Methods for Intelligent Systems

Lecture Notes on Feature Selection

2009 - 2010

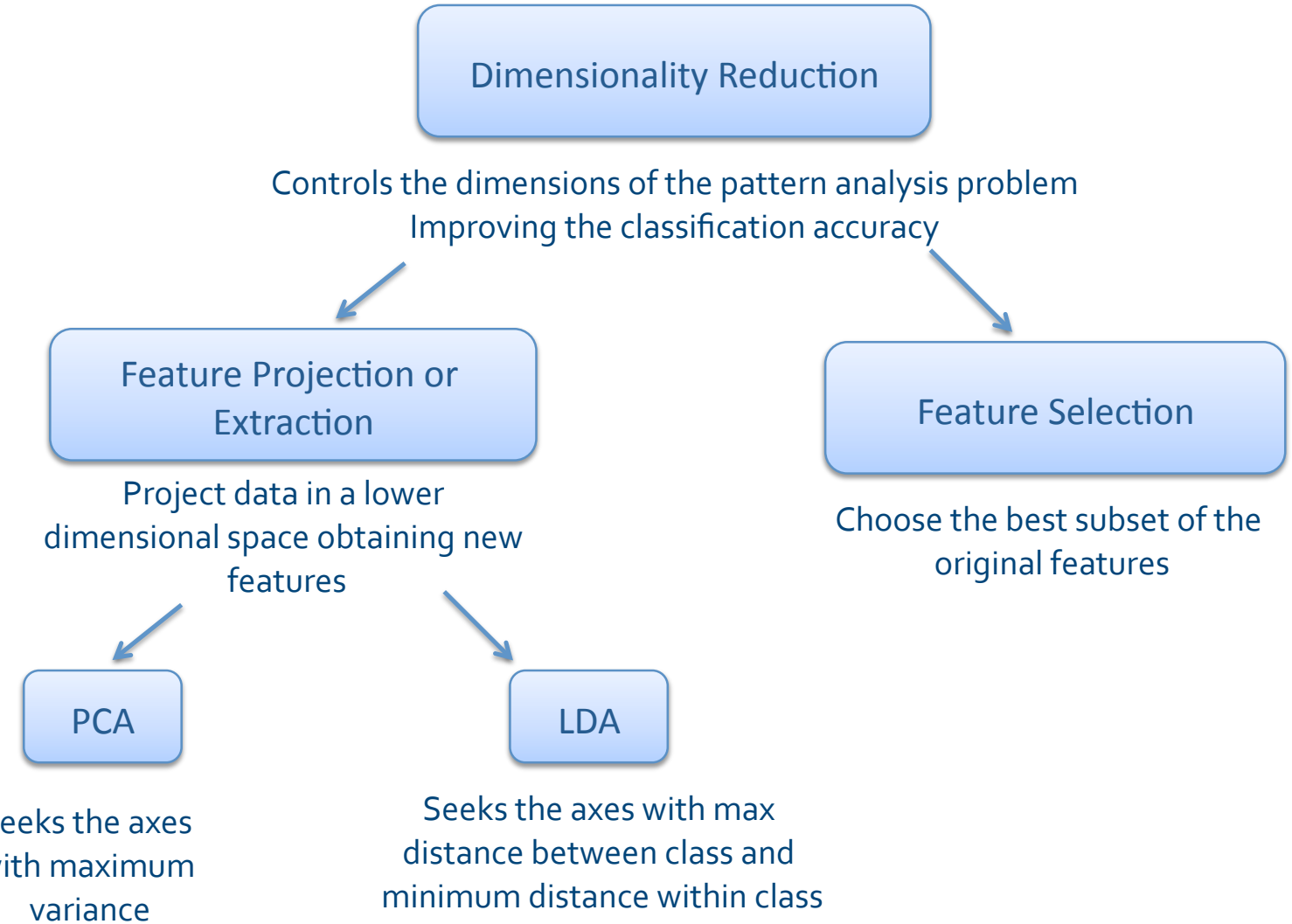
Simone Tognetti
tognetti@elet.polimi.it

Department of Electronics and Information
Politecnico di Milano

Dimensionality Reduction

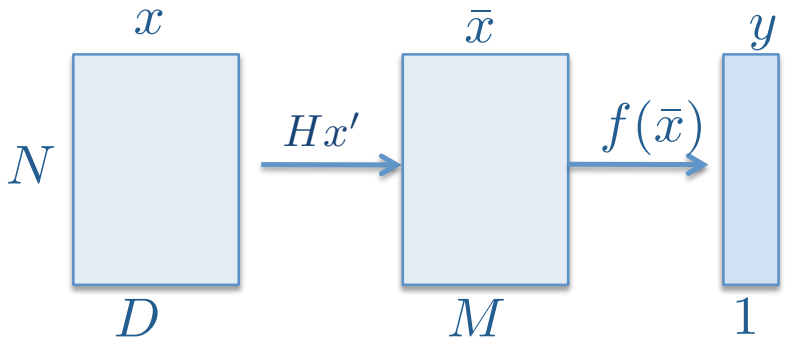
- Control of the dimensions of our classification problem
- Dimension of a classification problem
  - Number of examples (samples): N
  - Number of features: D
- No sense to reduce N
  - Data are correct by definition
  - The more data the better is the estimation of
- Not all the features are useful for the classification problems $f(x)$

Dimensionality Reduction



Feature Selection

- Refers to algorithms that select the best subset of the initial feature set
- Selected features maintain their original physical interpretation
  - useful to understand the physical process that generates patterns
- It leads to saving in measurement cost
- Feature Selection is also called "*Feature Subset Selection*"
- **Goal:** Given a feature set $X = \{x_1, x_2, \ldots, x_D\}$ find a transformation H that maps X to a subset $Y = \{x_{i1}, x_{i2}, \ldots, x_{iM}\}$ of X with M<D, that optimizes an objective function J(H)



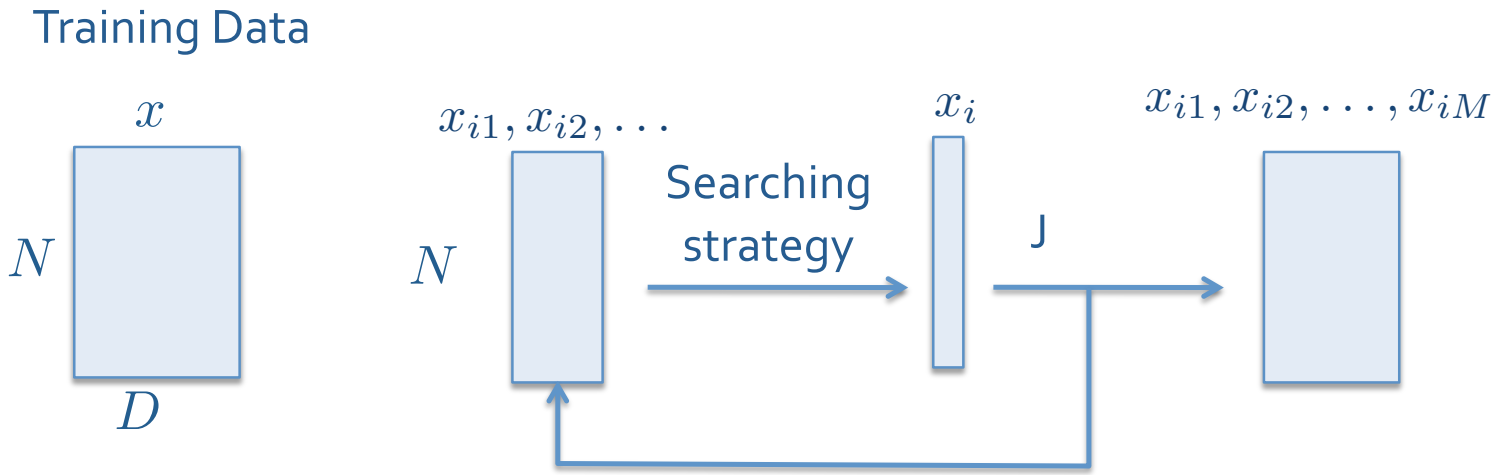
$$\bar{x} = \begin{bmatrix} h_{11} & 0 & \ldots & 0 \\ 0 & h_{22} & \ldots & 0 \\ \ldots & & & \\ 0 & & \ldots & h_{DD} \end{bmatrix} x'$$

$h_{ii} \in \{0, 1\}$

- 0- not selected feat.
- 1- selected feat.

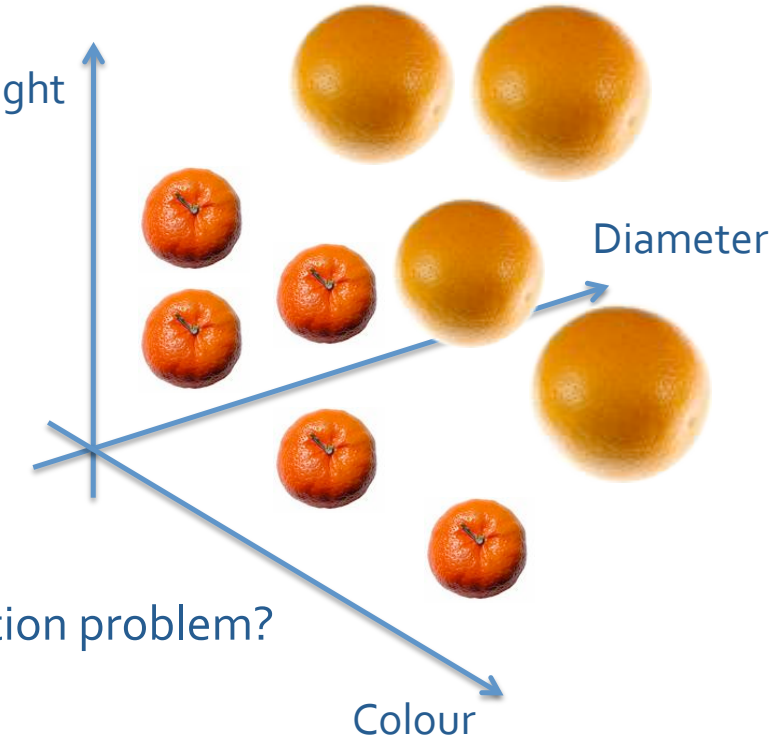
Feature Selection(2)

- Solving an optimization problem
- Optimization of J(H)
  - Choice of a measure of goodness J
    - Classification accuracy , Inter class distance, Entropy
- Find subset of feature
  - Searching strategies
    - Sequential , Casual, Heuristics, Complete, Partial



Feature Selection: Example

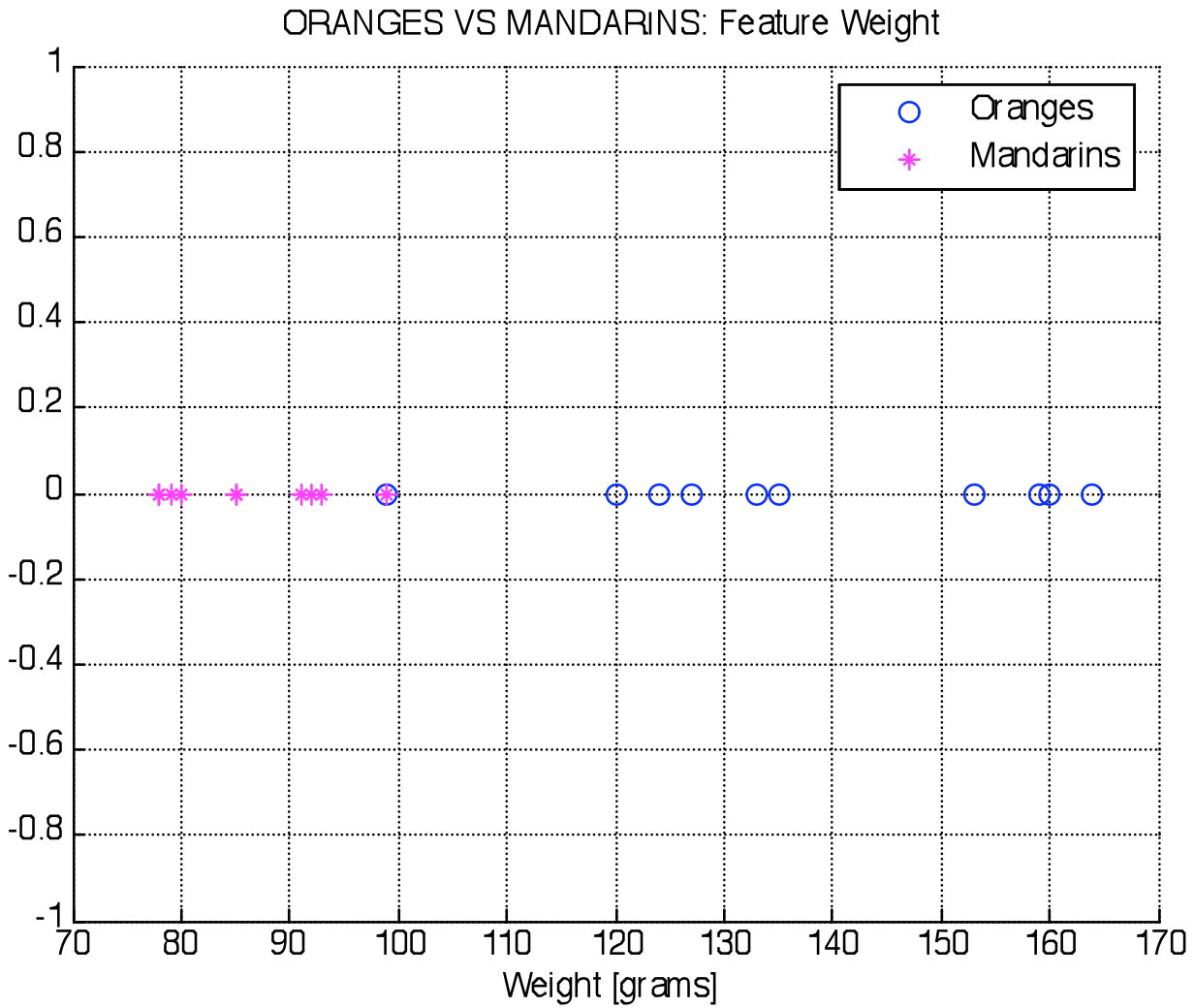
- Recognize oranges from mandarins…
- Features D = 3
  - Weight
  - Color intensity
  - Diameter
- Dataset N = 10 samples
- Classes = {Orange (= 1), Mandarin(=2)}
- Which feature should we use for the classification problem?



Feature Selection: Example

- 1-D classification problem: Weight

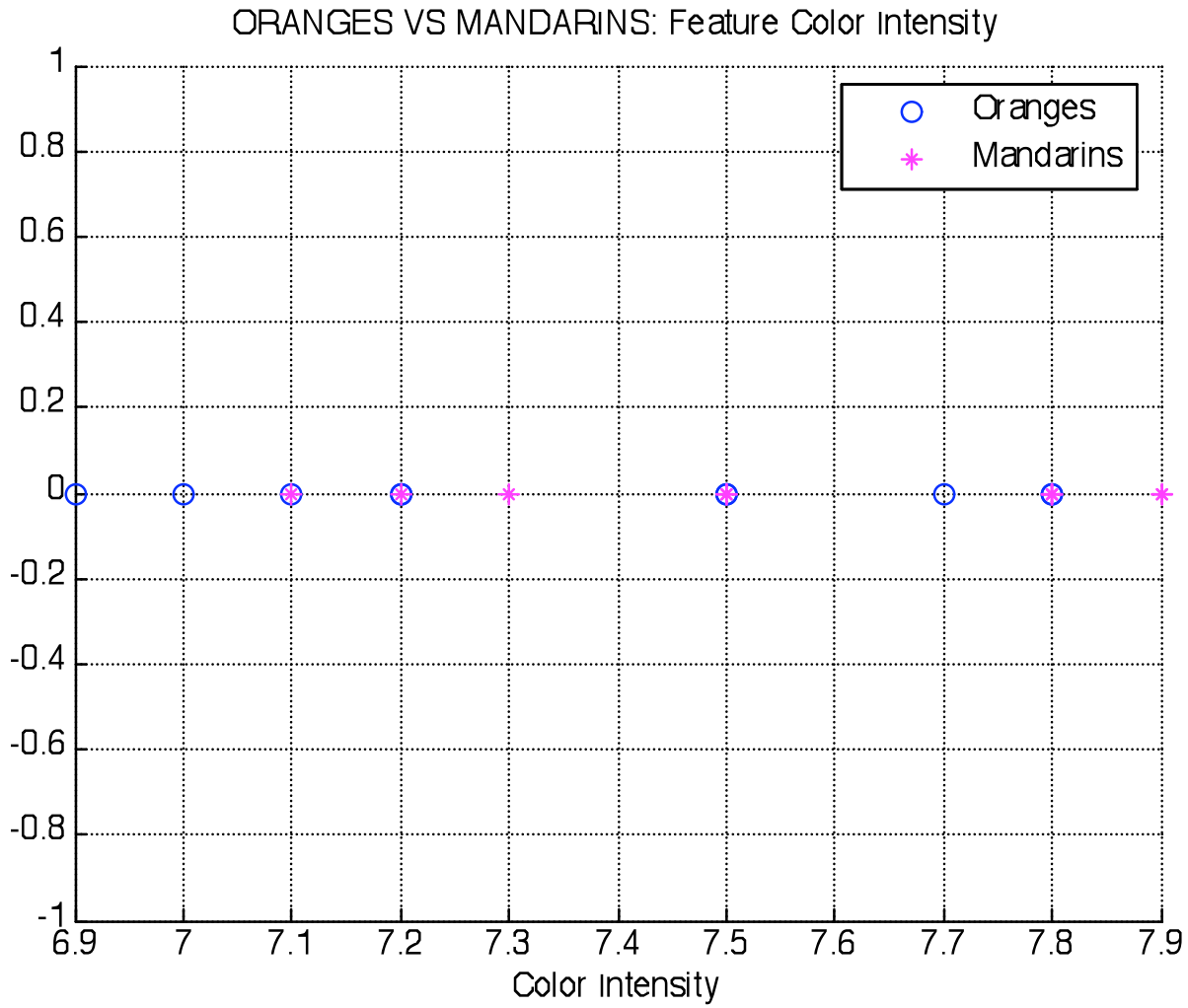
$$\begin{bmatrix} 120 & 1 \\ 135 & 1 \\ 99 & 1 \\ 160 & 1 \\ 159 & 1 \\ 153 & 1 \\ 127 & 1 \\ 164 & 1 \\ 133 & 1 \\ 124 & 1 \\ 80 & 2 \\ 85 & 2 \\ 99 & 2 \\ 78 & 2 \\ 93 & 2 \\ 85 & 2 \\ 92 & 2 \\ 91 & 2 \\ 79 & 2 \\ 78 & 2 \end{bmatrix}$$



Feature Selection: Example

- 1-D classification problem: colour intensity

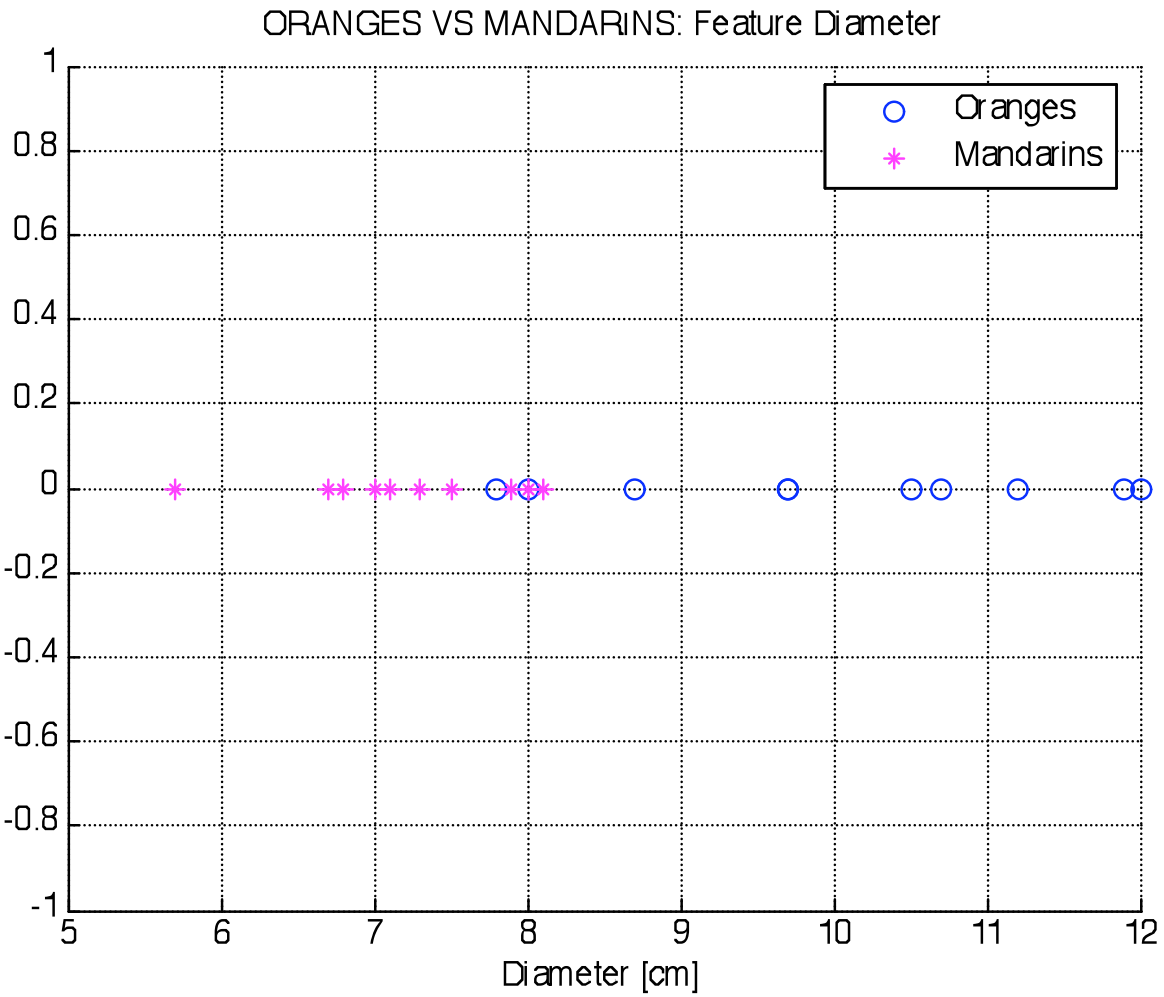
$$\begin{bmatrix} 0.7 & 1 \\ 0.75 & 1 \\ 0.71 & 1 \\ 0.69 & 1 \\ 0.78 & 1 \\ 0.72 & 1 \\ 0.75 & 1 \\ 0.77 & 1 \\ 0.78 & 1 \\ 0.72 & 1 \\ 0.78 & 2 \\ 0.75 & 2 \\ 0.79 & 2 \\ 0.78 & 2 \\ 0.73 & 2 \\ 0.75 & 2 \\ 0.72 & 2 \\ 0.71 & 2 \\ 0.79 & 2 \\ 0.78 & 2 \end{bmatrix}$$



Feature Selection: Example

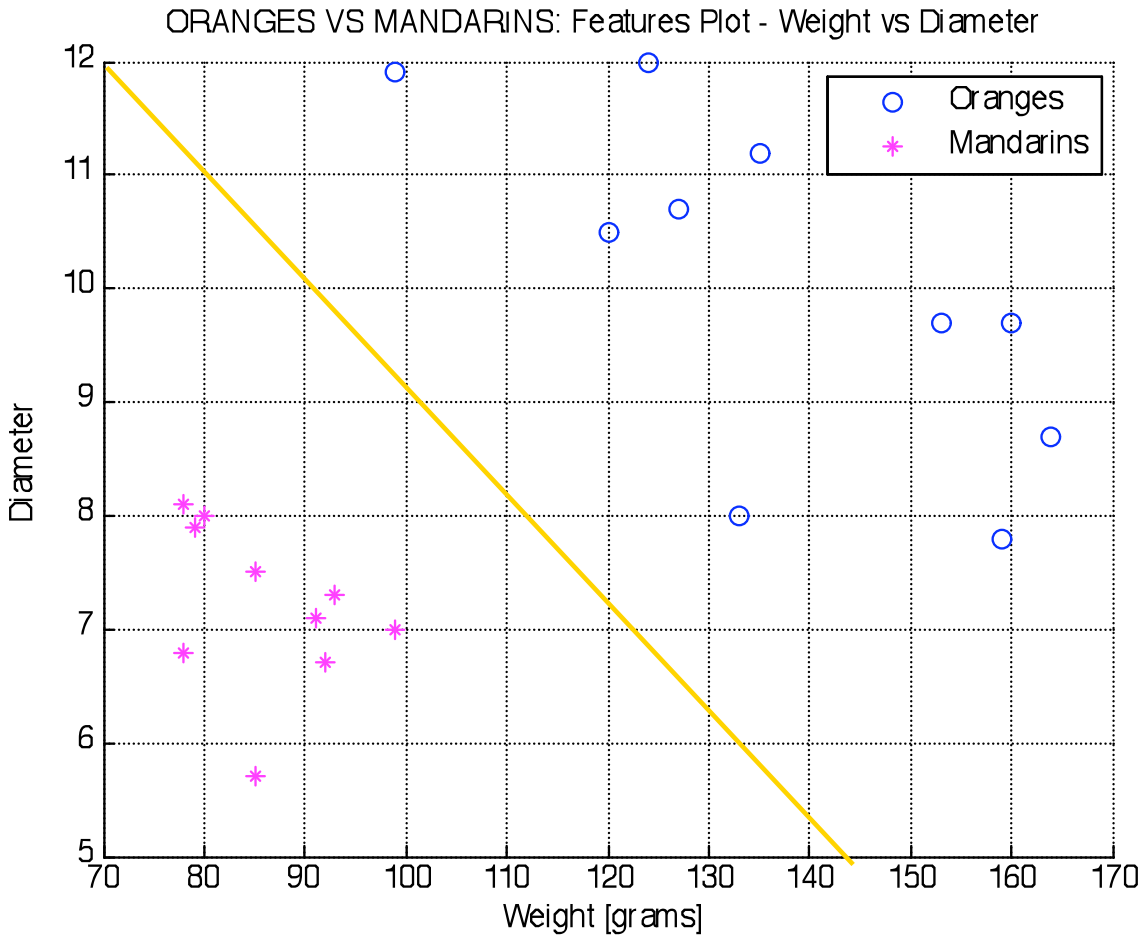
- 1-D classification problem: diameter

$$\begin{bmatrix} 10.5 & 1 \\ 11.2 & 1 \\ 11.9 & 1 \\ 9.7 & 1 \\ 7.8 & 1 \\ 9.7 & 1 \\ 10.7 & 1 \\ 8.7 & 1 \\ 8 & 1 \\ 12 & 1 \\ 8 & 2 \\ 7.5 & 2 \\ 7 & 2 \\ 6.8 & 2 \\ 7.3 & 2 \\ 5.7 & 2 \\ 6.7 & 2 \\ 7.1 & 2 \\ 7.9 & 2 \\ 8.1 & 2 \end{bmatrix}$$



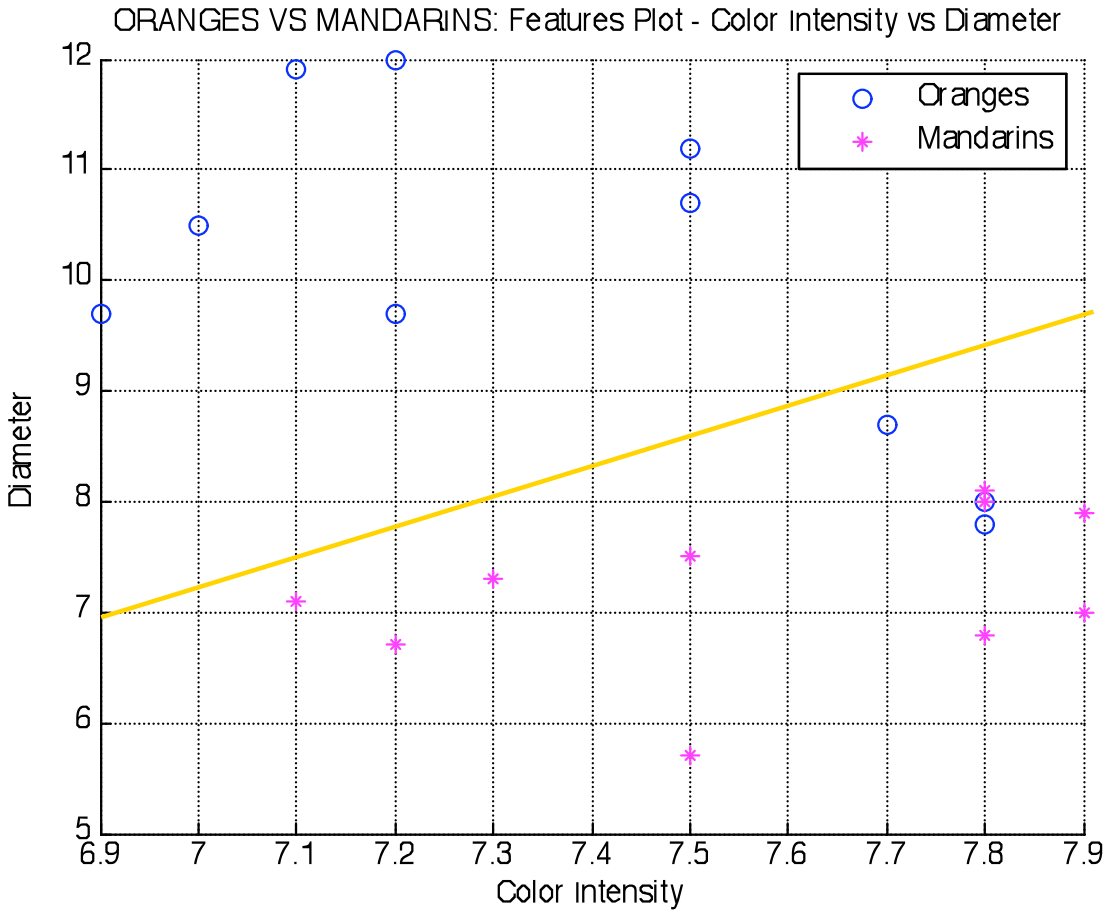
Feature Selection: Example

- 2-D classification problem: Diameter, Weight
  - Linear classifier: line (2 parameters)



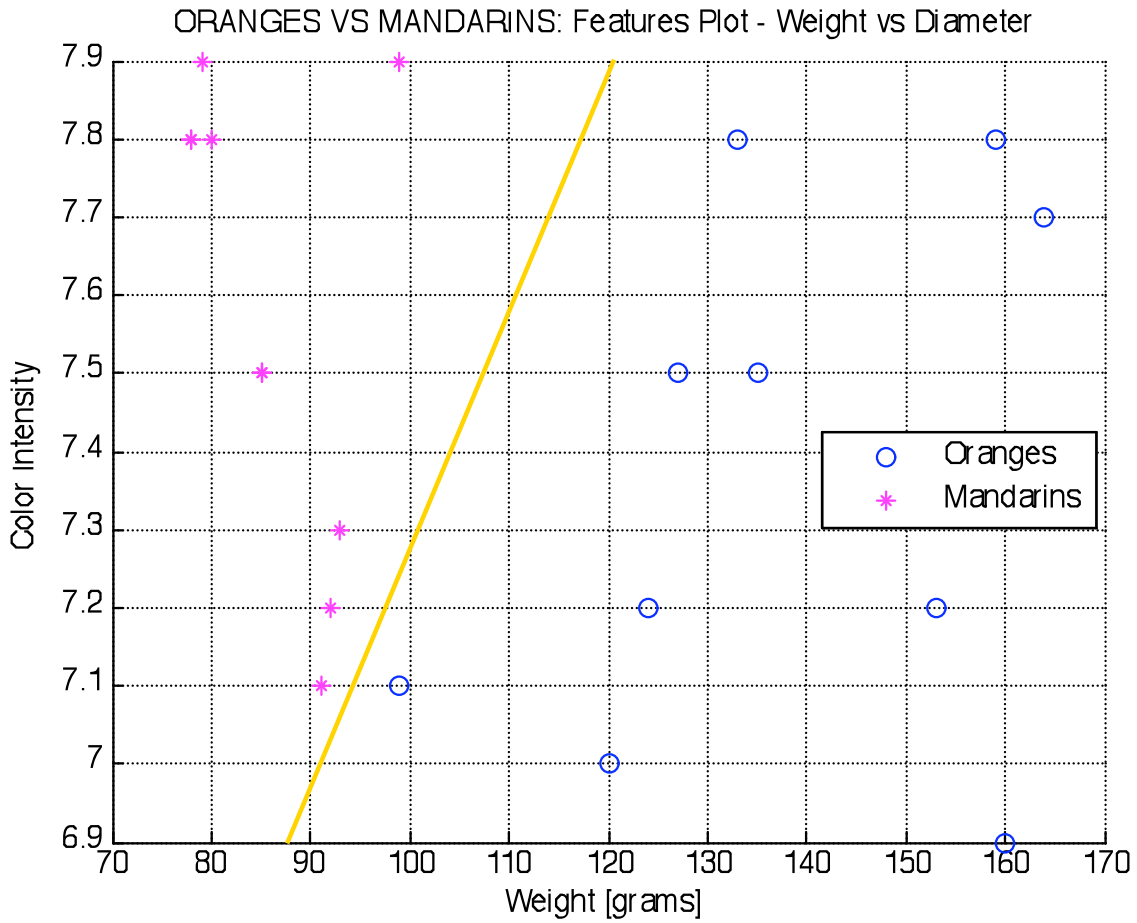
Feature Selection: Example

- 2-D classification problem: Diameter, Color Intensity
  - Linear classifier: line (2 parameters)



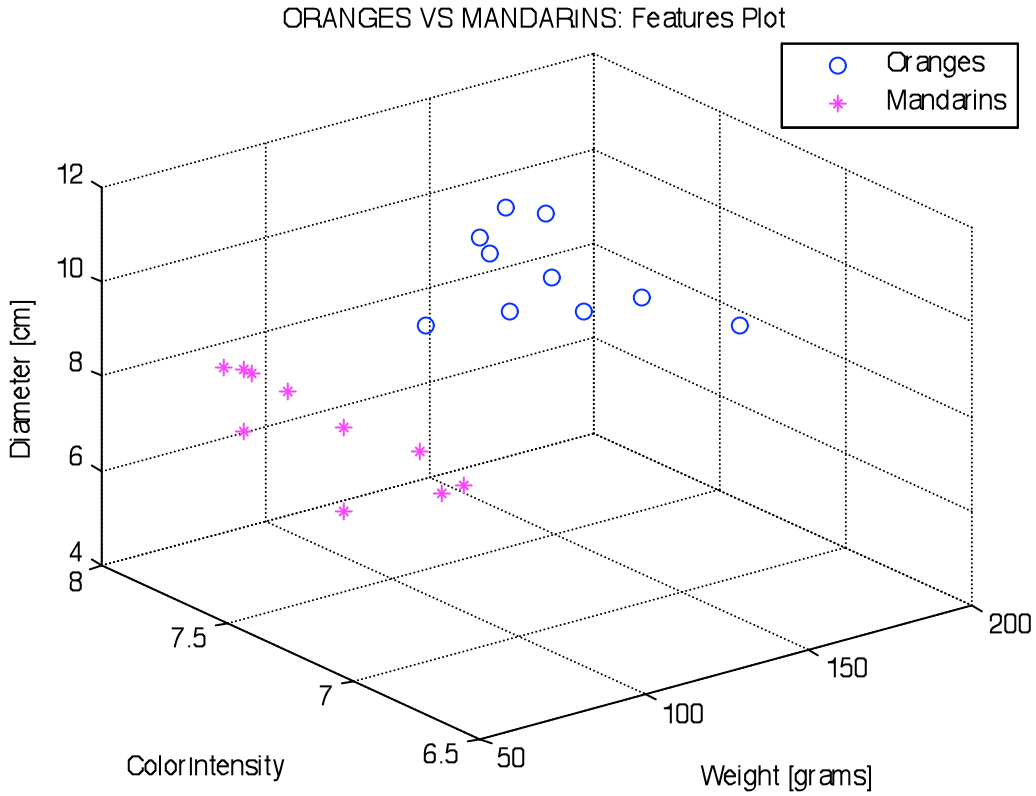
Feature Selection: Example

- 2-D classification problem: Color Intensity, Weight
  - Linear classifier: line (2 parameters)



Feature Selection: Example

- 3-D classification problem



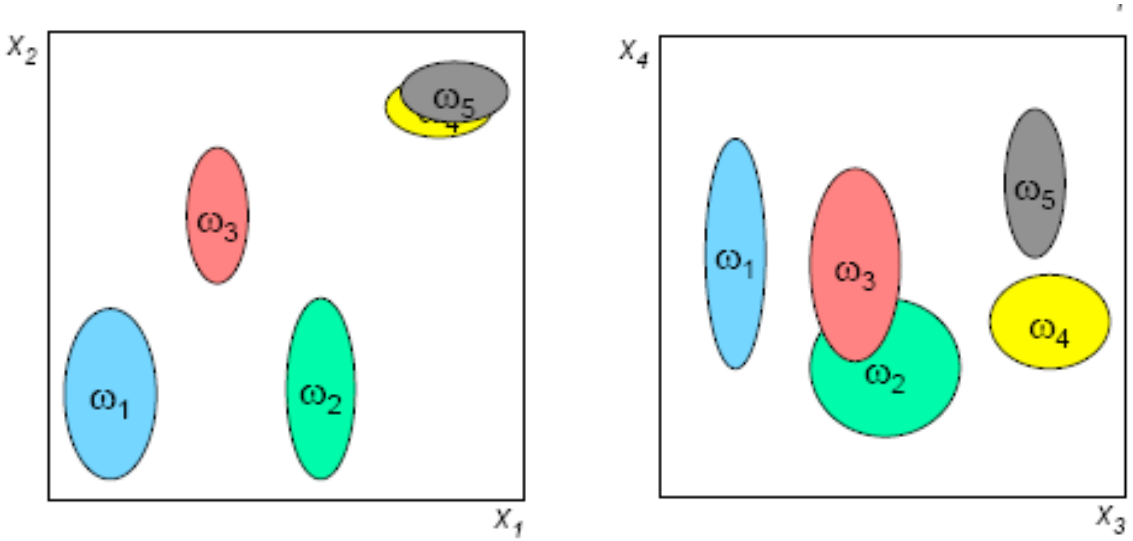
- How can features be evaluated?
- Which features should be considered?

Best Firsts: a naïve approach

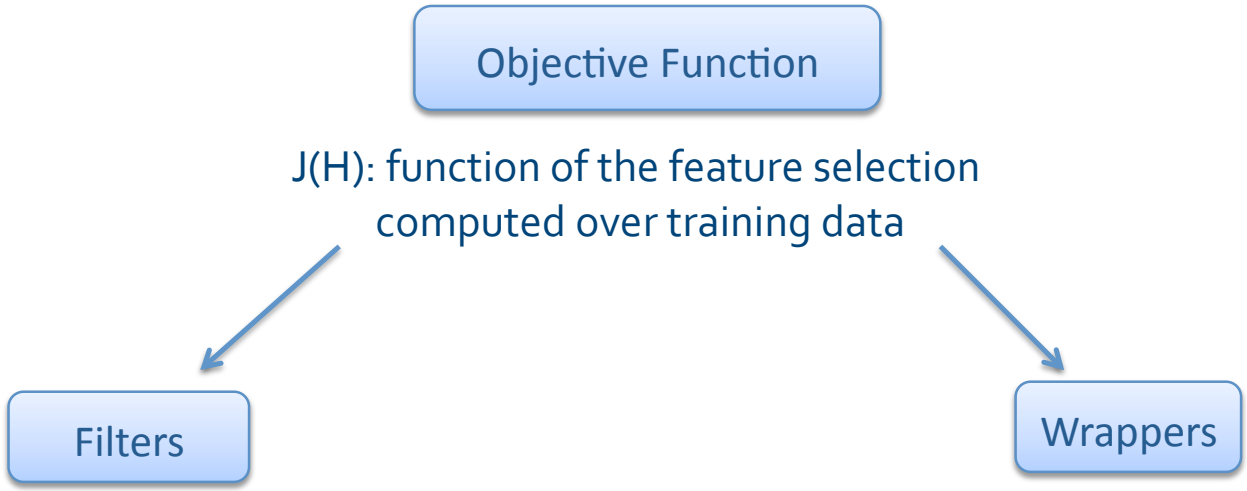
- Uses the M-best individual features
- Orange vs Mandarin Example:
  1. Weight (very good)
  2. Diameter (quite good)
  3. Colour (very bad)
- If M=2 'Weight' and 'Diameter' will be the winner
- Characteristics
  - \+ Fast
  - – Greedy approach
  - – Do not consider complementary information

Best Firsts: example

- 4-D classification problem – 5 classes
- Apply feature selection M=2
- Features ranking
  1. X_1 discriminate all clusters (except for ω_4 and ω_5)
  2. X_3 separates groups $\{\omega_1\}$, $\{\omega_2 - \omega_3\}$, $\{\omega_4 - \omega_5\}$)
  3. X_2 separates groups $\{\omega_1 - \omega_2\}$, $\{\omega_3\}$, $\{\omega_4 - \omega_5\}$)
  4. X_4 separates groups ω_4 and ω_5
- Best first selection : X_1 ,X_3 not allowing discrimination between ω_4 and ω_5
- Best selection x_1 and x_4



Objective Function



feature subsets evaluated by their information content

- interclass distance
- statistical dependence
- information theoretic measures

feature subsets evaluated by their predictive accuracy (correct classification rate) combined with

- Cross-validation
- Statistical resampling

Filter Objective Function

- Distance between classes
  - Euclidean
  - Mahalanobis
  - Determinant of $S_W^{-1}S_B$
- Correlation and information theoretic measures
  - Linear measure: Correlation Coefficient
    - Measures the correlation between selected features and class

$$J(H_M) = \frac{\sum_{i=1}^{M} \rho_{ic}}{\sum_{i=1}^{M} \sum_{j=i+1}^{M} \rho_{ij}}$$

ρ_{ic} correlation coefficient between x_i and class

ρ_{ij} Correlation between features

  - Non Linear measure: Mutual Information
    - it measures how much knowing the features reduces our uncertainty about the classe.

Filters vs Wrappers

Filters

- Fast Execution
  - non-iterative computation on the dataset
- Generality
  - Evaluates intrinsic properties of the data
  - Good solution for a large family of classifiers
- Tendency to select large subset
  - Since objective functions are generally monotonic
  - the user to select a cutoff on the number of features

Wrapper

- Accuracy
  - better recognition rates
  - they are tuned to the specific interactions between features and class
- Generalization
  - cross validation they can avoid overfitting
- Slow Execution
  - train a classifier for each feature subset
- Problem dependent

Search Strategies

- Exponential Algorithm
  - Their complexity increase exponentially with D
    - Exhaustive search
    - Branch & Bound
    - Approximate Monoticity with Branch & Bound
- Sequential Algorithms
  - Add and remove features sequentially
    - Sequential Forward Selection
    - Sequential Backward Selection
    - Bidirectional Selection
    - Plus-L Minus-R Selection
    - Sequential Floating Selection
- Randomized Algorithms
  - Incorporate randomness to escape local minima
    - Random Generation plus Sequential Selection
    - Simulated Annealing
    - Genetic Algorithms

Exponential Search: Exhaustive search

- Consider all possible subset of feature set $2^D - 1$ (empty set excluded!)
  - Exponential complexity $\Theta(2^D)$
- This number of combination is unfeasible, even for moderate value of M and N
- Guaranteed to find the optimal subset
- Oranges vs mandarins example: 3 features

$$\{x_1\}\{x_2\}\{x_3\}\{x_1, x_2\}\{x_1, x_3\}\{x_2, x_3\}\{x_1, x_2, x_3\}$$

- we obtain 3 subsets of 1 feature, 3 subsets of 2 features and 1 subset of 3 features
- With a more realistic number of features N=20
  - 2^{20} = 21048576 possible subsets to be evaluated
  - Considering 0.1 sec for each evaluation
  - 2^{20} * 0.1/3600 = 29.1271 hours to finish

Exponential Search: Branch & Bound

- Branch & Bound search method
  - Probem: select d features from the original set D w.r.t J criterion
  - Monoticity assumption

$$\bar{X}_j = X \setminus \{x_1, x_2, \ldots, x_j\}$$

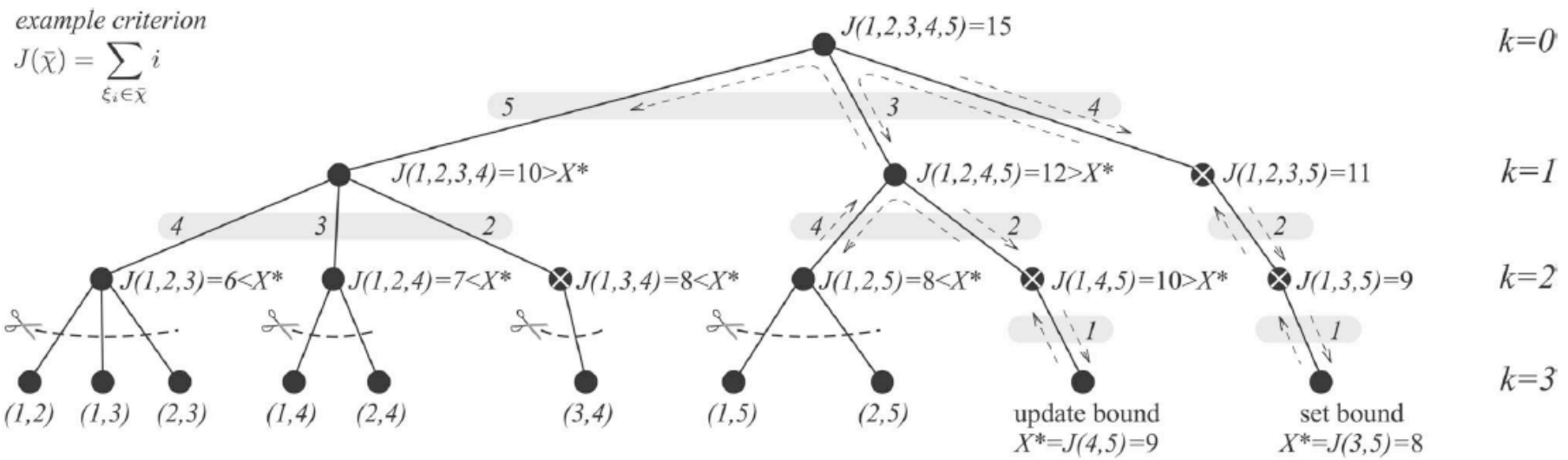
Subset of features obtained by removing j features from X

- For $\bar{X}_1, \bar{X}_2, \ldots, \bar{X}_j$ for which $\bar{X}_1 \supset \bar{X}_2 \supset \cdots \supset \bar{X}_j$

$$J(\bar{X}_1) \geq J(\bar{X}_2) \geq \cdots \geq J(\bar{X}_j)$$

- Branch & Bound algorithm
  - Starts from the full set
  - Removes features using a depth-first strategy
  - Sub selection with lower objective function are not explored
  - their children will not contain a better solution

Branch & Bound Example

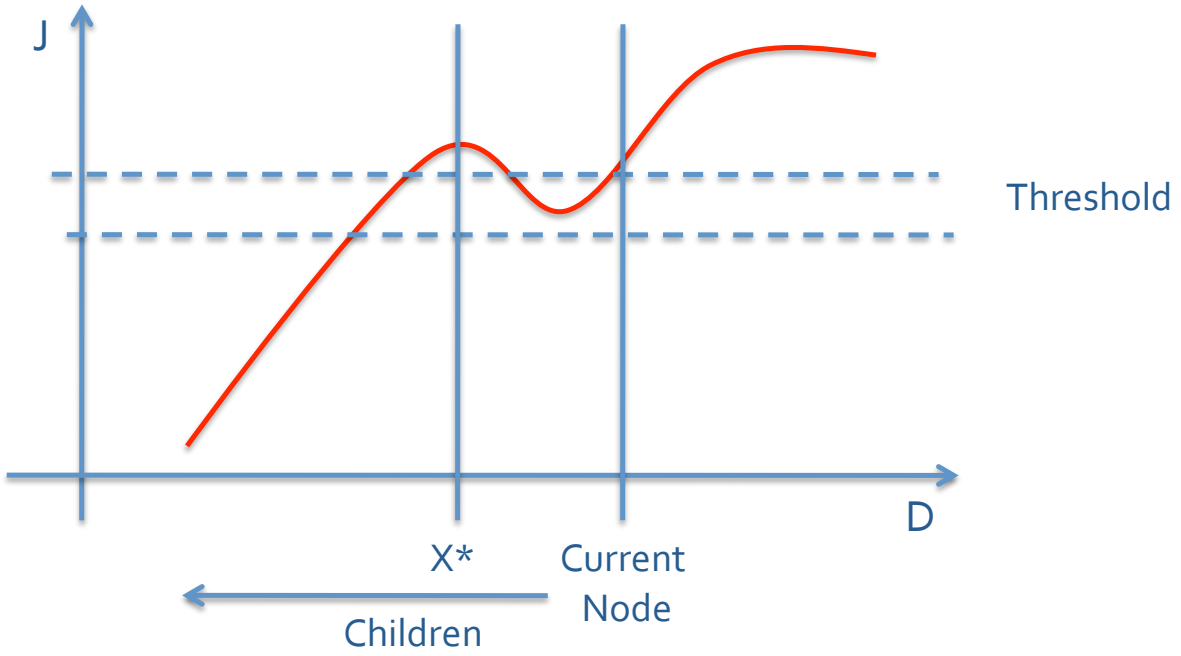


Reference:

Fast branch & bound algorithms for optimal feature selection, P. Somol, P. Pudil and J. Kittler. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 6. no.7, pp. 900-912, 2004

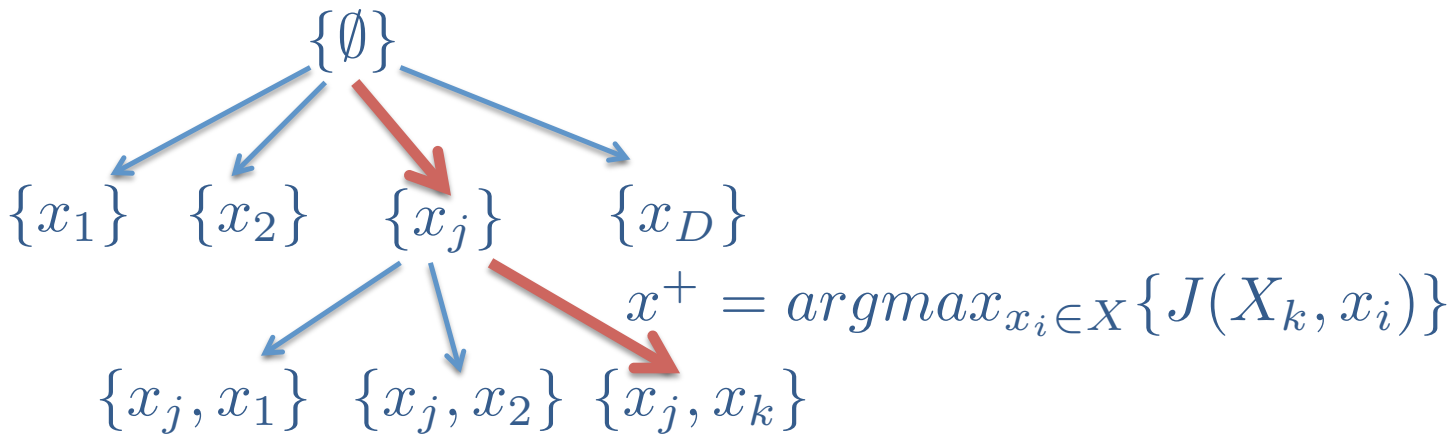
Approximate Monoticity with Branch & Bound

- AMB&B is a variation of the classic Branch & Bound algorithm
- Allows non-monotonic functions
  - relax the cutoff condition that terminates the search on a specific node
  - i.e. replace the limit of number of features with a threshold error rate



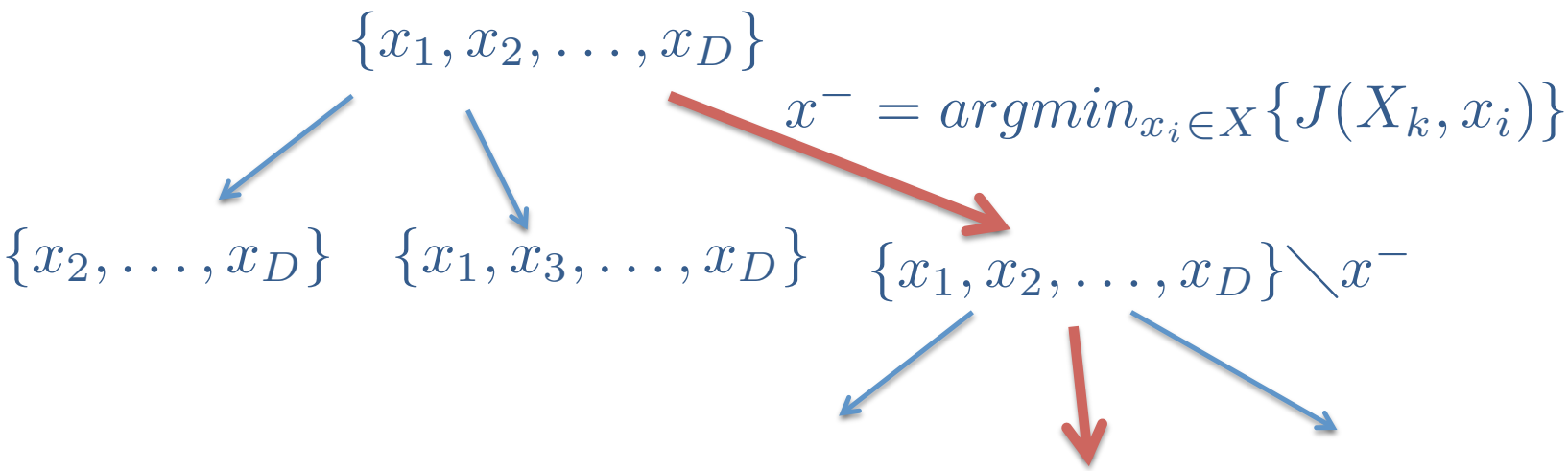
Sequential Search: Sequential Forward Selection (SFS)

- Simplest greedy search algorithm
- Algorithm
  1. Start from the empty set
  2. Select one feature at a time x_i and evaluate $J(X_k, x_i)$
  3. add x^+ the one that maximizes J , K=k+1
  4. If K=d exit else continue to step 2
- Where X_k are the features selected at step k
- It performs best when the optimal subset has a small number of features
- It is a greedy approach and easily fall into local minima



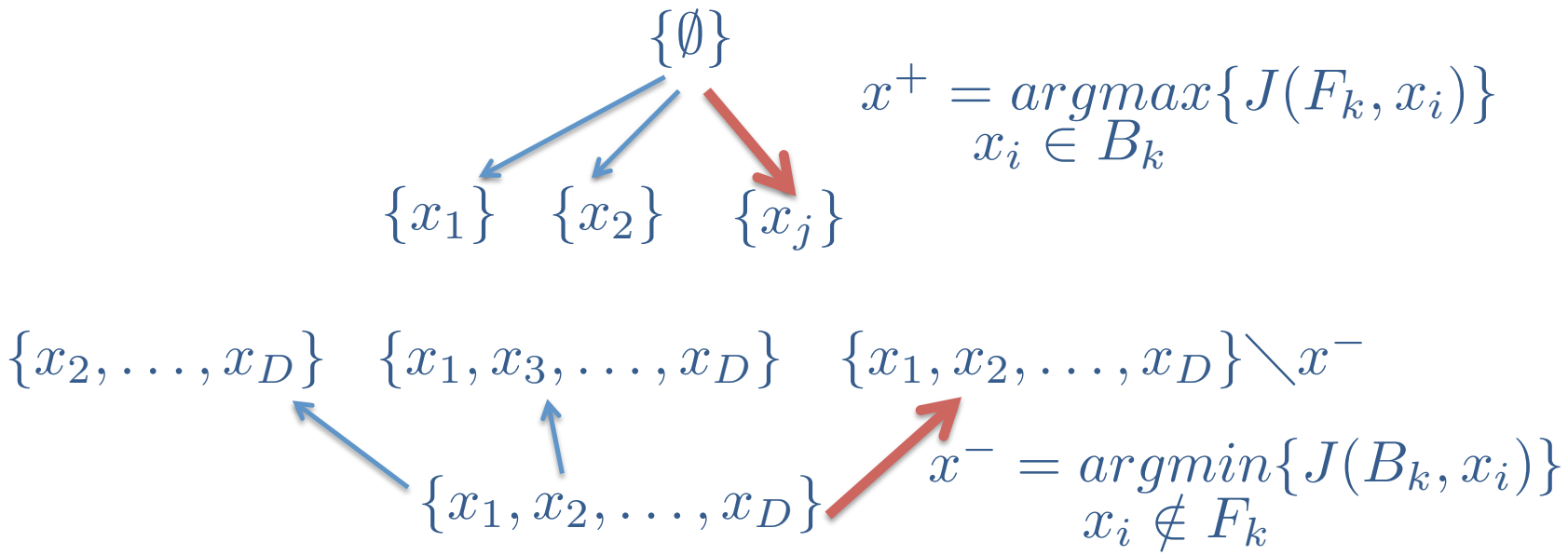
Sequential Search: Sequential Backward Selection (SBS)

- Similar to SFS, but it works in the opposite direction
- Algorithm
  1. Start from the full set, k=D
  2. Select one feature at a time x_i and evaluate $J(X_k, x_i)$
  3. remove x^- the one that minimize J , k=k-1
  4. If k=d exit else continue to step 2
- It performs best when the optimal subset has a large number of features



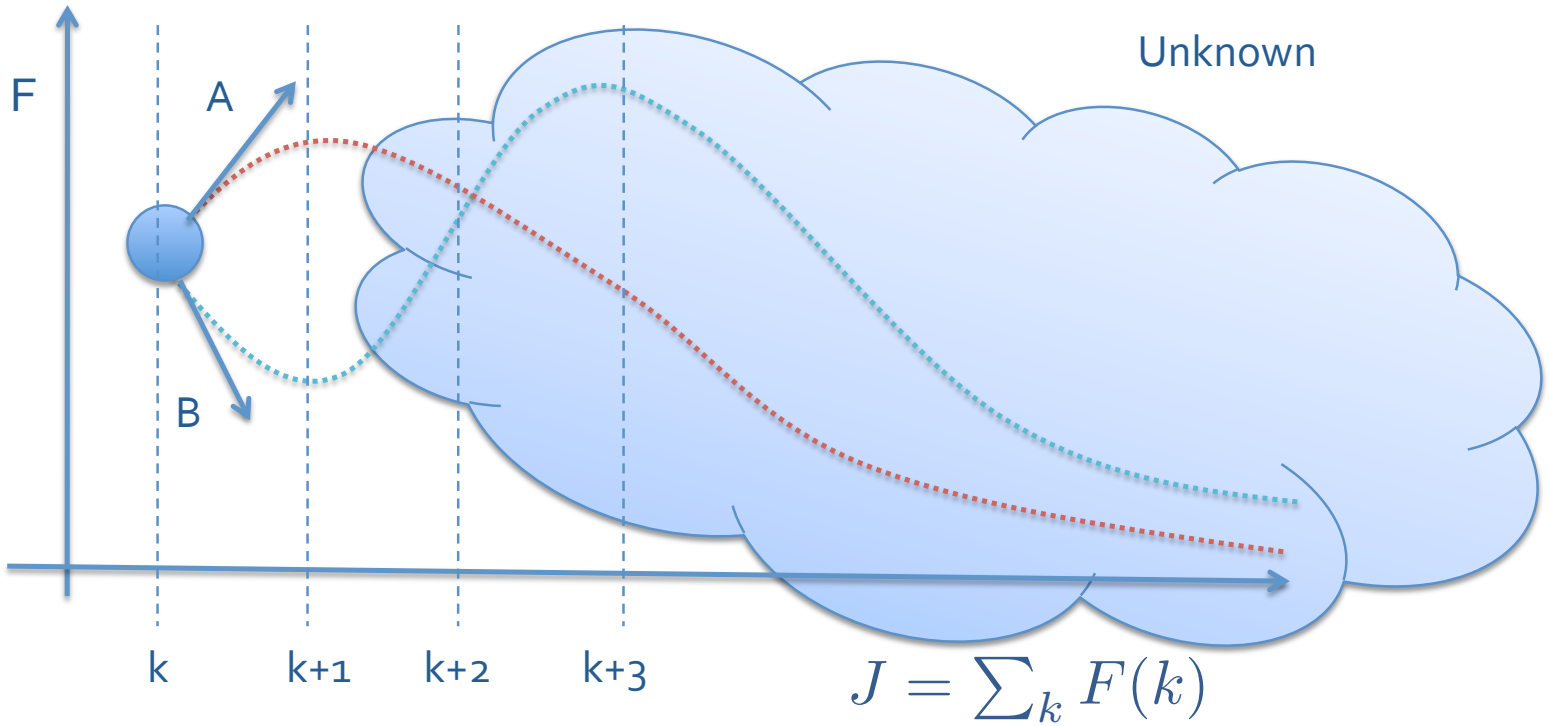
Sequential Search: Bidirectional Search (BDS)

- BDS is a parallel implementation of SFS and SBS:
  - SFS is performed from the empty set F_k
  - SBS is performed from the full set B_k
- SFS and SBS must converge to the same solution we must ensure that:
  - Feature already selected by SFS are not removed by SBS
  - Feature already deleted by SBS are not selected by SFS



Nesting Problem of greedy algorithm

- Once a feature is added or deleted it cannot be brought back anymore
- It is like..



Sequential Search: Plus-L Minus-R Selection (LRS)

- Plus-L Minus-R Selection is a generalization of SFS and SBS
- Given two parameters
  - L: number of features to be added
  - R: number of features to be removed
- Algorithm
  1. If L>R starts from the empty set else start from the Full set
  2. Repeat L time

$$x^+ = argmax_{x_i \in X}\{J(X_k, x_i)\} \quad X_{k+1} = \{X_k, x^+\}$$

  3. Repeat R time

$$x^- = argmin_{x_i \in X}\{J(X_k, x_i)\} \quad X_{k+1} = X_k \setminus x^-$$

  - Goto 2
- LRS attempts to compensate the nesting problem of SFS and SBS with some backtracking capabilities
- lack of a theory to choose L and R
- Suboptimal solution!

Sequential Search: Sequential Floating Selection

- Sequential Floating Selection methods are an extension to the LRS algorithms
- Flexible values for L and R determined automatically from the data and updated dynamically
- It goes very close to the optimal solution, but at an affordable computational cost
- No guarantees to reach the optimal solution!
- Algorithm (SFFS)
  1. Start with empty set
  2. Add best feature
     $$x^+ = argmax_{x_i \in X}\{J(X_k, x_i)\} \qquad X_k^+ = \{X_k, x^+\}$$
  3. Select worst feature
     $$x^- = argmin_{x_i \in X}\{J(X_k^+, x_i)\}$$
  4. If $J(X_k^+ \setminus x^-) > J(X_k^+)$
     Then $X_{k+1} = X_k^+ \setminus x^-$ k=k+1, goto step 3
     Else
     goto step 2

Randomized Search: Random Generation + Sequential Selection

- RGSS introduce randomness into SFS and SBS
- In this way it avoids to fall into local minima
- We consider a number of random combination of features and select the best one
- Algorithm
  1. Repeat for a number of iterations
     a. Generate a random feature subset
     b. Perform SFS on this subset
     c. Perform SBS on this subset
  2. Chose the best subset
- Obviously there is no guarantee to find the best solution

Randomized Search: Genetic Algorithms

- Genetic Algorithms are optimization techniques that mimic the evolutionary process of survival of the best
- They explore portion of solution space and find the solution that maximizes the objective function
  - Different statistical operators enable efficient search strategies

Feature Selection Search Strategies: Summary

Feature Selection Method	Method	Accuracy	Complexity	Notes
Exponential Algorithms	Exhaustive Search	Always find the optimal solution (B&B under the monoticity assumption)	Exponential $O(2^N)$	High accuracy but high complexity
	Branch & Bound			
	Approximate Monoticity with Branch & Bound			
Sequential Algorithms	Sequential Forward Selection (SFS)	No guarantees to find the optimal solution	Quadratic $O(N_{EX}^2)$	Simple and fast, but not optimal
	Sequential Backward Selection (SBS)			
	Plus-L Minus-R Selection			
	Bidirectional Selection			
	Sequential Floating Selection			
Randomized Algorithms	Random Generation plus Sequential Selection	Usually it finds the optimal solution	Generally low	Escape local minima, difficult to choose good parameters
	Genetic Algorithms			