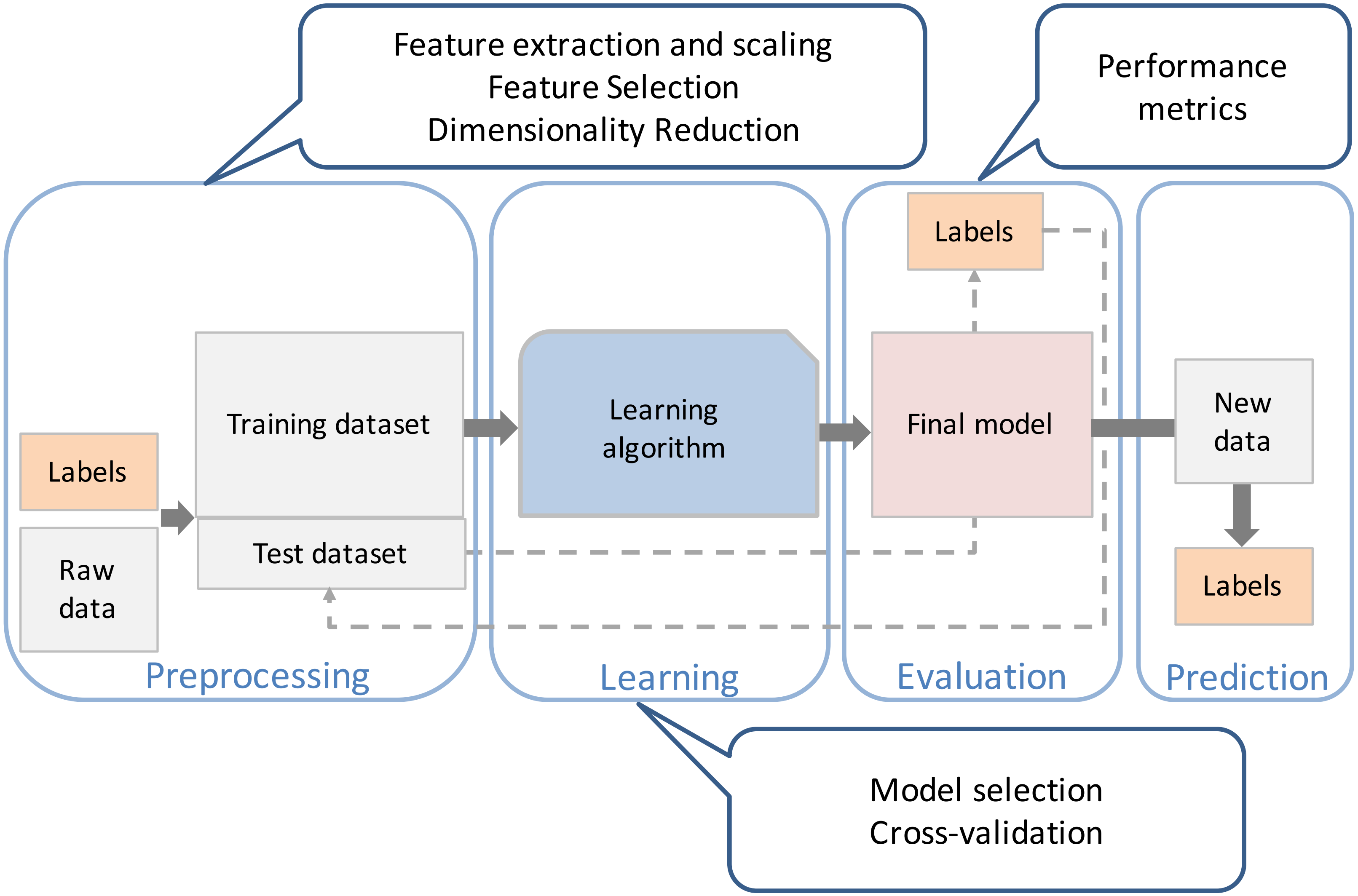
Overview of Machine Learning and Pattern Recognition

Dipartimento di Automatica e Informatica
Politecnico di Torino, Torino, ITALY

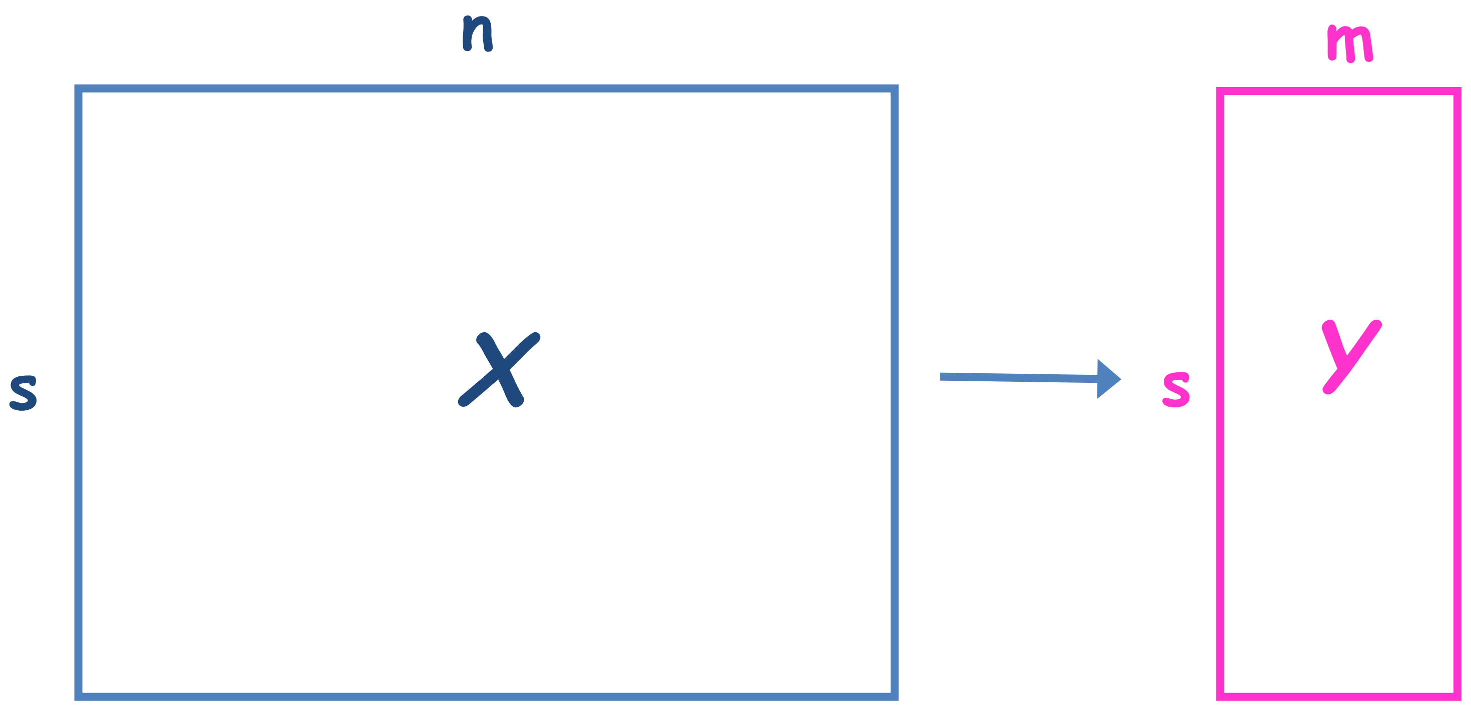
Building a ML system



3. Feature Reduction

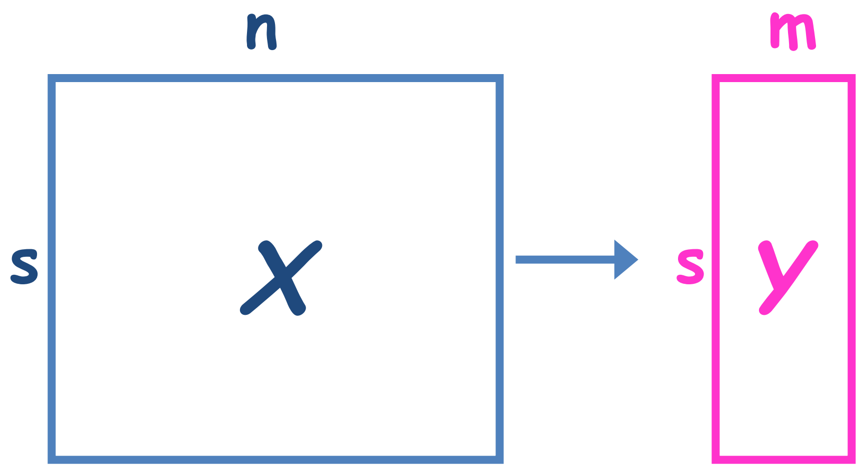
Rationale

- summarization of data with many (n) variables (features) by a smaller set of (m) derived variables.



Rationale

- Summarization of data with many (n) variables by a smaller set of (m) derived variables.



- Balancing of
  - ease of representation, computational speed
  - oversimplification: loss of important or relevant information.

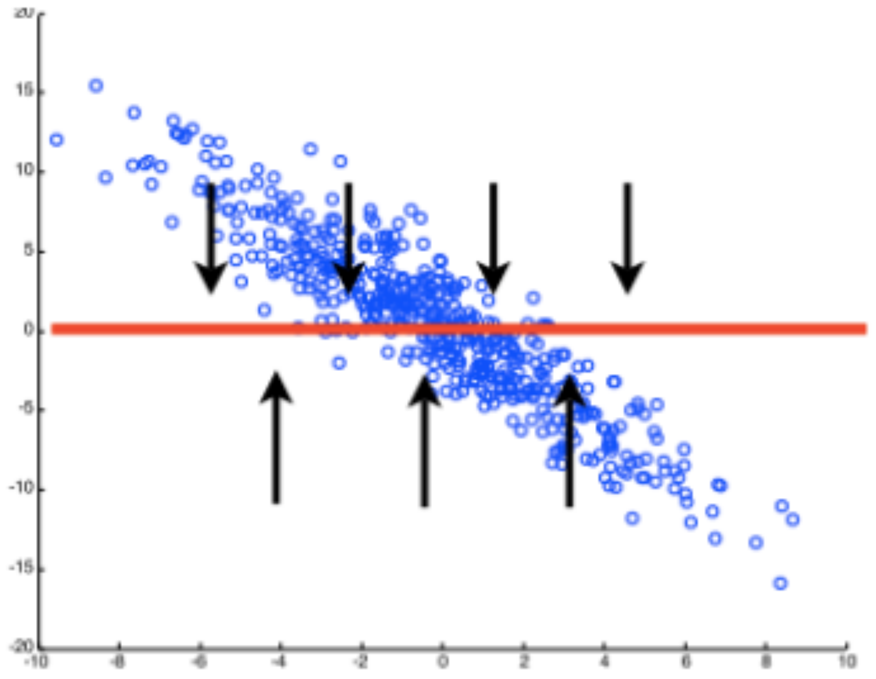
Two main approaches

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \\ x_N \end{bmatrix} \xrightarrow{\text{dimensionality reduction}} \begin{bmatrix} y_1 \\ y_2 \\ \\ y_M \end{bmatrix} = f\left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \\ x_N \end{bmatrix}\right) \qquad \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \\ x_N \end{bmatrix} \xrightarrow{\text{feature selection}} \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ \\ x_{i_M} \end{bmatrix}$$

- **Dimensionality reduction (DR)** is based on mathematical recombinations of the original features.
- **Feature selection (FS)** is based on choosing a sub-set of the original features

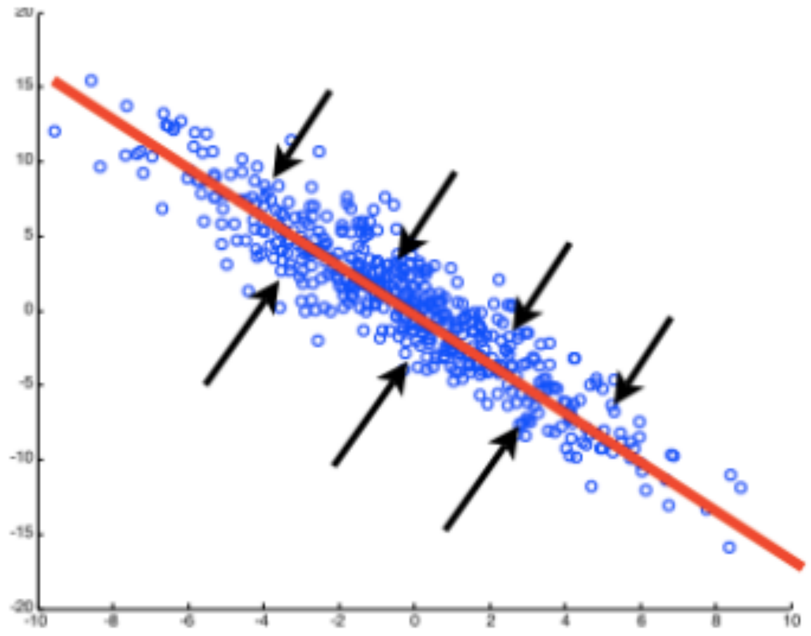
FS vs DR

- **feature selection**: equivalent to projecting feature space to a lower dimensional subspace perpendicular to removed feature, which is a subset of the original one



FS vs DR

- **dimensionality reduction**: allows other kinds of projection (e.g. PCA re-represents data using linear combinations of original features)
- The new space is not a subset, but a **recombination** of the original feature space



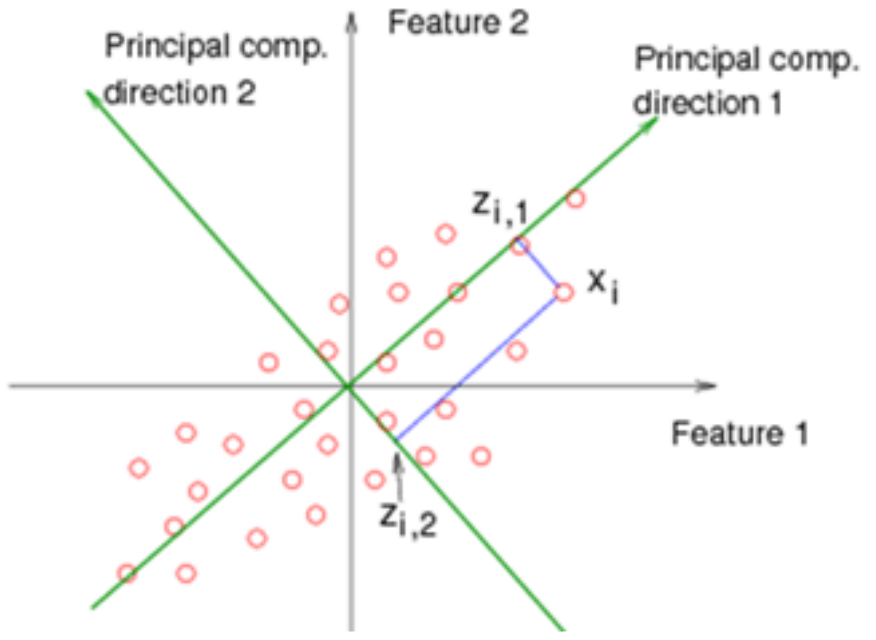
Principal Component Analysis (PCA)

- Takes a data matrix of *s* objects by *n* variables and remaps it to a different space (principal components or principal axes) that are **linear combinations** of the original n variables, in order to **maximize the variance of the data** in the new representation.
- Based on the **eigen-decomposition of the data's covariance** matrix
- The first **m<n** components are the ones that display the **most of the variation** among the objects. Hence, they are assumed to be the most representative ones.
- Ratio: reducing the size of the feature space **while retaining as much of the information as possible**.

Principal Component Analysis (PCA)

Example in a 2-dimensional space:

- PCA Performs a linear mapping of the data to a lower-dimensional space, in such a way that the variance of the data in the low-dimensional representation is maximized



Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)

PCA

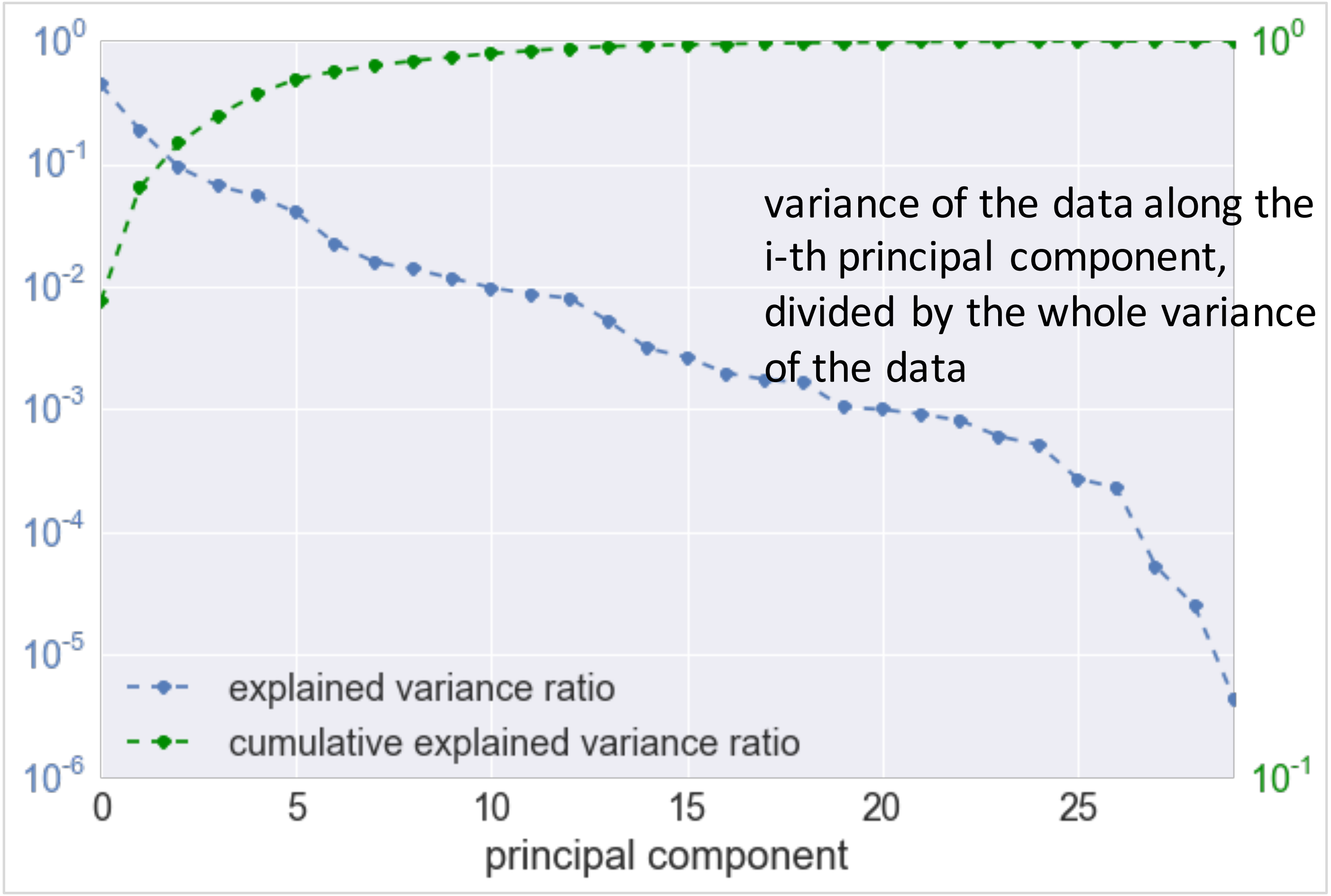
- Choose directions such that a total variance of data will be maximum
  - Maximize Total Variance
  - Higher variance $\rightarrow$ Better **separability** of the data

- Choose directions that are orthogonal
  - Minimize correlation between different features
  - Lower correlation $\rightarrow$ Lower **redundancy**

- Choose m<n orthogonal directions which maximize total variance

PCA example

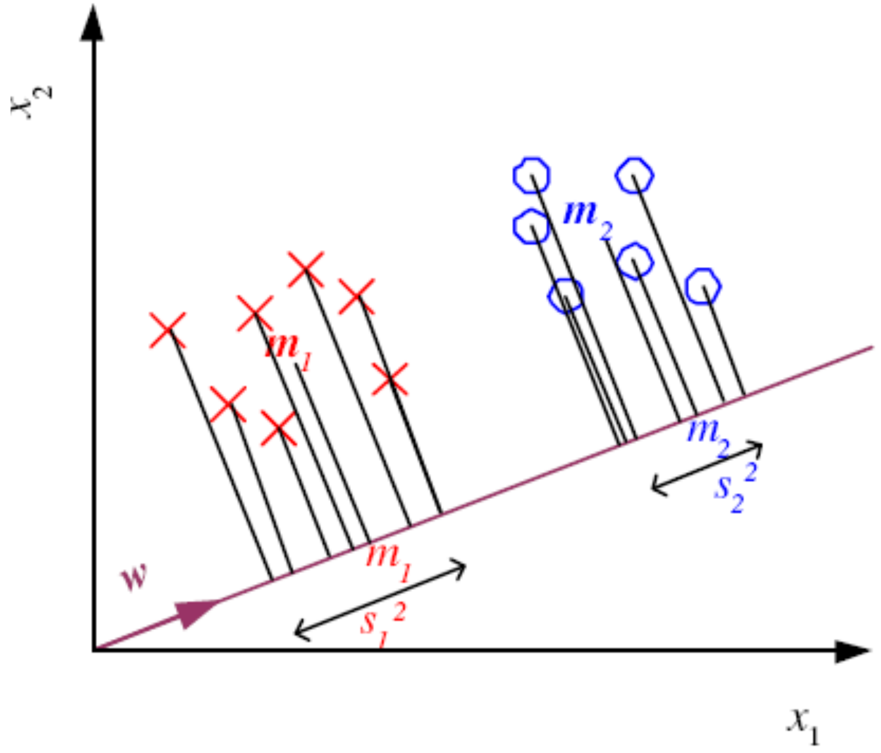
PCA applied on UCI ML Breast Cancer Wisconsin (Diagnostic) dataset (32 features)



variance of the data along the i-th principal component, divided by the whole variance of the data

Linear Discriminant Analysis (LDA)

- PCA is unsupervised (does not take into account class information)
- Supervised extension: LDA
- LDA: find a low-dimensional space such that when ***x*** is projected, classes are well-separated

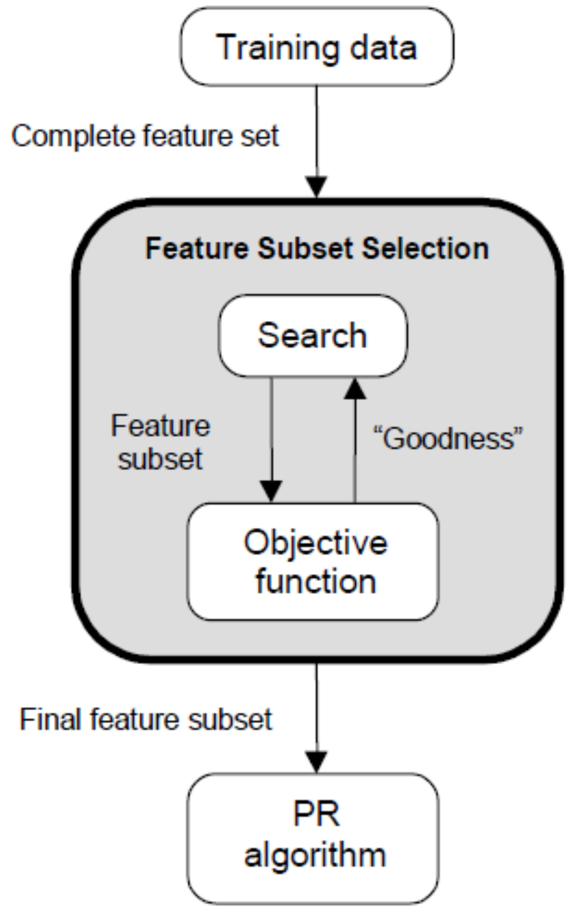


Feature selection vs Dimensionality reduction

- Dimensionality Reduction
  - When classifying novel patterns, all features need to be computed.
  - The measurement units of the original features are lost.
  - PCA,LDA works on linear combinations of the features (does not provide good results for all data distributions)
- Feature Selection
  - When classifying novel patterns, only a small number of features need to be computed (i.e., faster classification).
  - The measurement units of the original features are preserved.

FS: steps

- Feature selection is an **optimization** problem.
  - Step 1: Search the space of possible feature subsets.
  - Step 2: Pick the subset that is optimal or near-optimal with respect to some objective function.



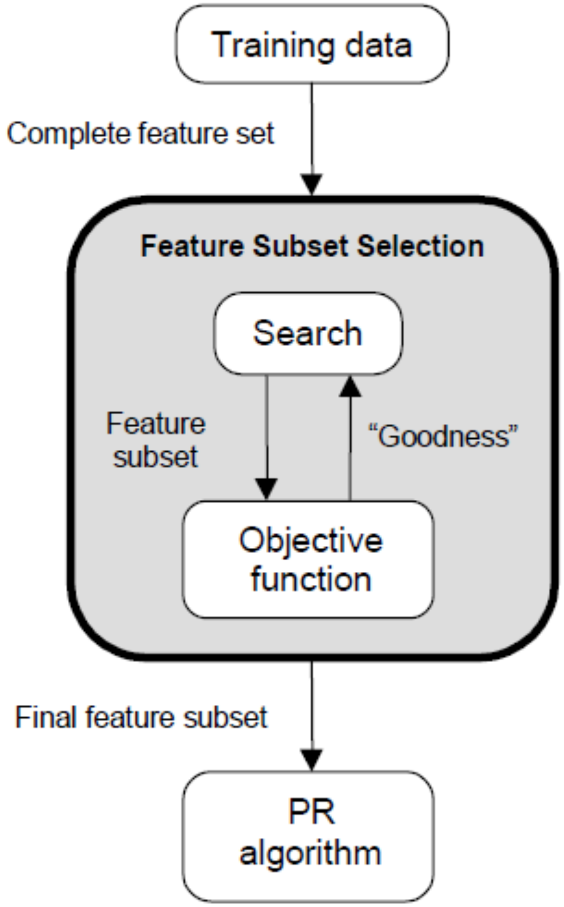
FS: steps

Evaluation strategies

- Filter methods
- Wrapper methods

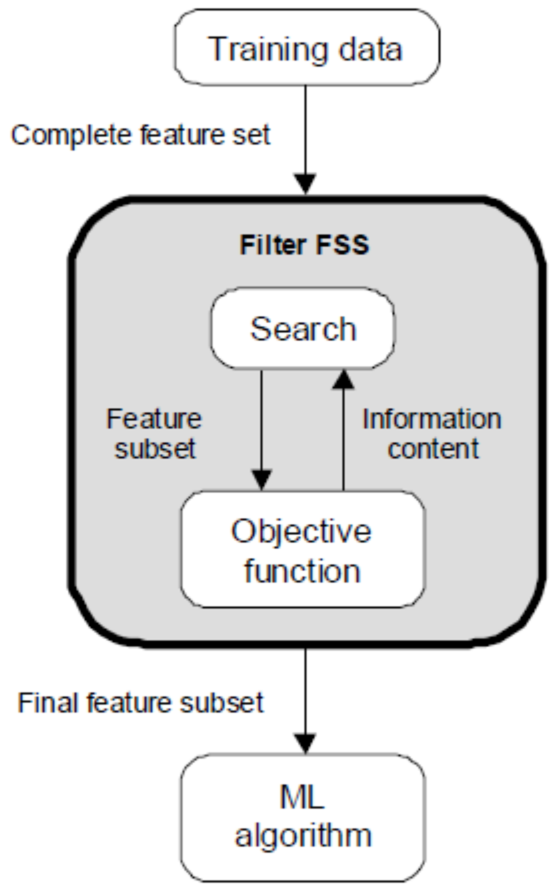
Search strategies

- Exhaustive
- Heuristic
  - Sequential (SFS, etc.)
  - Randomized (GA)



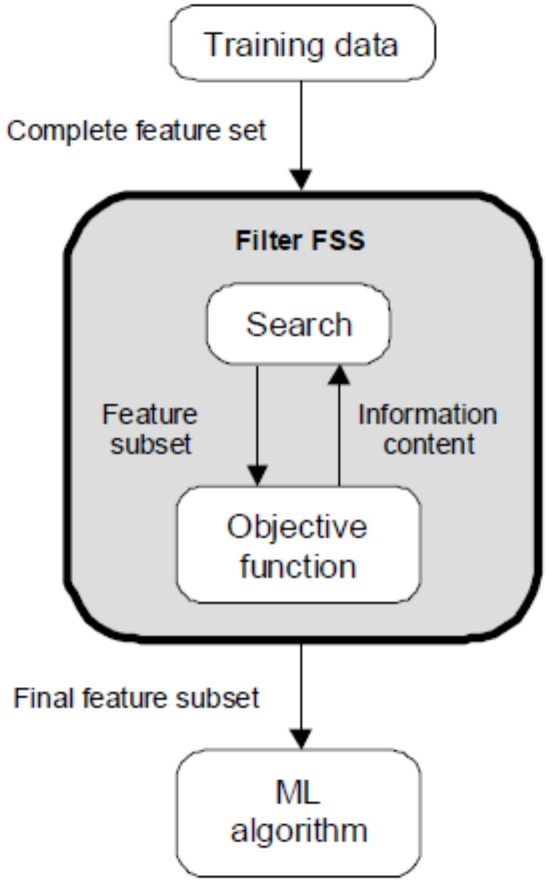
Evaluation strategies

- Wrapper Methods
  - Evaluation uses criteria **related** to the classification algorithm.
  - The objective function is a pattern classifier, which evaluates feature subsets by their predictive accuracy.



Evaluation strategies

- Filter Methods
  - Evaluation is **independent** of the classification algorithm.
  - The objective function evaluates feature subsets by their information content, typically interclass distance, statistical dependence or information-theoretic measures.



Wrapper FS: pros and cons

- **Accuracy:** wrappers generally achieve better recognition rates than filters, as they are tuned to the specific classifier

- **Generalization capability:** Wrappers have inherent mechanism to avoid overfitting, as they are typically based on crossvalidation

- **Execution time:** the wrapper must train a classifier for each feature subset, which is unfeasible in case of computationally intensive ML methods

- **Generality:** the solution is tied to a specific classifier.

Filter FS: pros and cons

- **Execution time:** filters generally involve a non-iterative computation, which is usually faster than a classifier training session

- **Generality:** filters evaluate intrinsic properties of the data. Hence, their results may apply to a large set of ML methods

- **Tendency to select large subsets:** as most of the times the objective function of the filter is monotonic, the filter may tend to select the full feature set as optimal solution. This forces the user to set an arbitrary cutoff on the number of features to be selected

Search strategies

- Assuming n features, an exhaustive search would require:
  - Examining all the $\binom{n}{d}$ possible subsets of size *d*.
  - Selecting the subset that performs the best according to the criterion function.
- The number of subsets grows with n, making exhaustive search impractical.
- In practice, heuristics are used to speed-up search but they **cannot** guarantee full optimality.

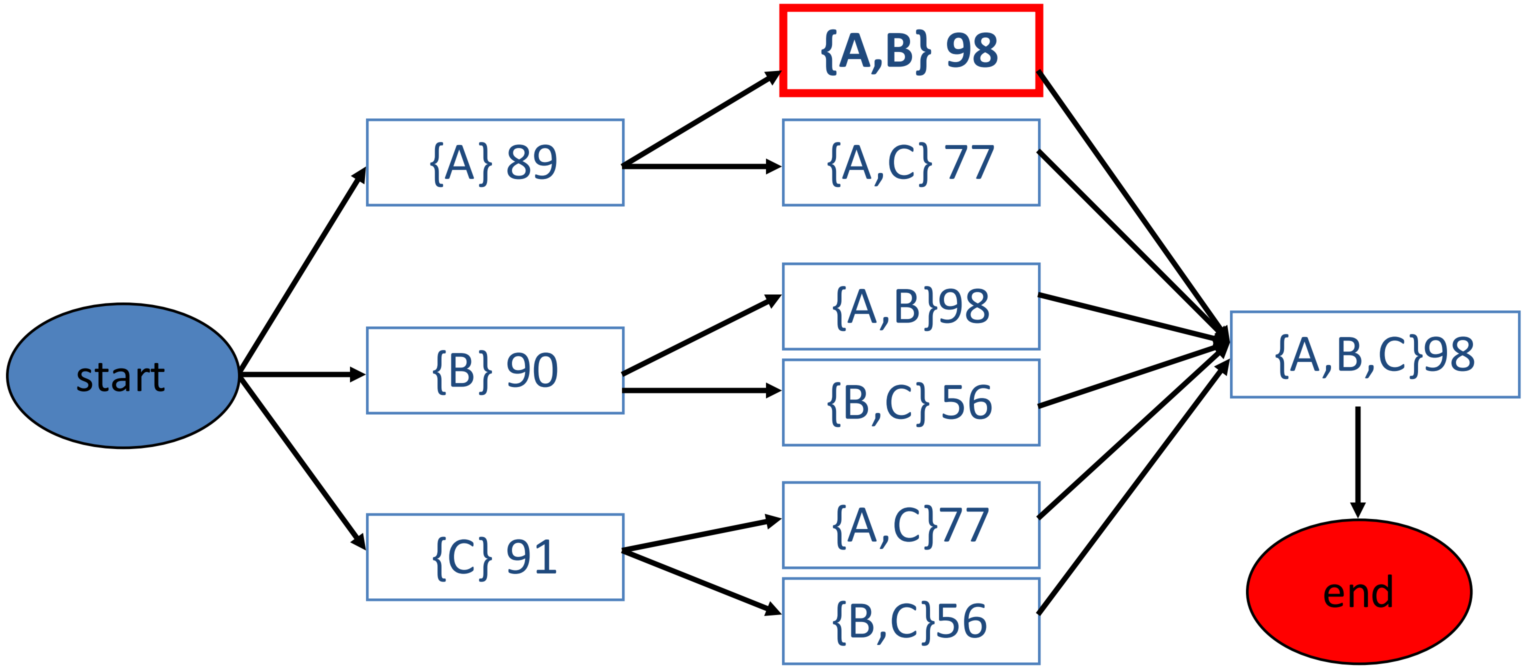
Example

Say we have features A, B, C and classifier *M*. We want to predict T given the smallest possible subset of {A,B,C}, while achieving maximal classification performance (accuracy)

FEATURE SET	CLASSIFIER	PERFORMANCE
{A,B,C}	*M*	**98%**
{A,B}	*M*	**98%**
{A,C}	*M*	77%
{B,C}	*M*	56%
{A}	*M*	89%
{B}	*M*	90%
{C}	*M*	91%

Example

For large sets of features it is unfeasible to search for all possible subsets exhaustively; instead, we rely on ***heuristic search*** of the space of all possible feature subsets.



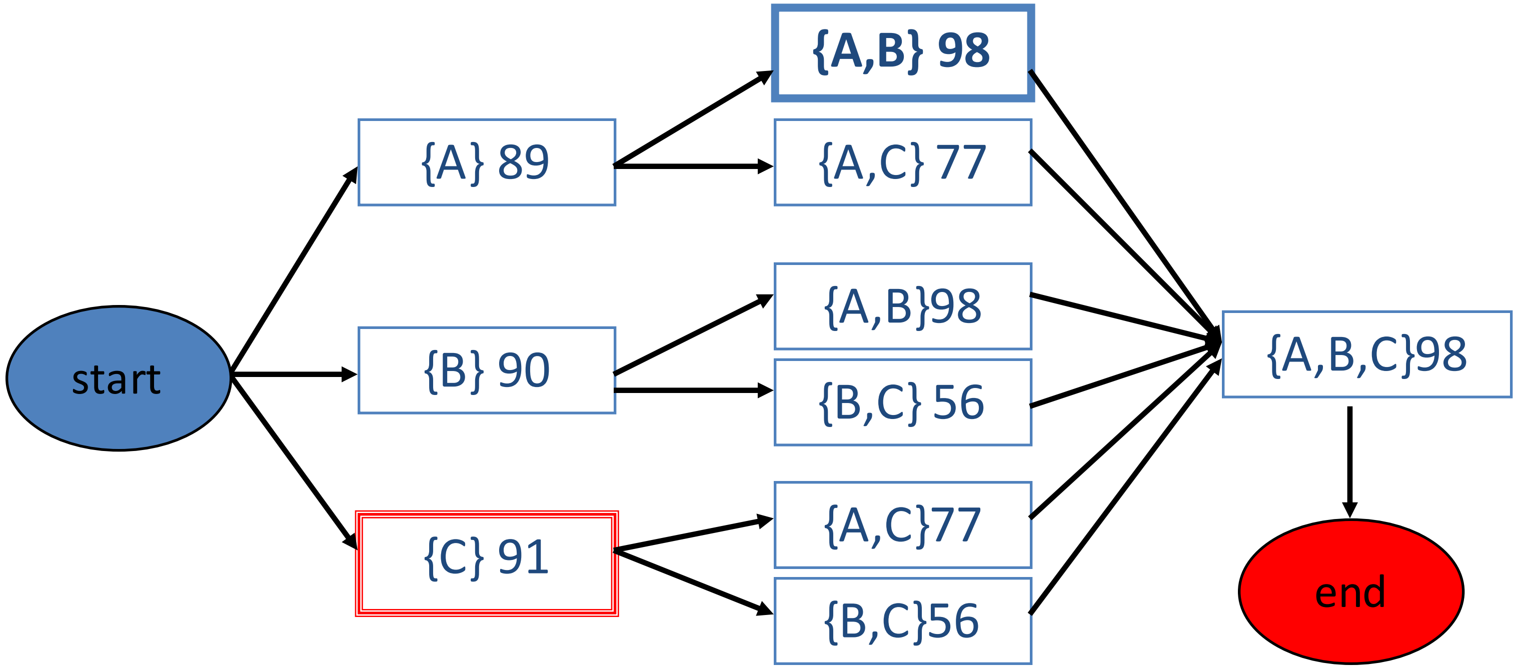
Example

A common example of heuristic search is **hill climbing**: keep adding features one at a time, until no further improvement can be achieved.

Disadvantage: the subset space is not fully explored → sub-optimal result

---

Forward selection: Start from empty set and keep adding one feature



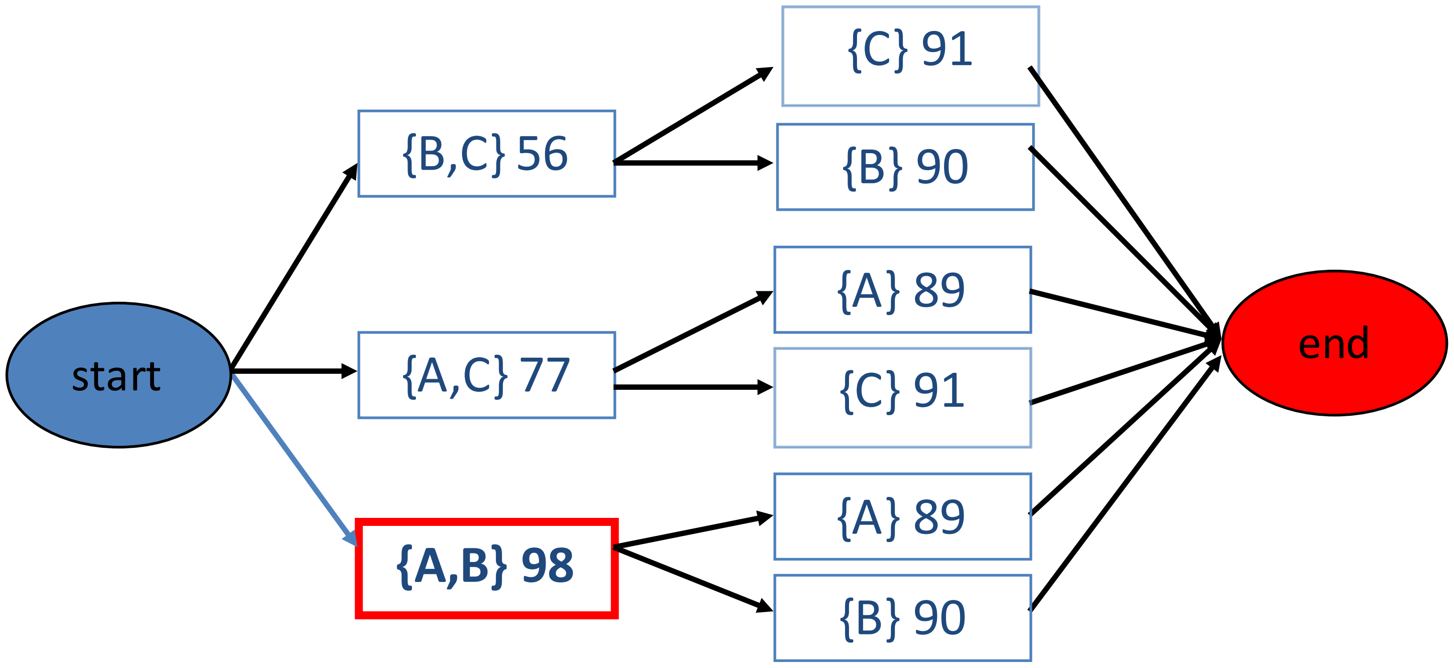
Example

A common example of heuristic search is **hill climbing**: keep adding features one at a time, until no further improvement can be achieved.

Disadvantage: the subset space is not fully explored → sub-optimal result

---

Backward elimination: Start from full set and keep removing one feature



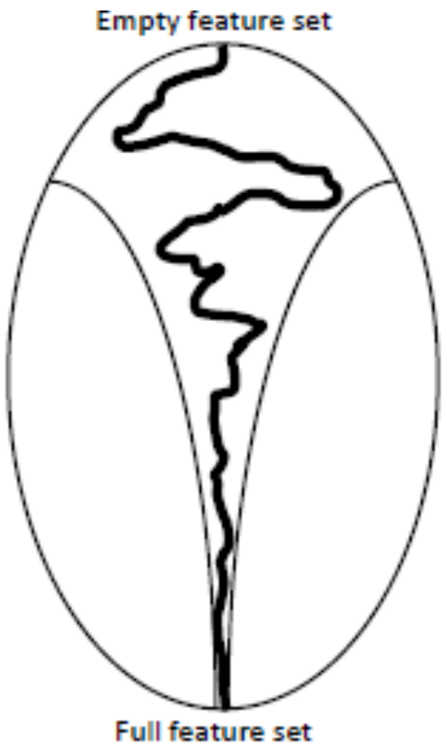
Naïve Search

- Train the ML model with each individual feature, taken alone. Measure classification accuracy.
- Sort the given n features in order of the classification accuracy obtained with each of them.
- Select the top d features from this sorted list.
- Disadvantage
  - Correlation among features is not considered.
  - The best pair of features may not even contain the best individual feature.

Sequential forward selection (SFS) (heuristic search)

- First, the best **single** feature is selected (i.e., using classification accuracy as criterion function).
- Then, **pairs** of features are formed using one of the remaining features and this best feature, and the best pair is selected.
- Next, **triplets** of features are formed using one of the remaining features and these two best features, and the best triplet is selected.
- This procedure continues until a predefined number of features are selected, or even after ALL of them have been selected.

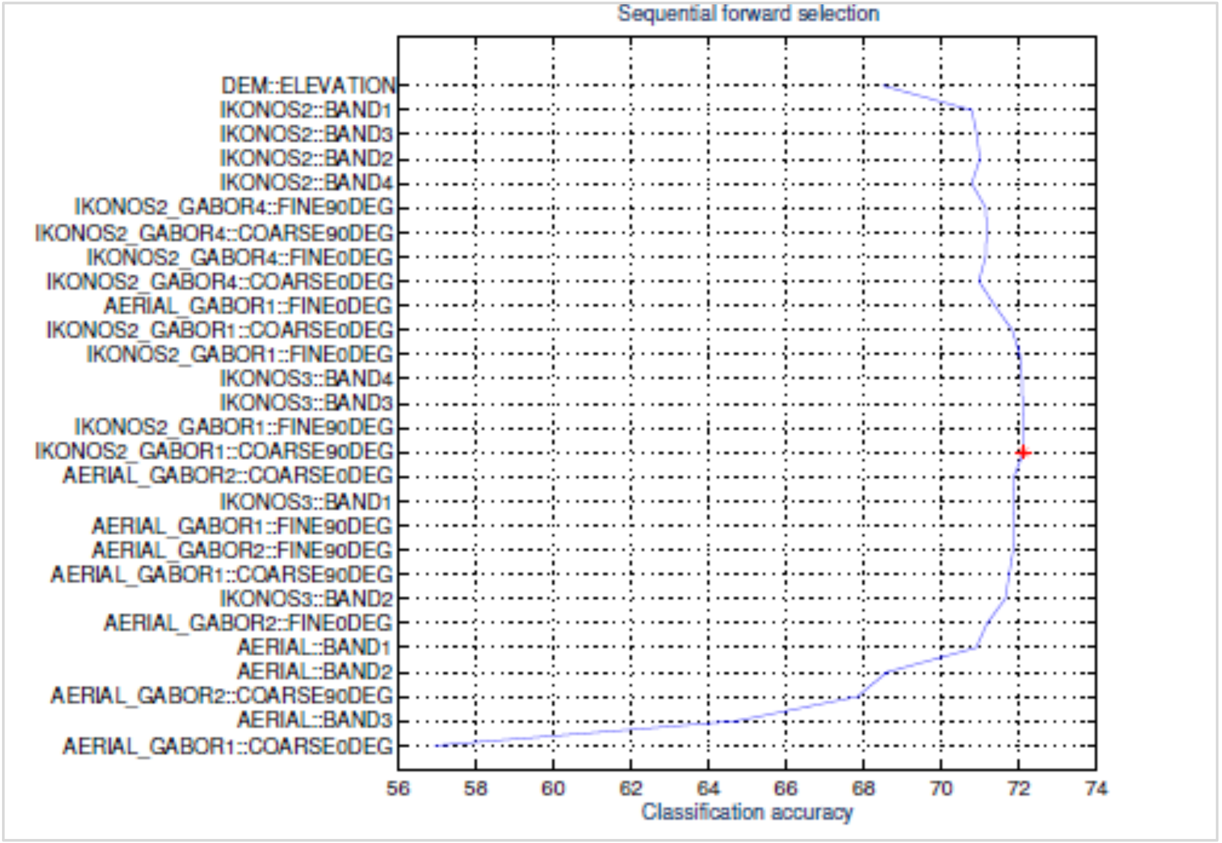
1. Start with the empty set $Y_0 = \{\emptyset\}$
2. Select the next best feature $x^+ = \underset{x \notin Y_k}{\arg\max}\, J(Y_k + x)$
3. Update $Y_{k+1} = Y_k + x^+;\ k = k + 1$
4. Go to 2



SFS performs best when the optimal subset is small.

Example

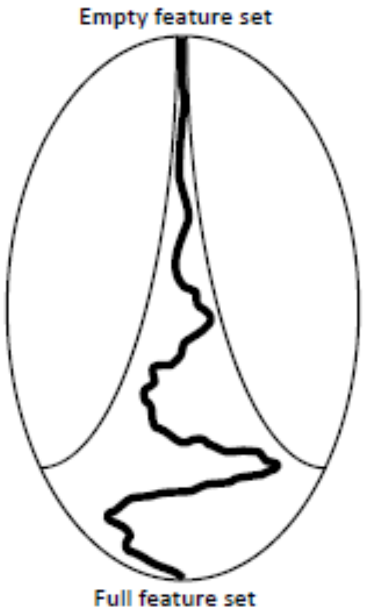
features added at each iteration



Results of sequential forward feature selection for classification of a satellite image using 28 features. x-axis shows the classification accuracy (%) and y-axis shows the features added at each iteration (the first iteration is at the bottom). The highest accuracy value is shown with a star.

Sequential backward selection (SBS) (heuristic search)

- First, the criterion function is computed for all **n** features.
- Then, each feature is deleted one at a time, the criterion function is computed for all subsets with **n-1** features, and the worst feature is discarded.
- Next, each feature among the remaining **n-1** is deleted one at a time, and the worst feature is discarded to form a subset with **n-2** features.
- This procedure continues until a predefined number (or none) of features are left.

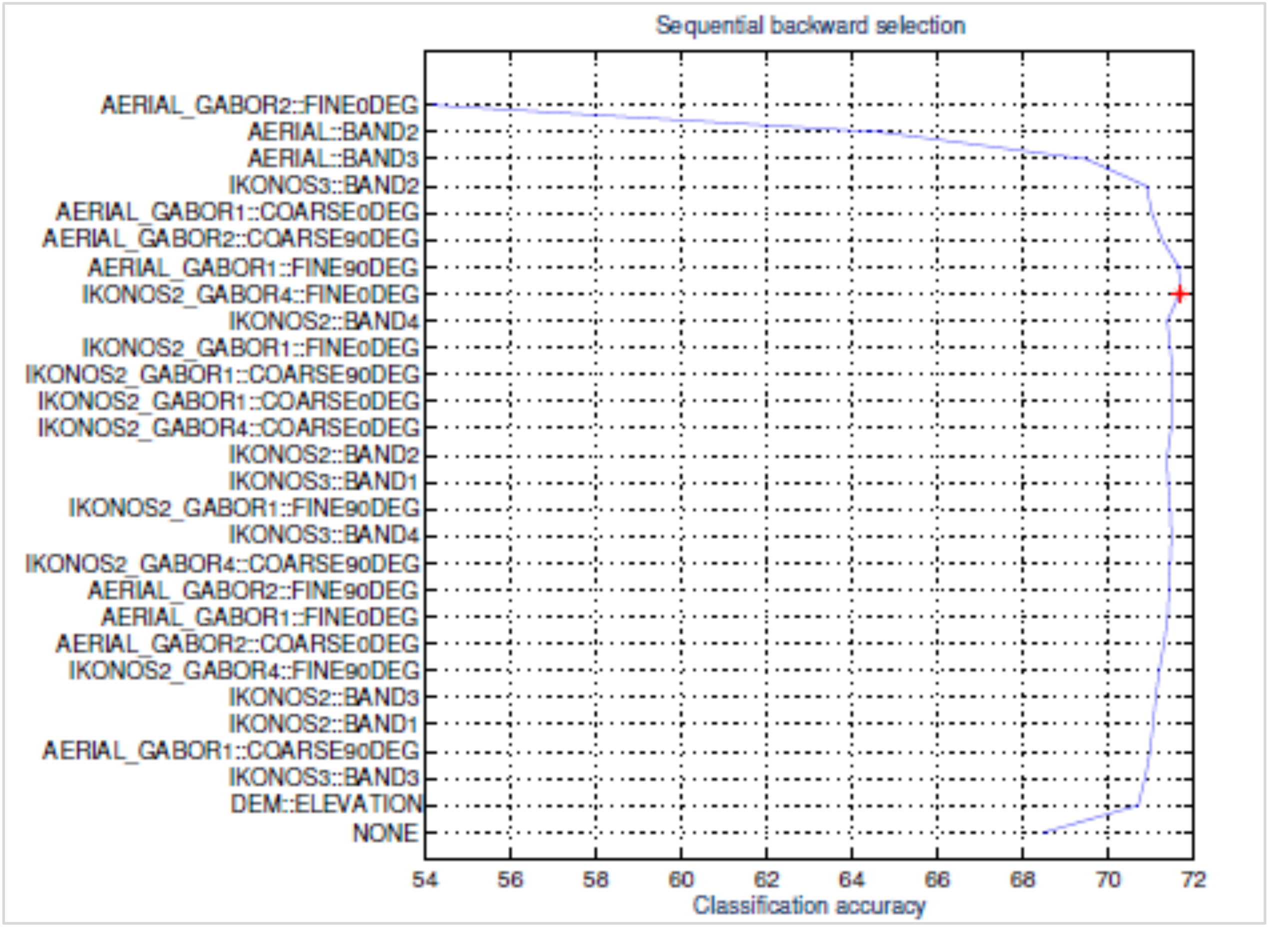


```
1. Start with the full set $Y_0 = X$
2. Remove the worst feature $x^- = \arg\max_{x \in Y_k} J(Y_k - x)$
3. Update $Y_{k+1} = Y_k - x^-$; $k = k + 1$
4. Go to 2
```

SBS performs best when the optimal subset is large.

Example

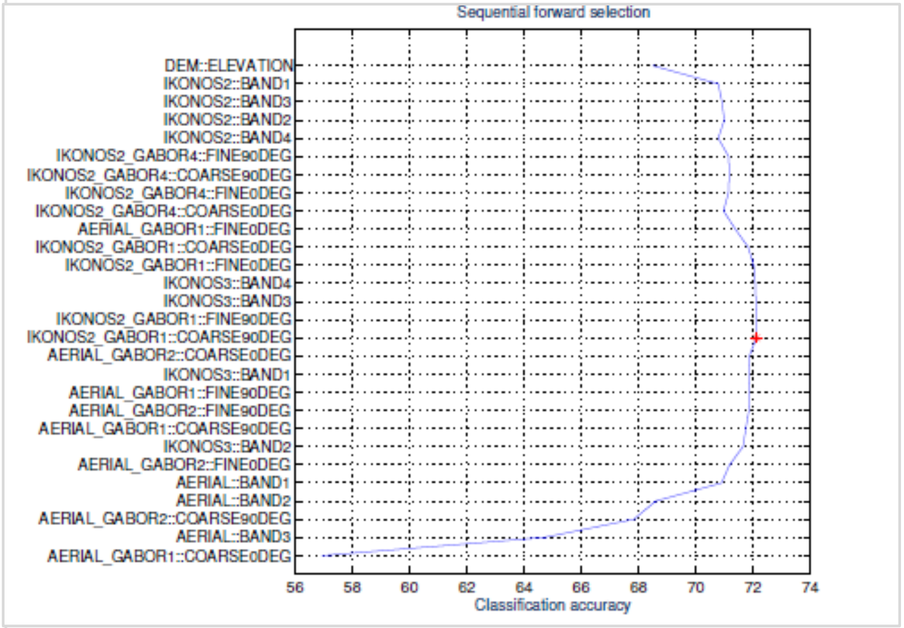
features removed at each iteration

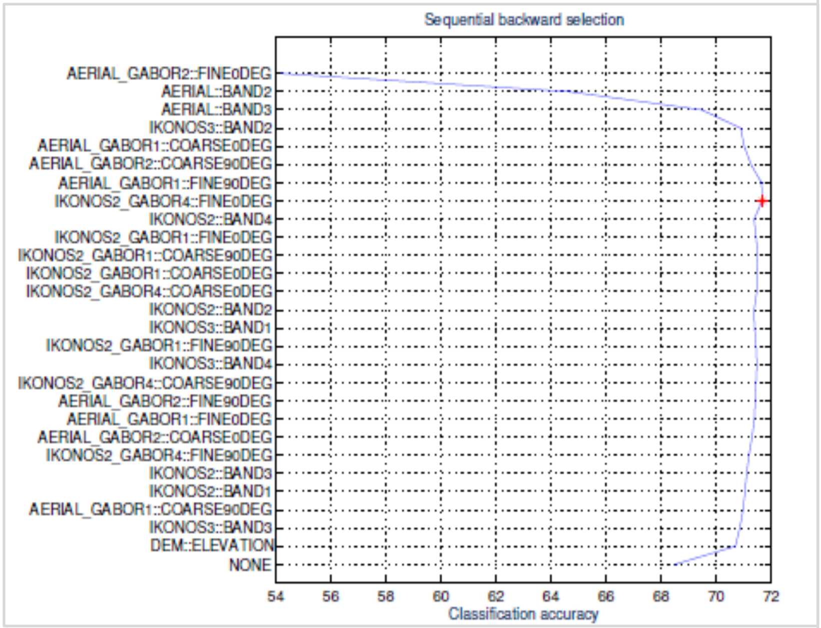


Results of sequential backward feature selection for classification of a satellite image using 28 features. x-axis shows the classification accuracy (%) and y-axis shows the features removed at each iteration (the first iteration is at the top). The highest accuracy value is shown with a star.

SFS vs SBS

- If there is no way to guess the approximate size of the optimal feature set, the methods are equivalent
- As the search is heuristic, there is no guarantee that the two approaches, even on the same dataset, will converge to the same result!



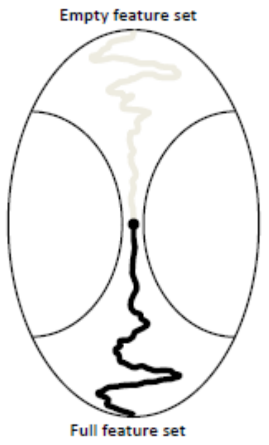


Limitations of SFS and SBS

- The main limitation of SFS is that it is unable to remove features that become non useful after the addition of other features.
- The main limitation of SBS is its inability to re-evaluate the usefulness of a feature after it has been discarded.
- There are some generalizations of SFS and SBS:
  - Bidirectional search (BDS)
  - Plus-L, minus-R" selection (LRS)
  - Sequential floating forward/backward selection (SFFS and SFBS)

Bidirectional search (BDS)

- BDS applies SFS and SBS simultaneously:
  - SFS is performed from the empty set.
  - SBS is performed from the full set.
- To guarantee that SFS and SBS converge to the same solution:
  - Features already selected by SFS are not removed by SBS.
  - Features already removed by SBS are not added by SFS.



1. Start SFS with $Y_F = \{\emptyset\}$
2. Start SBS with $Y_B = X$
3. Select the best feature

$$x^+ = \underset{\substack{x \notin Y_{F_k} \\ x \in F_{B_k}}}{\arg\max} J\left(Y_{F_k} + x\right)$$

$$Y_{F_{k+1}} = Y_{F_k} + x^+$$

4. Remove the worst feature

$$x^- = \underset{\substack{x \in Y_{B_k} \\ x \notin Y_{F_{k+1}}}}{\arg\max} J\left(Y_{B_k} - x\right)$$

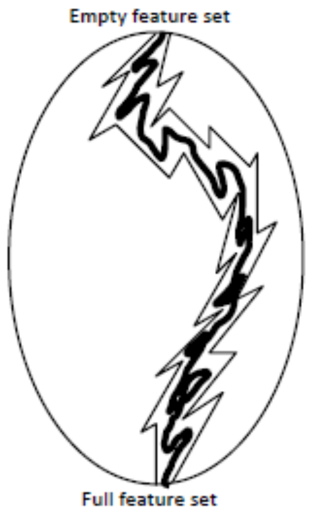
$$Y_{B_{k+1}} = Y_{B_k} - x^-;\ k = k+1$$

5. Go to 2

“Plus-L, minus-R” selection (LRS)

- A generalization of SFS and SBS
  - If L>R, LRS starts from the empty set and:
    - Repeatedly add L features
    - Repeatedly remove R features
  - If L<R, LRS starts from the full set and:
    - Repeatedly removes R features
    - Repeatedly add L features

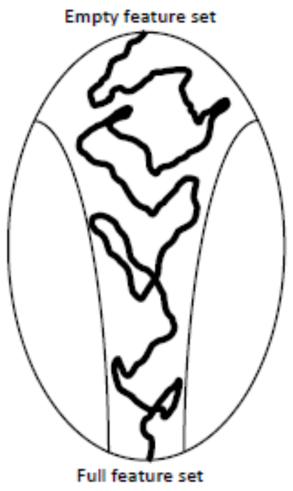
```
1. If L>R      then $Y_0 = \{\emptyset\}$
               else $Y_0 = X$; go to step 3
2. Repeat L times
               $x^+ = \arg\max_{x \notin Y_k} J(Y_k + x)$
               $Y_{k+1} = Y_k + x^+;\ k = k + 1$
3. Repeat R times
               $x^- = \arg\max_{x \in Y_k} J(Y_k - x)$
               $Y_{k+1} = Y_k - x^-;\ k = k + 1$
4. Go to 2
```



Main limitation: how to choose the optimal values of L and R??

Sequential floating forward/backward selection (SFFS and SFBS)

- An extension to LRS:
  - Rather than fixing the values of L and R, floating methods determine these values from the data.
  - The dimensionality of the subset during the search can be thought to be "floating" up and down



- Two floating methods:
  - Sequential floating forward selection (SFFS)
  - Sequential floating backward selection (SFBS)

P. Pudil, J. Novovicova, J. Kittler, Floating search methods in feature selection, Pattern Recognition Lett. 15 (1994) 1119–1125.

Sequential floating forward selection (SFFS)

- Sequential floating forward selection (SFFS) starts from the empty set.
- After each forward step, SFFS performs backward steps as long as the objective function increases.

```
1. $Y = \{\emptyset\}$
2. Select the best feature
       $x^+ = \underset{x \notin Y_k}{\arg\max}\, J(Y_k + x)$
       $Y_k = Y_k + x^+; k = k + 1$
3. Select the worst feature*
       $x^- = \underset{x \in Y_k}{\arg\max}\, J(Y_k - x)$
4. If $J(Y_k - x^-) > J(Y_k)$ then
       $Y_{k+1} = Y_k - x^-;\ k = k + 1$
       Go to step 3
   Else
       Go to step 2
```

*Notice that you'll need to do book-keeping to avoid infinite loops

Sequential floating backward selection (SFBS)

- Sequential floating backward selection (SFBS) starts from the full set.
- After each backward step, SFBS performs forward steps as long as the objective function increases.

Genetic Algorithms (GAs)

- A way to perform feature selection via a **randomized** search is to use GAs
- What are GAs?
  - A global optimization technique for searching very large solution spaces.
  - Inspired by the biological mechanisms of natural selection and reproduction.
- Main characteristics of GAs
  - Searches probabilistically using a population of possible solutions.
  - Each solution is properly encoded as a string of symbols.
  - Uses an objective (or fitness) function to evaluate the "goodness" of each solution.

Population of encoded solutions

10010110…
01100011…
01100011…
10100100…

Encoding

- Each solution in the search space is represented as a finite length string (chromosome) over some finite set of symbols.

e.g., using binary encoding

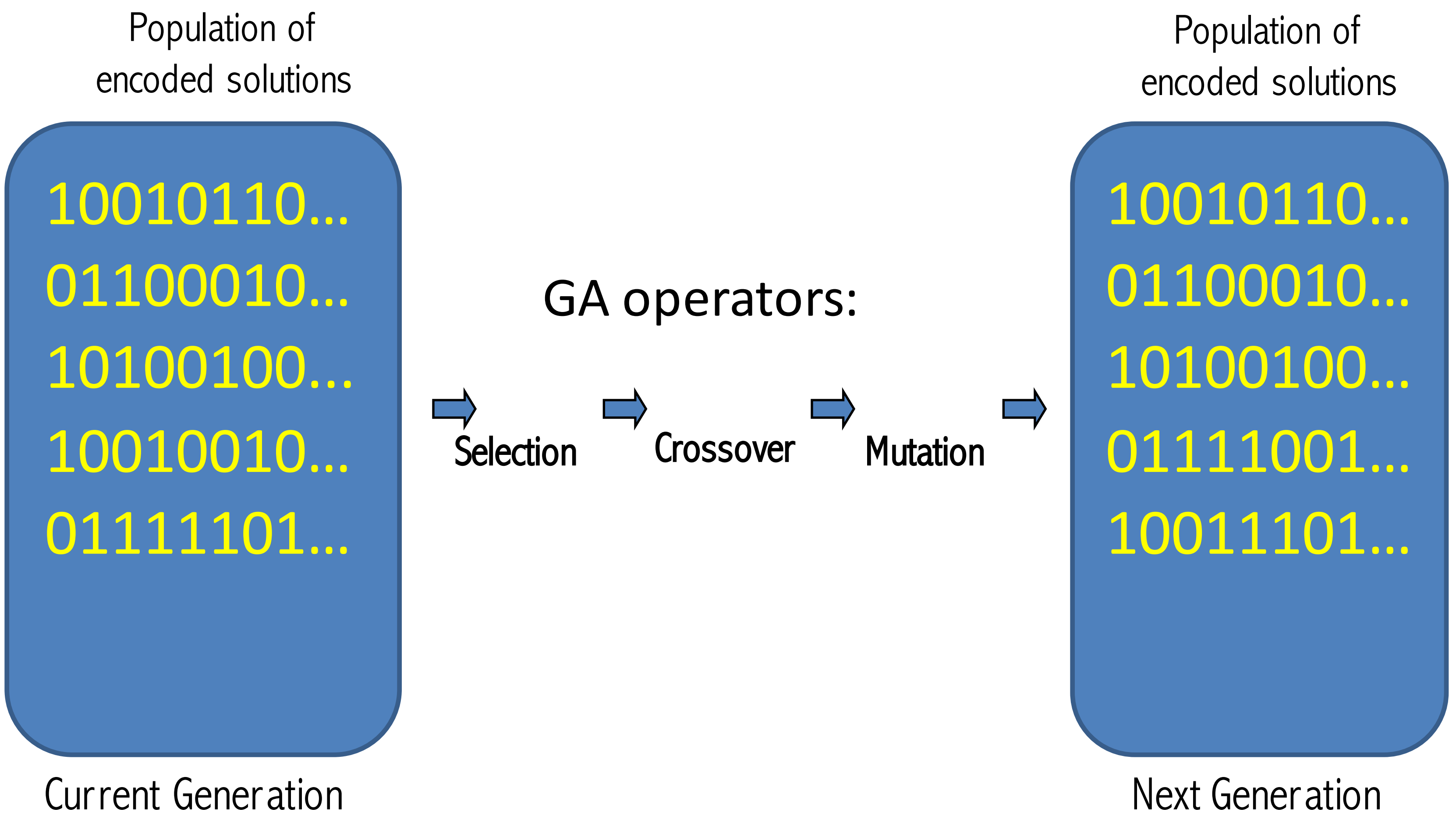
$$(11,6,9) \rightarrow (1011\_0110\_1001) \rightarrow (101101101001)$$

Fitness evaluation

- A fitness function is used to evaluate the goodness of each solution.
- The fitness function is “problem specific”.

$$Fitness = f(decode(chromosome))$$

Searching



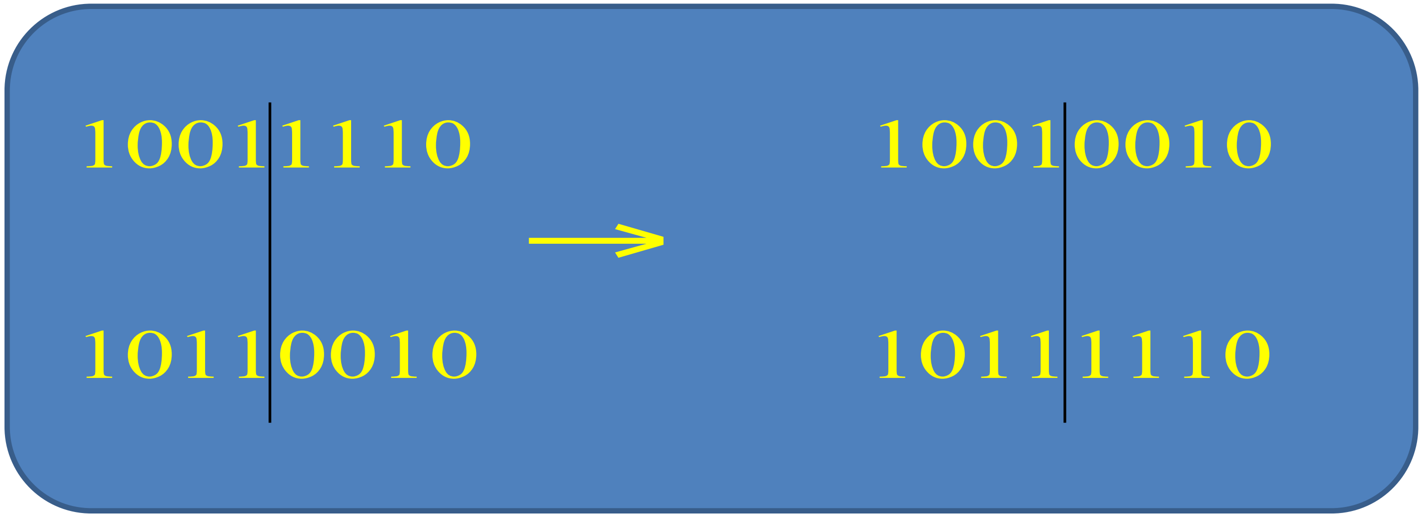
Population of encoded solutions
10010110…
01100010…
10100100…
10010010…
01111101…
Current Generation
GA operators:
Selection
Crossover
Mutation
Population of encoded solutions
10010110…
01100010…
10100100…
01111001…
10011101…
Next Generation

Selection

- Probabilistically filters out solutions that perform poorly, choosing high performance solutions to exploit:
  - Rank chromosomes based on fitness
  - Copy the “best” chromosome as it is to the next generation (elitism)
  - Set a random fitness threshold
  - Select chromosomes with fitness higher than the threshold as parents of the next generation of chromosomes
  - Generate the new chromosomes by crossover and mutation
  - Replace the old generation with the new one

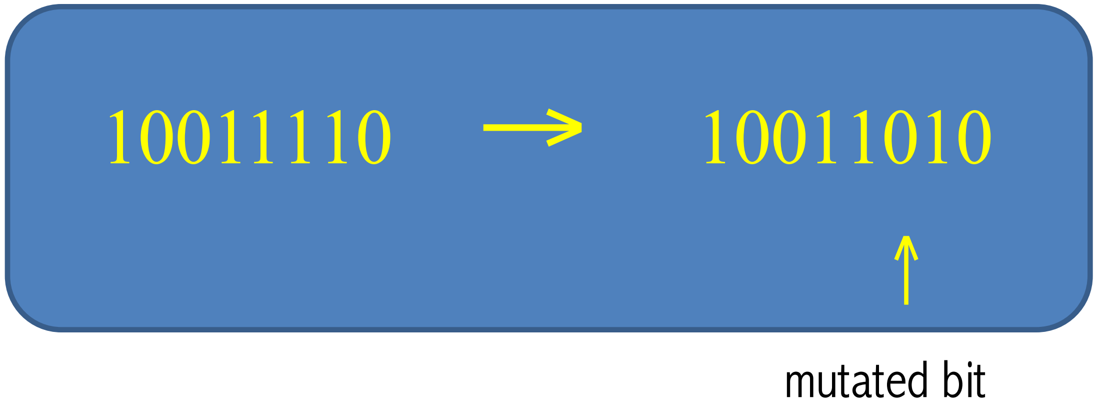
Crossover

- Explore new solutions
- **Crossover:** information exchange between strings.
  - Generate new chromosomes that, hopefully, will retain good features from the previous generation.
  - It is applied to randomly selected pairs of chromosomes with a probability equal to a given *crossover rate*.



Mutation

- Explore new solutions
- **Mutation**: restore lost genetic material.
  - It protects GAs against irrecoverable loss of good solution features.
  - It changes a character of some chromosome with a probability equal to a very low given *mutation rate*.

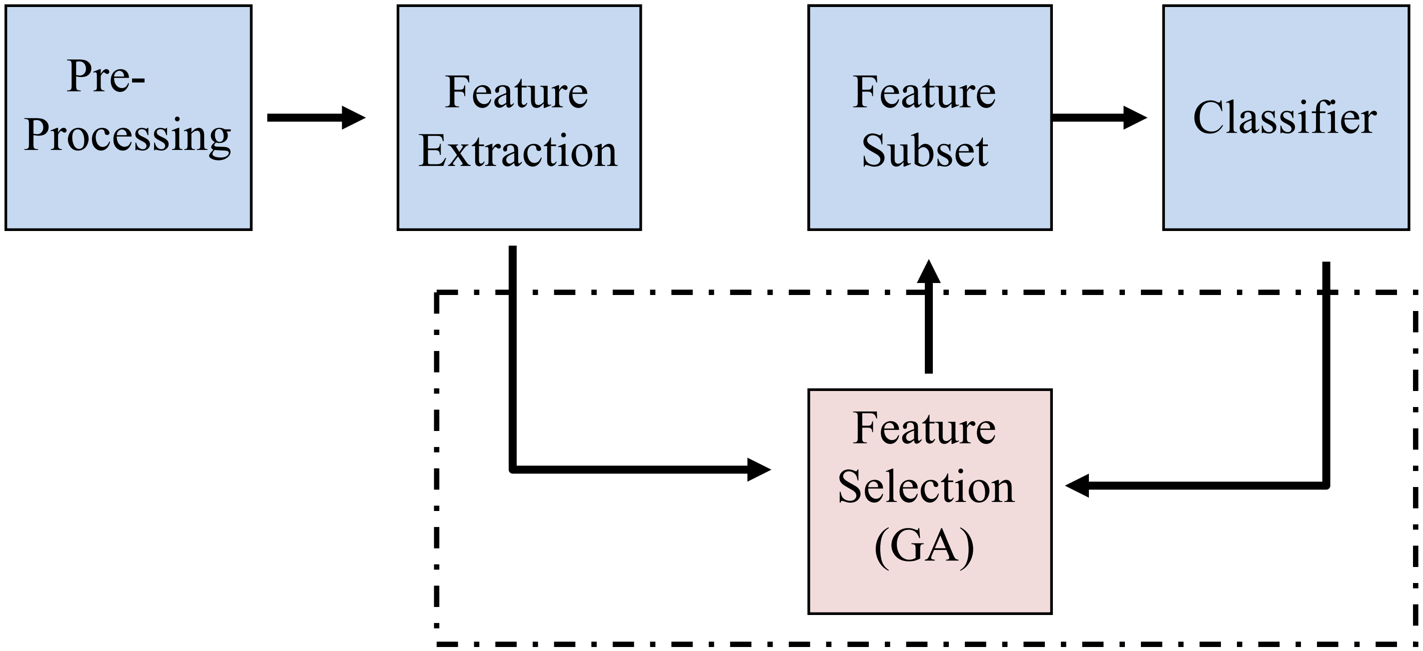


GA: steps

1. Initialize a population with randomly generated individuals
2. Evaluate fitness of each individual
3. *Selection* - Reproduce high fitness chromosomes in the new population, removing poor fitness chromosomes
4. *Crossover* – Construct new chromosomes
5. *Mutation* – Recover lost features
6. Repeat steps 2-6 until some stopping criterion is met (e.g. population fitness, number of iterations)

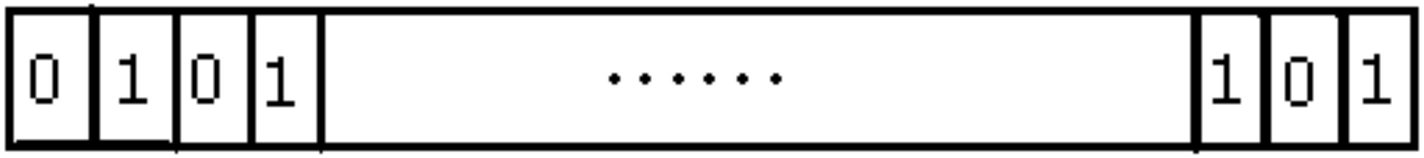
Feature Selection using GAs

- GAs provide a simple, general, and powerful framework for feature selection via a randomized search of the solutions.



Feature Selection using GAs

- **Binary encoding**: 1 means "choose feature" and 0 means "do not choose" feature.



- **Fitness evaluation** (to be maximized)

$$\text{Fitness} = w_1 \times \text{accuracy} + w_2 \times \#\text{zeros}$$

Classification accuracy using a validation set

Number of discarded features

$w_1 >> w_2$

Conclusive remarks

- Different type of approaches
  - Feature selection
    - Filters, Wrappers
    - Euristics, GAs
  - Dimensionality reduction
    - PCA, LDA, etc.
- Unclear how to tell in advance if reducing the number of features will work
  - Only known way is to check, but for very high dimensional data it helps most of the time
- How many features?
  - Just try! Perform cross-validation…
- There is not a "best" method… just methods that work well for your data