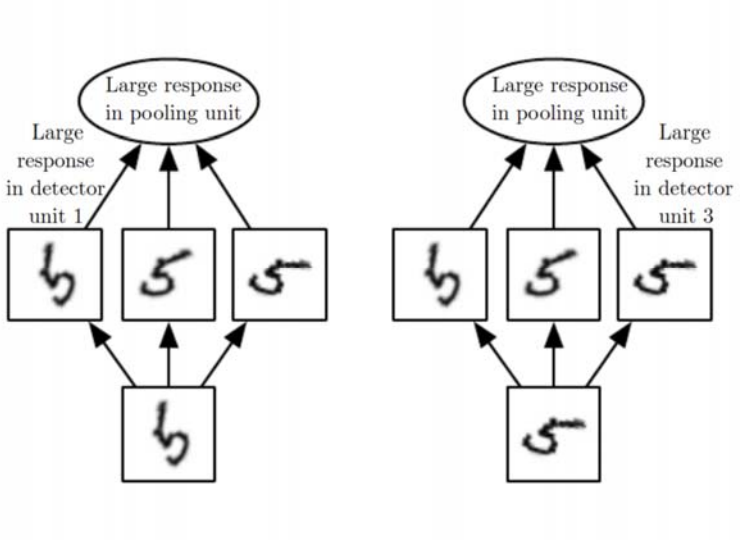
Convolution Layer

$$Z_{i,j,k} = \sum_{l,m,n} V_{l,j+m-1,k+n-1} K_{i,l,m,n}$$

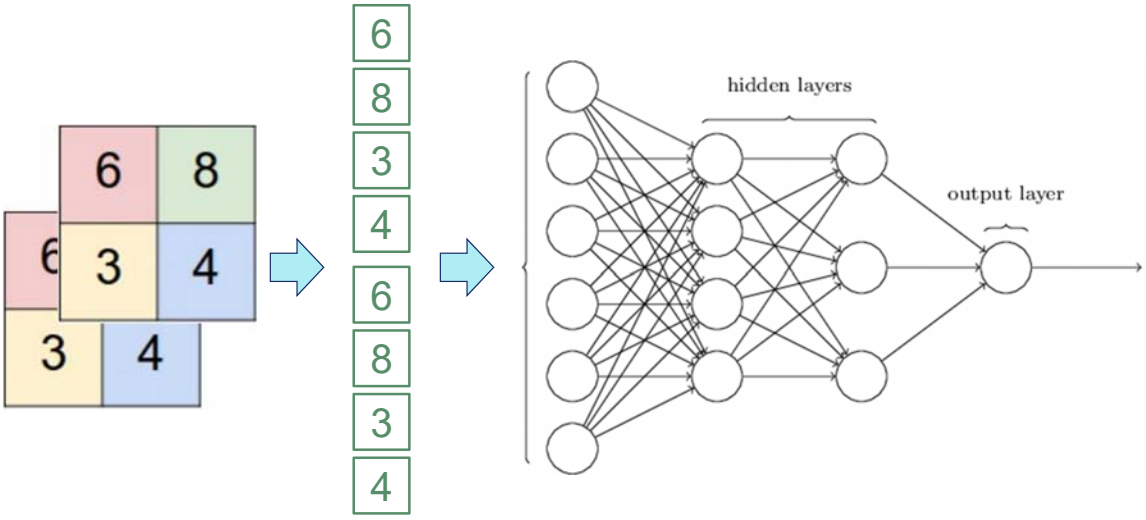


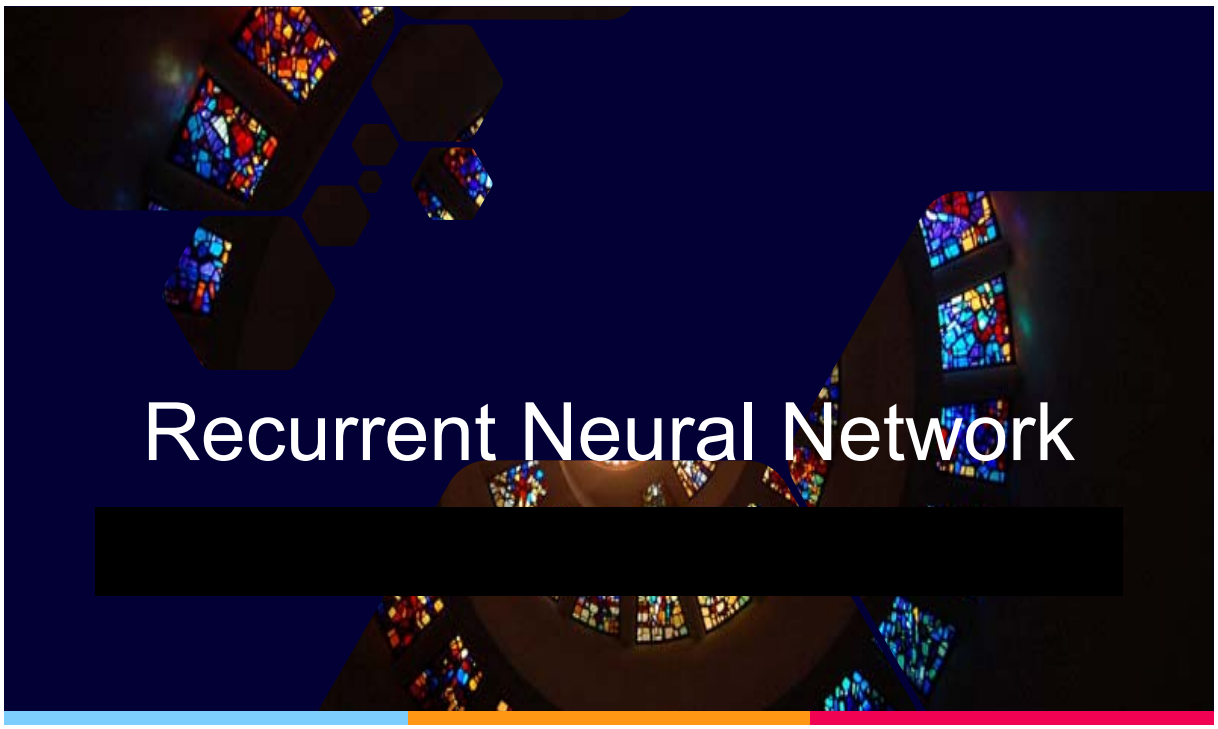
Different Angle

- Filters are able to capture patterns in different angles



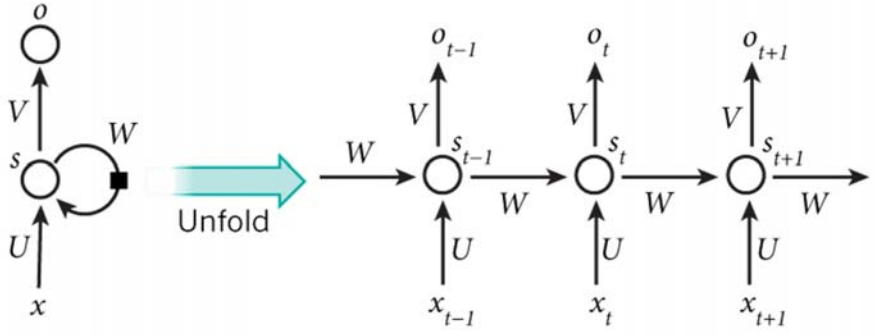
Convolution Layer - Flatten





Overview of RNN

- Design for obtaining language models
- Sequential information → all the inputs are dependent
- Memory in hidden layer



X_t: input at time step t
S_t: hidden state at time step t
O_t: output at time step t
U,V,W: parameters

Training a RNN Language Model

- Get a big corpus of text which is a sequence of words
  - Feed into RNN-LM; compute output distribution for every step t
  - i.e. predict probability dist of every word, given words so far
- Loss function on step t is cross-entropy between
  - Predicted probability distribution
  - The true next word
- Average this to get overall loss for entire training set

Advantages/Disadvantages

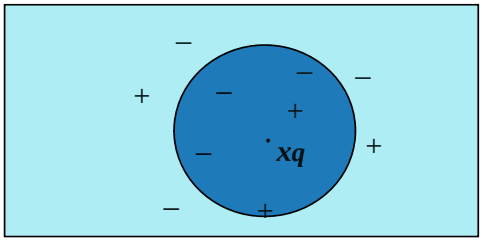
- RNN Advantages:
  - Can process any length input
  - Computation for step t can use information from many steps back
  - Model size doesn't increase for longer input
  - Same weights applied on every timestep
- RNN Disadvantages:
  - Recurrent computation is slow
  - In practice, difficult to access information from many steps back

Other Classification Methods

- K-nearest neighbor classifier
- Case-based reasoning
- Genetic algorithm
- Rough set approach
- Fuzzy set approaches

The *k*-Nearest Neighbor Algorithm

- All instances correspond to points in the n-D space.
- The nearest neighbor are defined in terms of Euclidean distance.
- The target function could be discrete- or real- valued.
- For discrete-valued, the *k*-NN returns the most common value among the k training examples nearest to x_q.



Discussion on the *k*-NN Algorithm

- Distance-weighted nearest neighbor algorithm
  - Weight the contribution of each of the k neighbors according to their distance to the query point x_q
    - Giving greater weight to closer neighbors
- Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes.
  - To overcome it, elimination of the least relevant attributes.

Genetic Algorithm

- Genetic Algorithms (GA) were introduced by Holland at 1975
- GA is an iterative search technique to find approximate answers in the search problems
- It is inspired by evolutionary biology such as inheritance, mutation, selection, and crossover

Operation

- a GA operates on a population of candidate solutions called chromosomes
  - Which is composed of numerous genes, represents an encoding of the problem
  - Associates with a fitness value evaluated by the fitness function
  - This fitness value determines the goodness and the survival ability of the chromosome

Chromosome

- Chromosomes could be:
  - Bit strings: (0101 ... 1100)
  - Real numbers: (0.02 -12.5 ... 0.0 102.3)
  - Permutations of element: (A1 A5 A9 ... A2 A16)
  - Lists of rules: (R1 R5 R6 ... R8)
  - Program elements (line256 …..)
  - other data structures/types

Algorithm

- Initializing the population and evaluating its corresponding fitness values
- While (Not terminated condition)
  - Produces newer generations
  - A portion of the chromosomes is selected according to the survival ability for reproducing offspring
    - with a probability consistent with their fitness values.
  - The offspring are generated through crossover and mutation processes