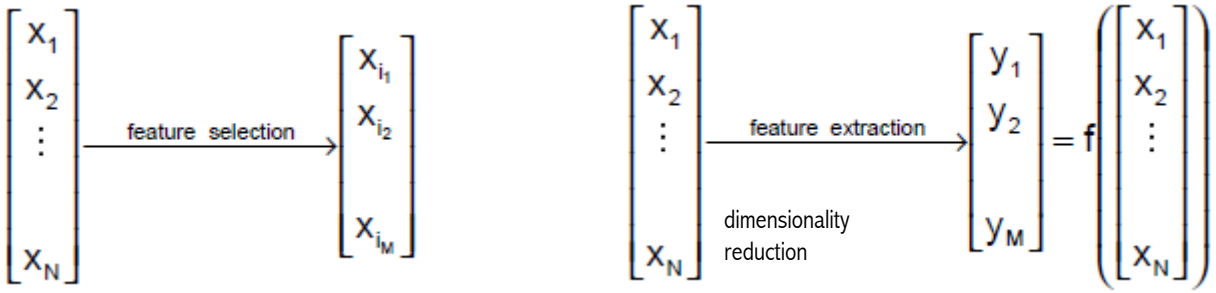
Feature Selection

Feature Selection

- Given a set of **n** features, the goal of **feature selection** is to select a subset of **d** features (**d** < **n**) in order to minimize the classification error.



- Why perform feature selection?
  - Data interpretation\knowledge discovery (insights into which factors which are most **representative** of your problem)
  - Curse of dimensionality (amount of data grows **exponentially** with # of features $O(2^N)$)

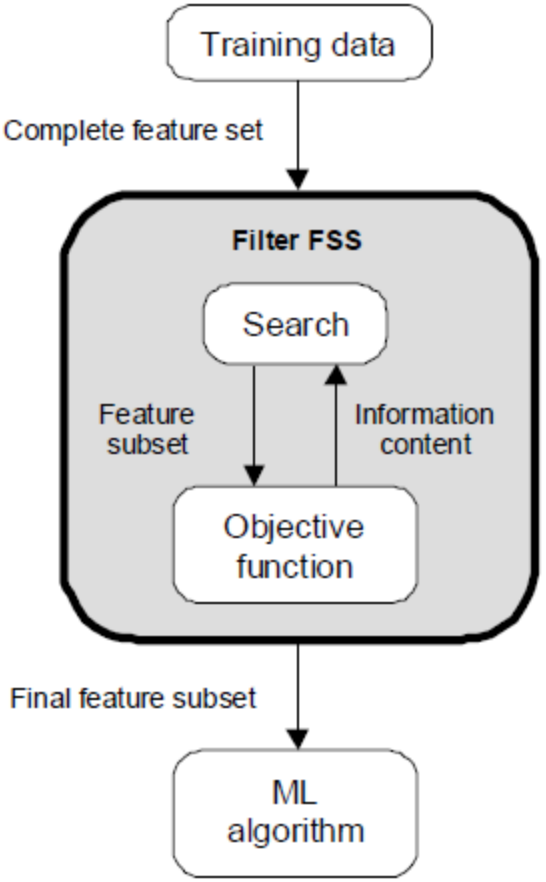
- Fundamentally different from dimensionality reduction (we will discuss next time) based on feature combinations (i.e., **feature extraction**).

Feature Selection vs. Dimensionality Reduction

- Feature Selection
  - When classifying novel patterns, only a small number of features need to be computed (i.e., faster classification).
  - The measurement units (length, weight, etc.) of the features are preserved.
- Dimensionality Reduction (next time)
  - When classifying novel patterns, all features need to be computed.
  - The measurement units (length, weight, etc.) of the features are lost.

Feature Selection 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.



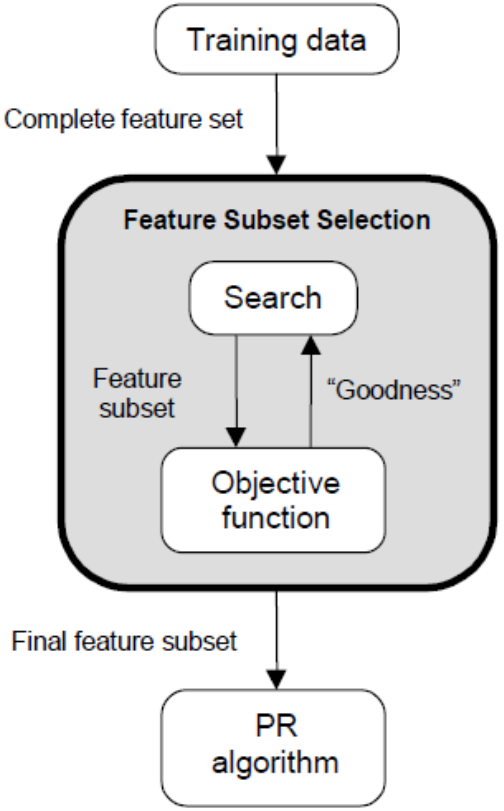
Feature Selection Steps (cont'd)

Search strategies

- Optimal
- Heuristic

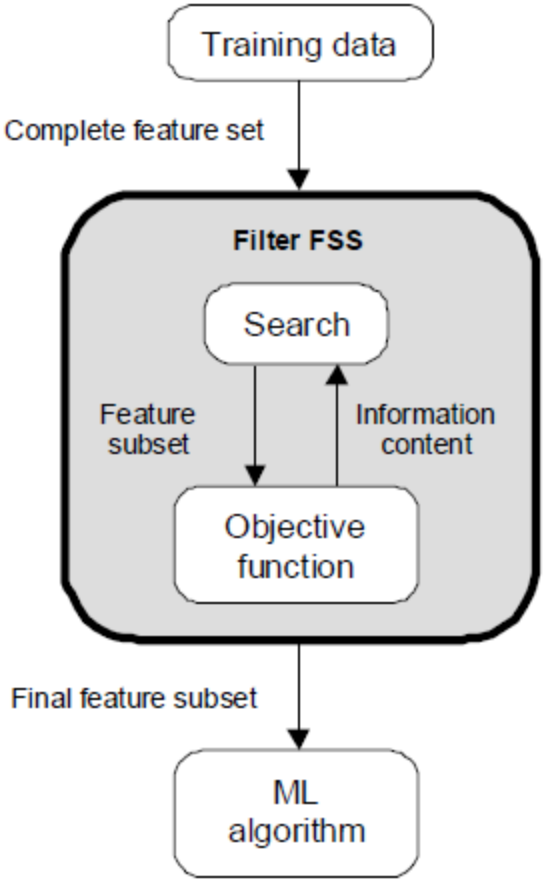
Evaluation strategies

- Filter methods
- Wrapper methods



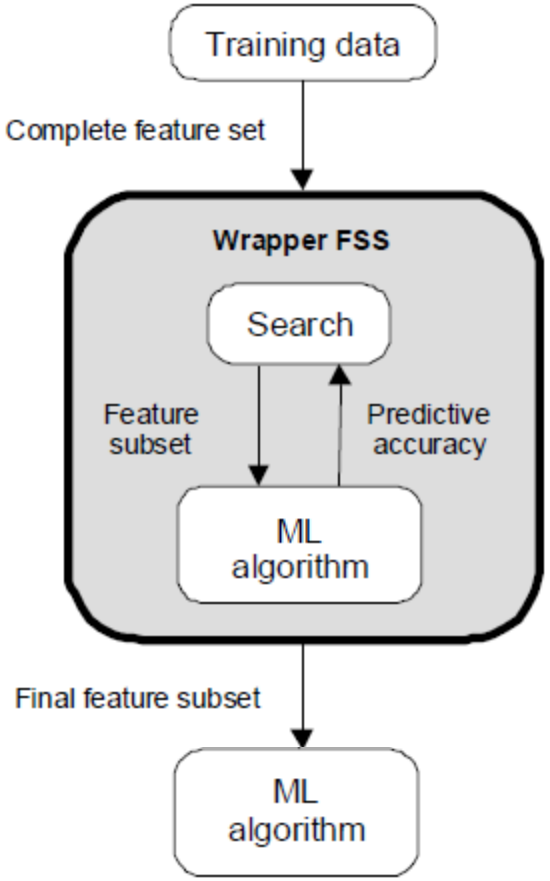
Evaluation Strategies

- 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.



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 (recognition rate on test data) by statistical resampling or cross-validation.



Filter vs. Wrapper Approaches

- **Filters**
  - Advantages
    - **Fast execution**: Filters generally involve a non-iterative computation on the dataset, which can execute much faster than a classifier training session
    - **Generality**: Since filters evaluate the intrinsic properties of the data, rather than their interactions with a particular classifier, their results exhibit more generality: the solution will be "good" for a larger family of classifiers
  - Disadvantages
    - **Tendency to select large subsets**: Since the filter objective functions are generally monotonic, the filter tends to select the full feature set as the optimal solution. This forces the user to select an arbitrary cutoff on the number of features to be selected

Filter vs Wrapper Approaches

- **Wrappers**
  - Advantages
    - **Accuracy**: wrappers generally achieve better recognition rates than filters since they are tuned to the specific interactions between the classifier and the dataset
    - **Ability to generalize**: wrappers have a mechanism to avoid overfitting, since they typically use cross-validation measures of predictive accuracy
  - Disadvantages
    - **Slow execution**: since the wrapper must train a classifier for each feature subset (or several classifiers if cross-validation is used), the method can become unfeasible for computationally intensive methods
    - **Lack of generality**: the solution lacks generality since it is tied to the bias of the classifier used in the evaluation function. The "optimal" feature subset will be specific to the classifier under consideration

Search Strategies

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

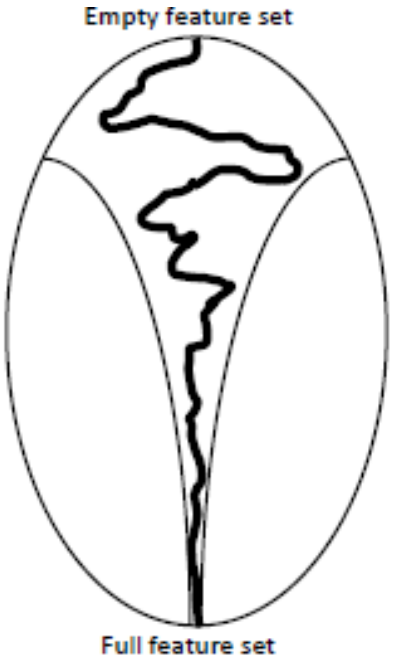
Naïve Search

- Sort the given n features in order of their probability of correct recognition.
- 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 some 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.

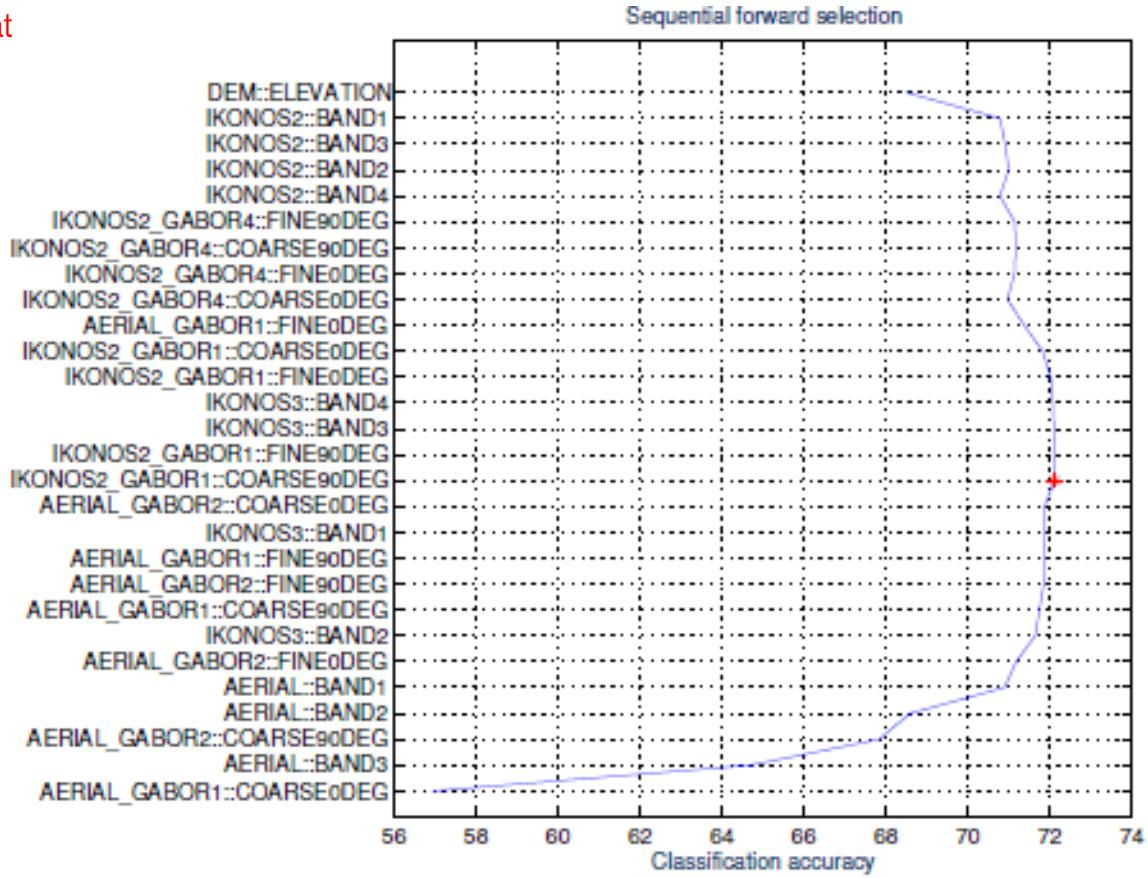


1. Start with the empty set $Y_0 = \{\emptyset\}$
2. Select the next best feature $x^+ = \arg\max_{x \notin Y_k} J(Y_k + x)$
3. Update $Y_{k+1} = Y_k + x^+;\ k = k + 1$
4. Go to 2

SFS performs best when the optimal subset is small.

Example

features added at each iteration

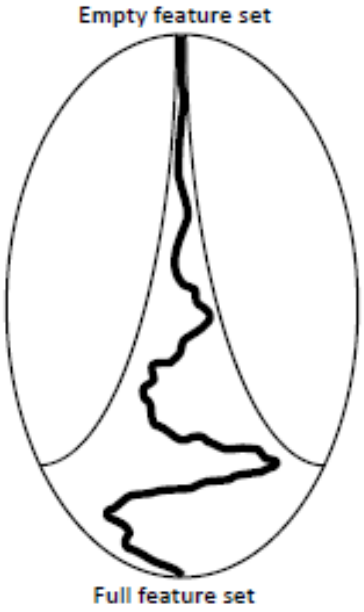


Results of sequential forward feature selection for classification of a satellite image using 28 features. x-axis shows the classification accuracy (%) and y-axis shows the features added at each iteration (the first iteration is at the bottom). The highest accuracy value is shown with a star.

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 of features are left.

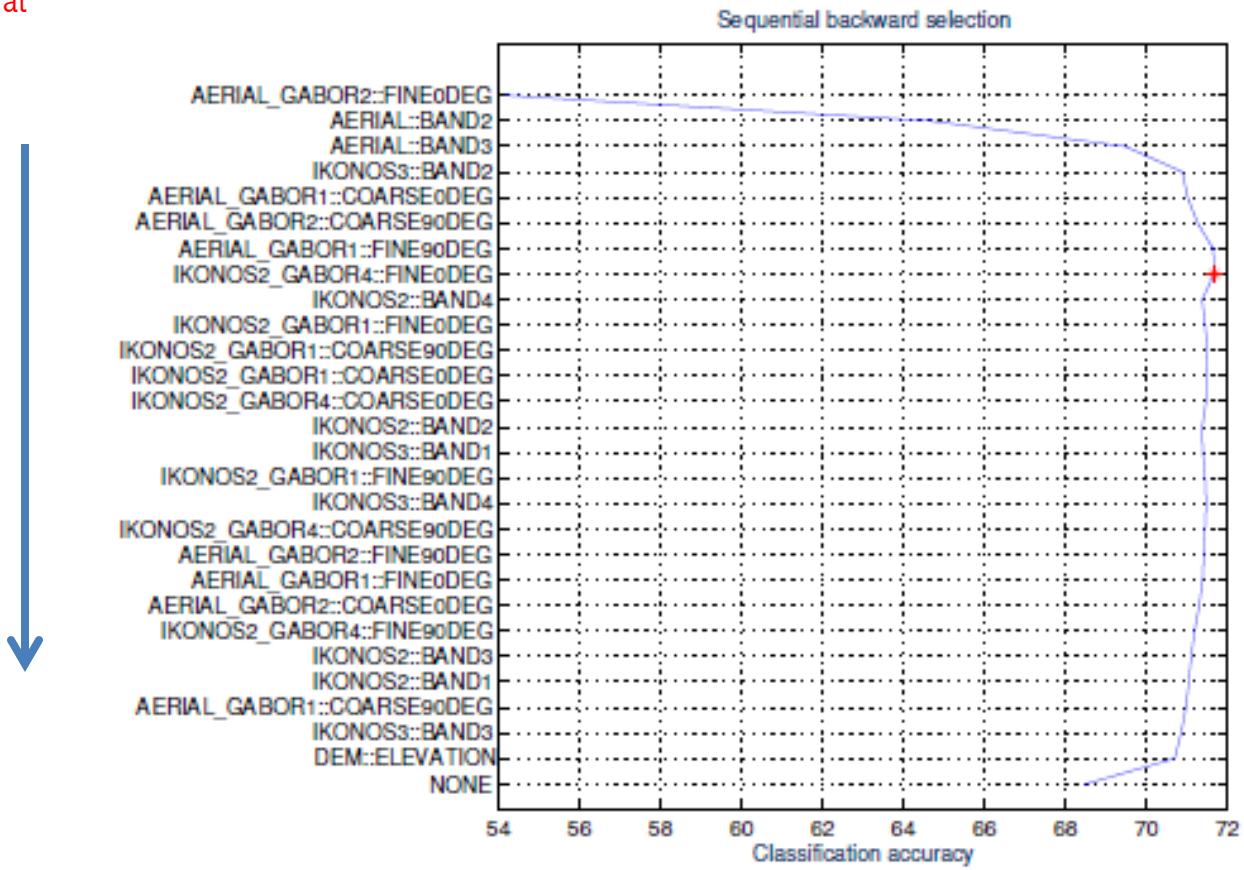


1. Start with the full set $Y_0 = X$
2. Remove the worst feature $x^- = \arg\max_{x \in Y_k} J(Y_k - x)$
3. Update $Y_{k+1} = Y_k - x^-;\ k = k + 1$
4. Go to 2

SBS performs best when the optimal subset is large.

Example

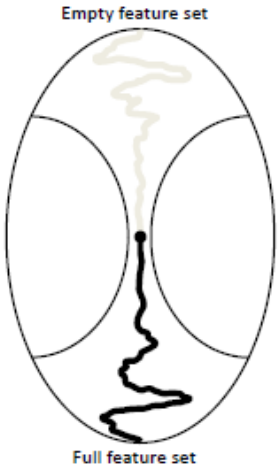
features removed at each iteration



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.

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

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

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

4. Remove the worst feature

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

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

5. Go to 3

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 reevaluate the usefulness of a feature after it has been discarded.

Feature Selection Summary

- Has two-fold advantage of providing some interpretation of the data and making the learning problem easier
- Finding global optimum impractical in most situations, rely on heuristics instead (greedy\random search)
- Filtering is fast and general but can pick a large # of features
- Wrapping considers model bias but is MUCH slower due to training multiple models