Redukcija dimenzija prostora obeležja

Milan M.Milosavljević

Zašto redukujemo ulaznu dimenzionalnost

- Redukcija vremenske kompleksnosti: manje računanja
- Redukcija prostorne kompleksnosti: manje parametara
- Smanjenje troškova observacija obeležja
- Jednostavniji modeli su robusniji na manjim obučavajućim skupovima
- Bolja interpretabilnost; jednostavnija objašnjenja
- Omogućuje bolju vizualizacjiu podataka ukoliko se predstave u 2 ili 3 dimenzije

Selekcija vs ekstrakcija obeležja

Selekcija obeležja:

- □ Izabrati $k<d$ važnih obeležja, ignorišući preostalih $d-k$

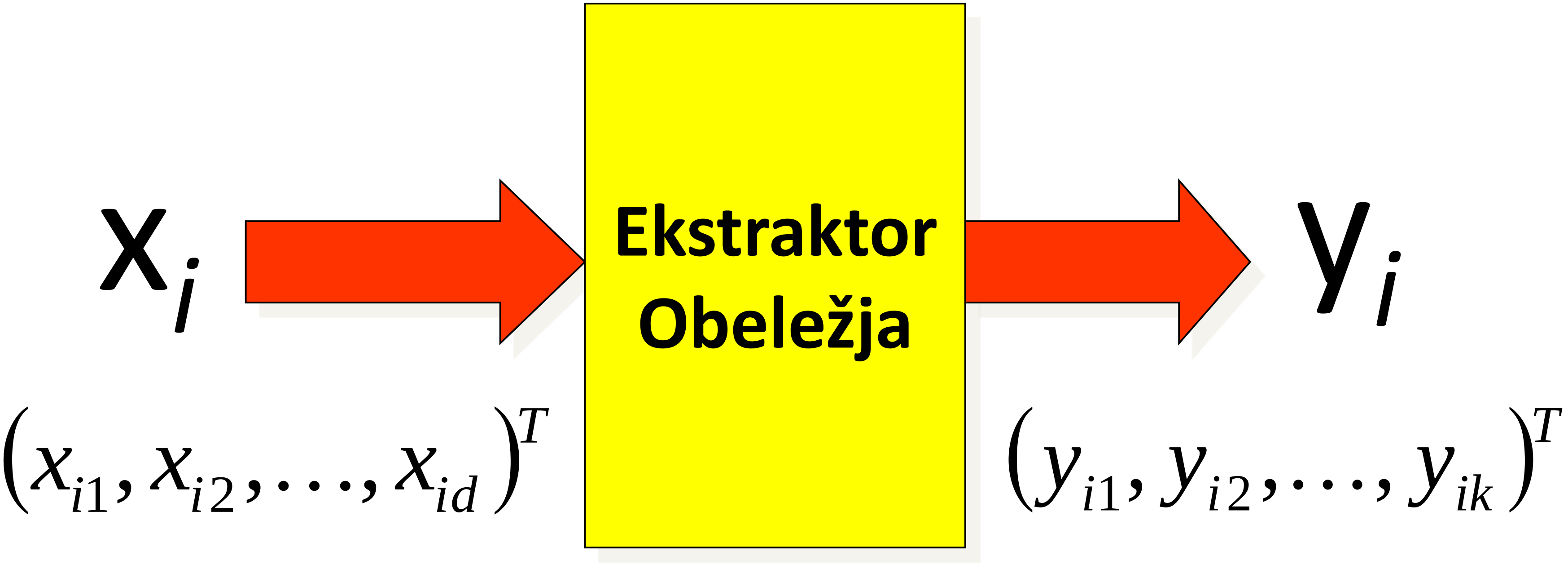
  Ovde su važni algoritmi za selekciju podskupova obeležja.

Ekstrakcija obeležja:

- □ Projektovati originalne dimenzije x_i , $i=1,...,d$ u novih $k<d$ dimenzija, y_j , $j=1,...,k$

- Optimalno preslikavanje bi bilo ono kod koga optimalno Bajesovsko odlučivanje u originalnom i novom projektovanom prostoru obeležja daje istu minimalnu Bajesovsku grešku odlučivanja.

Ekstrakcija obeležja



$\mathbf{x}_i$

Ekstraktor Obeležja

$\mathbf{y}_i$

$$\left(x_{i1}, x_{i2}, \ldots, x_{id}\right)^T$$

$$\left(y_{i1}, y_{i2}, \ldots, y_{ik}\right)^T$$

$K \leq d$, po pravilu

Neke važnije metode

- Principal Component Analysis (PCA)
  - or Karhunen-Loeve Expansion
- Project Pursuit
- Independent Component Analysis (ICA)
- Factor Analysis
- Discriminate Analysis

Linearni pristup

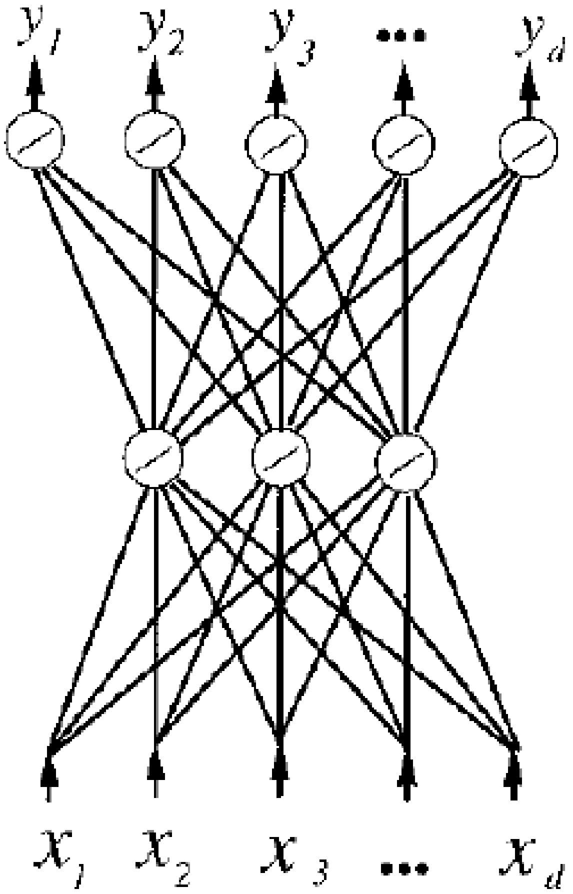
- Kernel PCA
- Multidimensional Scaling (MDS)

Nelinearni pristup

- Feed-Forward Neural Networks
- Self-Organizing Map

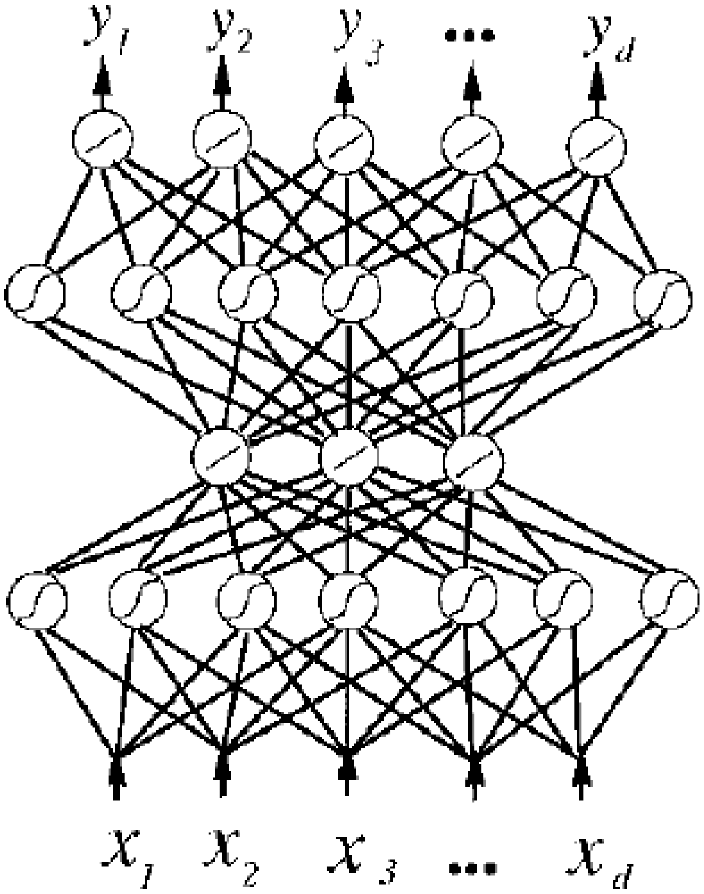
Neuronske mreže

Neuronske mreže



(a)

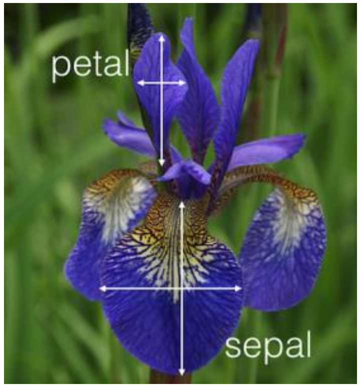
Linearna PCA



(b)

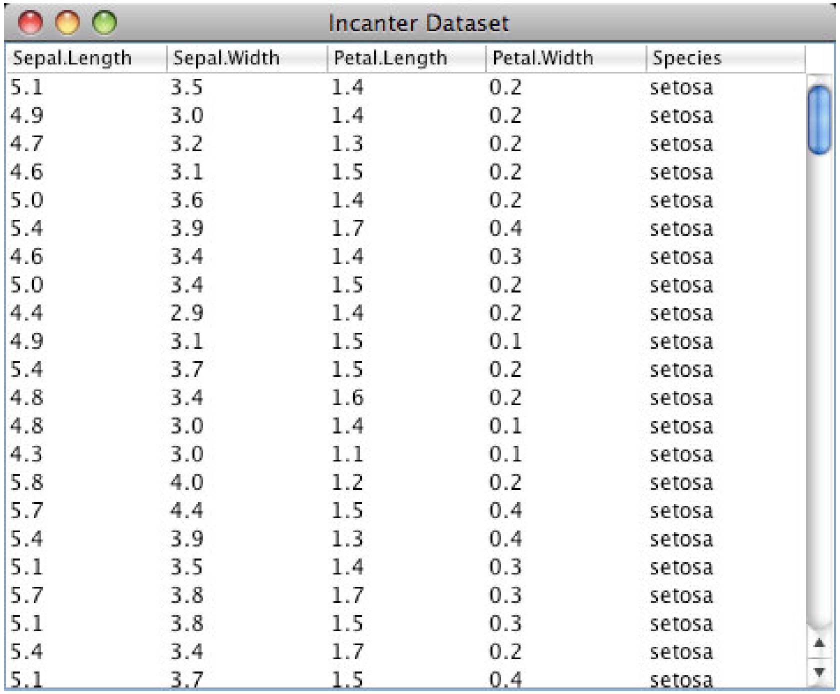
Nelinearna PCA

Primer: Iris podaci (Fišer 1936)



<u>Tri klase:</u>

1. Setosa
2. Versicolor
3. Virginica



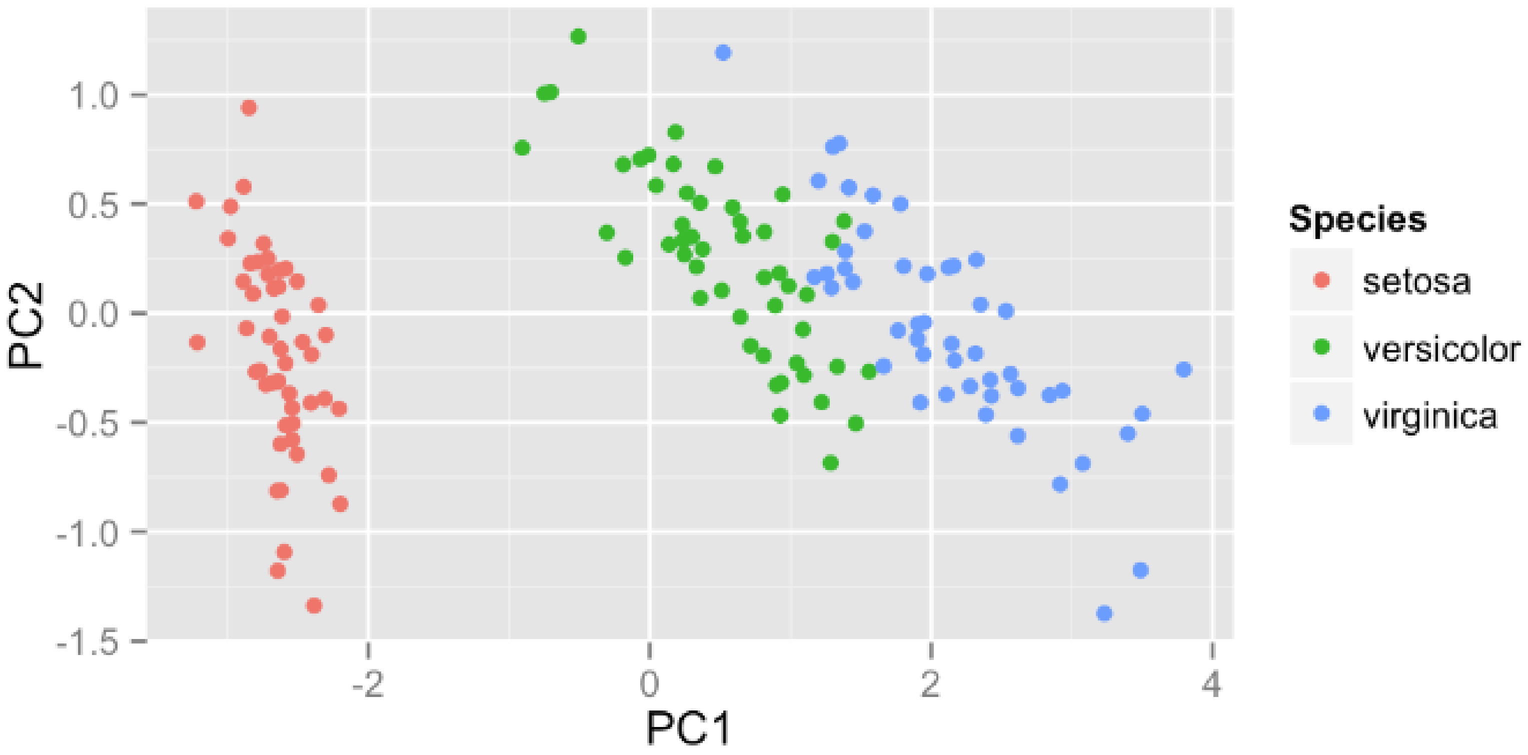
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3.0	1.4	0.1	setosa
4.3	3.0	1.1	0.1	setosa
5.8	4.0	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa
5.4	3.4	1.7	0.2	setosa
5.1	3.7	1.5	0.4	setosa

1

2

3

Primer: Iris PCA

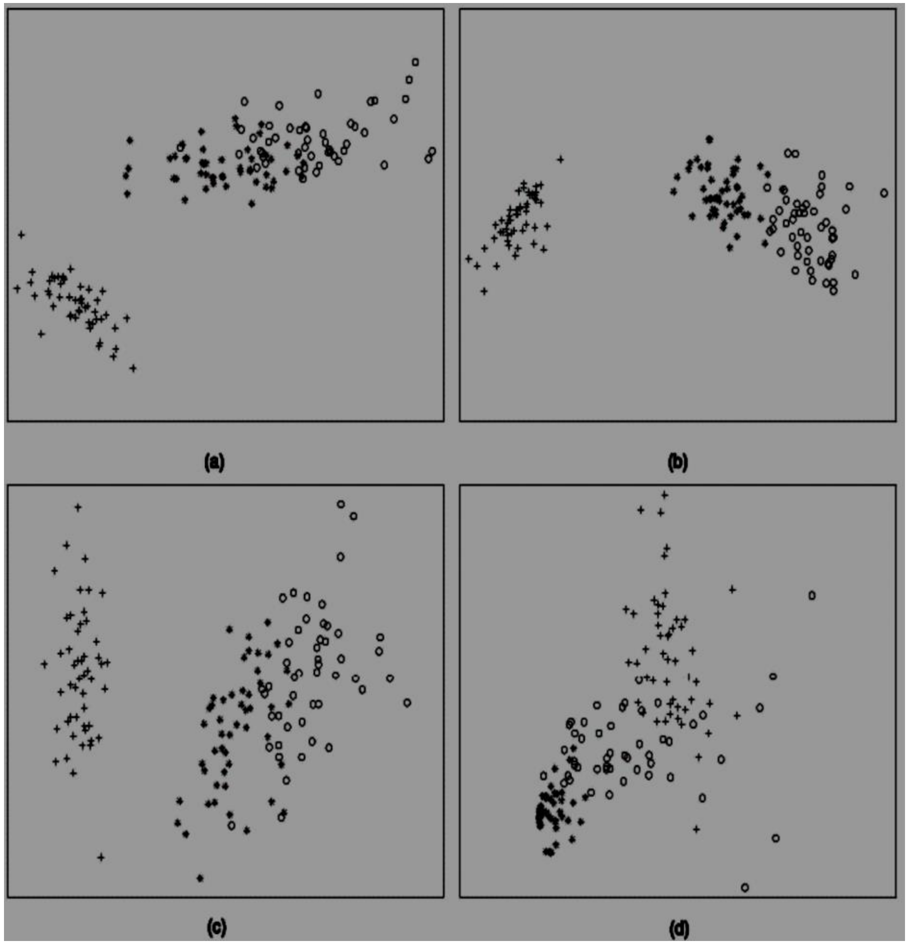


Primer

+ : Iris Setosa

* : Iris Versicolor

○ : Iris Virginica



PCA

Fišerovo preslikavanje

Samonovo preslikavanje

Kernel PCA sa polinomijalnim kernelom drugog reda

Višedimenziono skaliranje (Multidimensional Scaling – MDS)

- Neka su data medjusobna rastojanja izmedju *N* tačaka,

  d_{ij}, $i,j = 1,\dots,N$

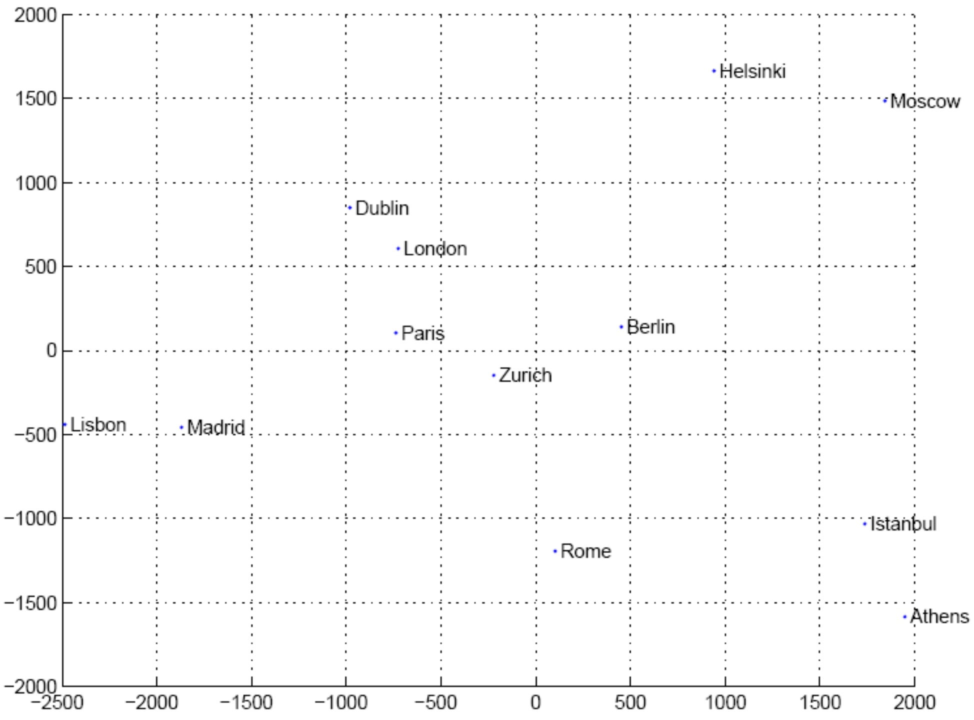
  rasporediti ove tačke u nižoj dimenziji tako da se rastojanja što više očuvaju.

- $z = g\,(x \mid \vartheta\,)$, Naći ϑ koje minimizira tzv. Sammon stress

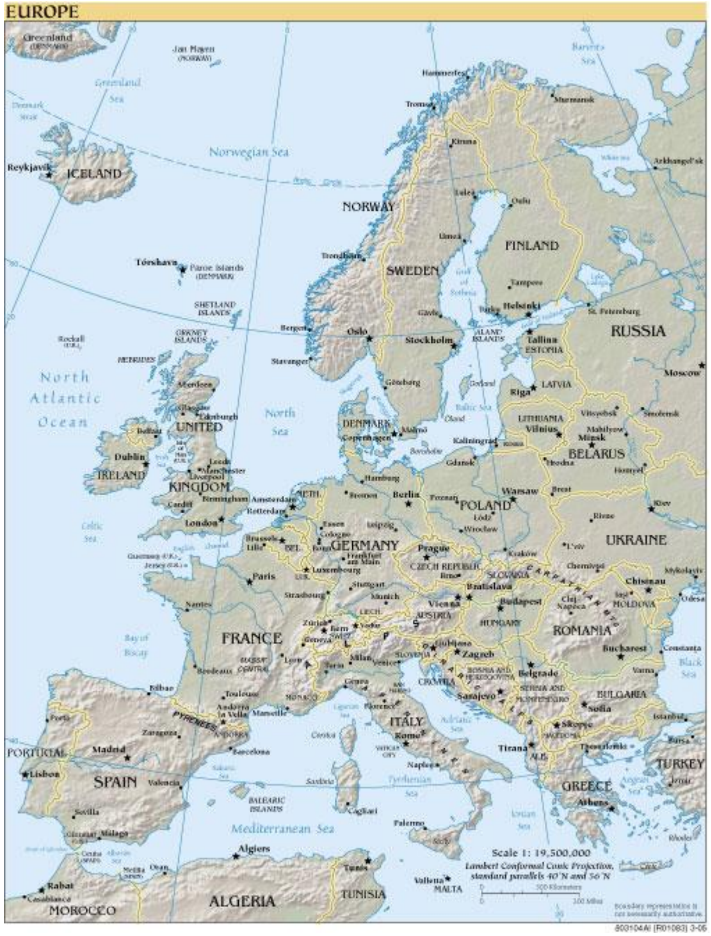
$$E(\theta \mid \mathcal{X}) = \sum_{r,s} \frac{\left(\left\|\mathbf{z}^r - \mathbf{z}^s\right\| - \left\|\mathbf{x}^r - \mathbf{x}^s\right\|\right)^2}{\left\|\mathbf{x}^r - \mathbf{x}^s\right\|^2}$$

$$= \sum_{r,s} \frac{\left(\left\|\mathbf{g}\left(\mathbf{x}^r \mid \theta\right) - \mathbf{g}\left(\mathbf{x}^s \mid \theta\right)\right\| - \left\|\mathbf{x}^r - \mathbf{x}^s\right\|\right)^2}{\left\|\mathbf{x}^r - \mathbf{x}^s\right\|^2}$$

Mapa Evrope dobijena pomoću MDS



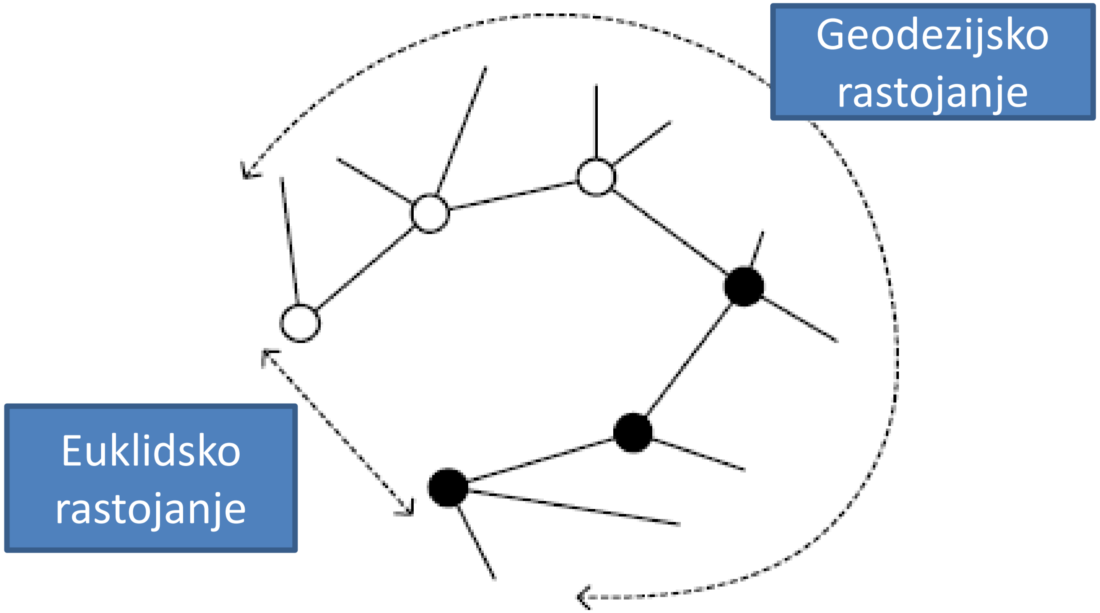
Ulaz u MDS je matrica medjusobnih rastojanja gradova Evrope



Map from CIA – The World Factbook: http://www.cia.gov/

Isomap

- Geodezijsko rastojanje je rastojanje mereno na višestrukosti na kojoj leže analizirani podaci, za razliku od običnog euklidovog rastojanja u ulaznom prostoru obeležja.



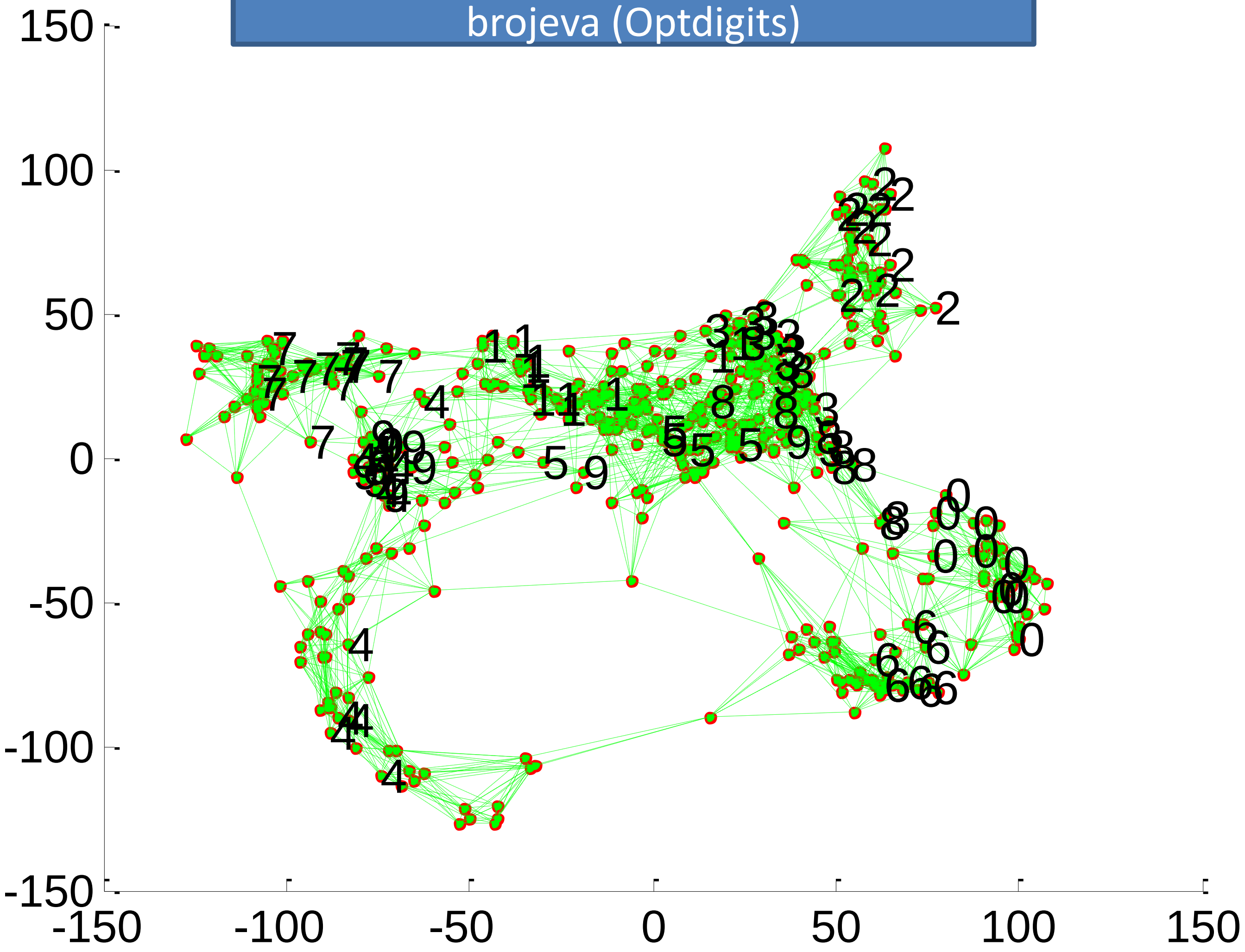
Isomap

- Instance r i s su povezane, ako važi

  $||\boldsymbol{x}^r-\boldsymbol{x}^s||<\varepsilon$ ili ako je $\boldsymbol{x}^s$ jedan od *k* suseda $\boldsymbol{x}^r$,

  dok je težina odgovarajuće grane jednaka $||\boldsymbol{x}^r-\boldsymbol{x}^s||$.
- Za dve istance r i s koje nisu povezane, rastojanje je jednako najkraćem putu izmedju njih
- Na ovaj način se formira *N*x*N* matrica rastojanja, a zatim se korišćrnjrm MDS nalazi preslikavanje u niže dimenzije

Isomap primenjen na bazu rukom pisanih brojeva (Optdigits)



Matlab source from http://web.mit.edu/cocosci/isomap/isomap.html

Lokalni linearno preslikavanje- (Locally Linear Embeding-LLE)

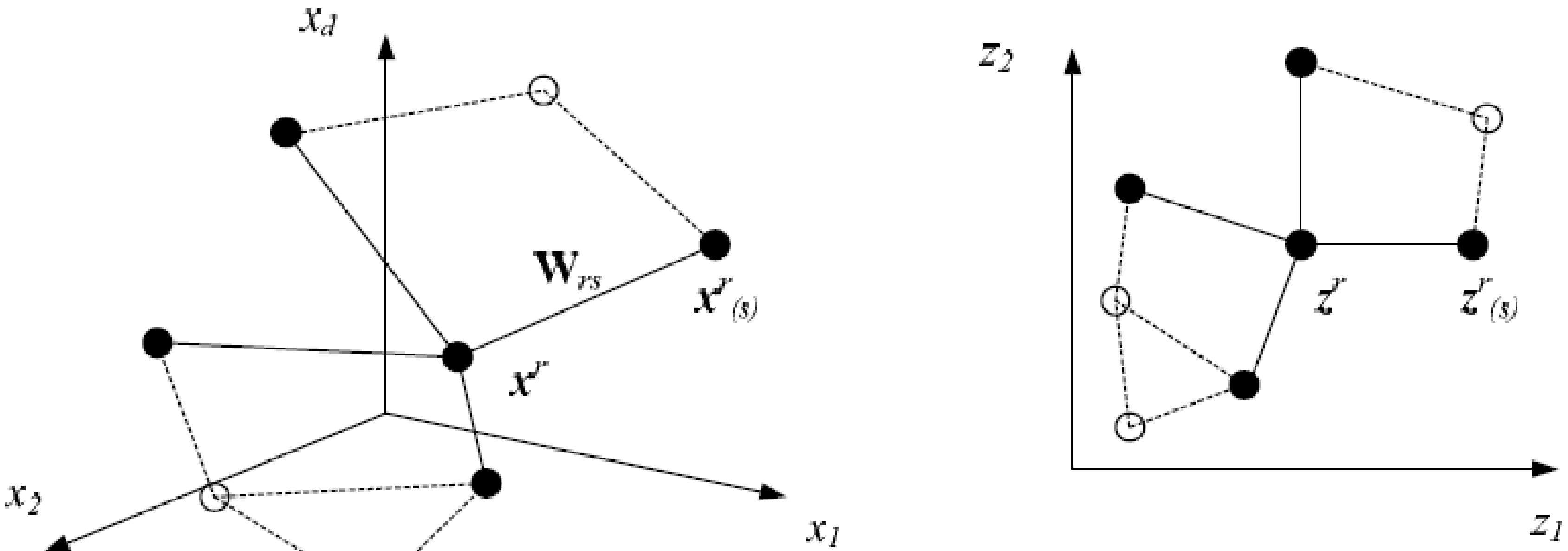
1. Za zadato $\boldsymbol{x}^r$ naći njegove najbliže susede $\boldsymbol{x}^s{}_{(r)}$
2. Naći $\mathbf{W}_{rs}$ koji minimiziraju

$$E(\mathbf{W}\,|\,X)=\sum_r \left\| \mathbf{x}^r - \sum_s \mathbf{W}_{rs}\mathbf{x}^s_{(r)} \right\|^2$$

3. Naći nove koordinate $\boldsymbol{z}^r$ koje minimiziraju

$$E(\mathbf{z}\,|\,\mathbf{W})=\sum_r \left\| z^r - \sum_s \mathbf{W}_{rs} z^s_{(r)} \right\|^2$$

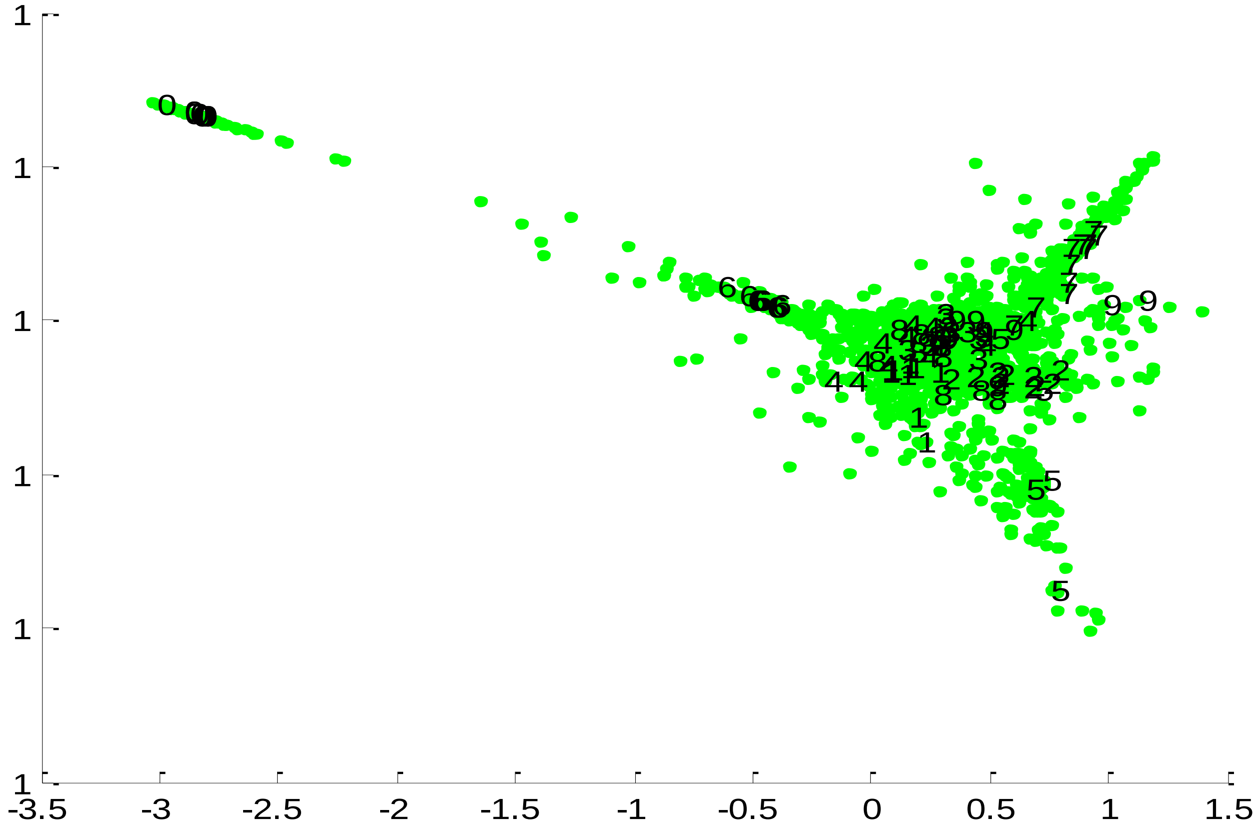
Lokalni linearno preslikavanje- (Locally Linear Embeding-LLE)



x space

z space

LLE na skupu Optdigits



Matlab source from http://www.cs.toronto.edu/~roweis/lle/code.html

Laplacian Eigenmaps

- Neka su *r* i *s* dve instance i B_{rs} je njihova sličnost. Cilj je naći $\boldsymbol{z}^r$ i $\boldsymbol{z}^s$ koje minimiziraju kriterijum

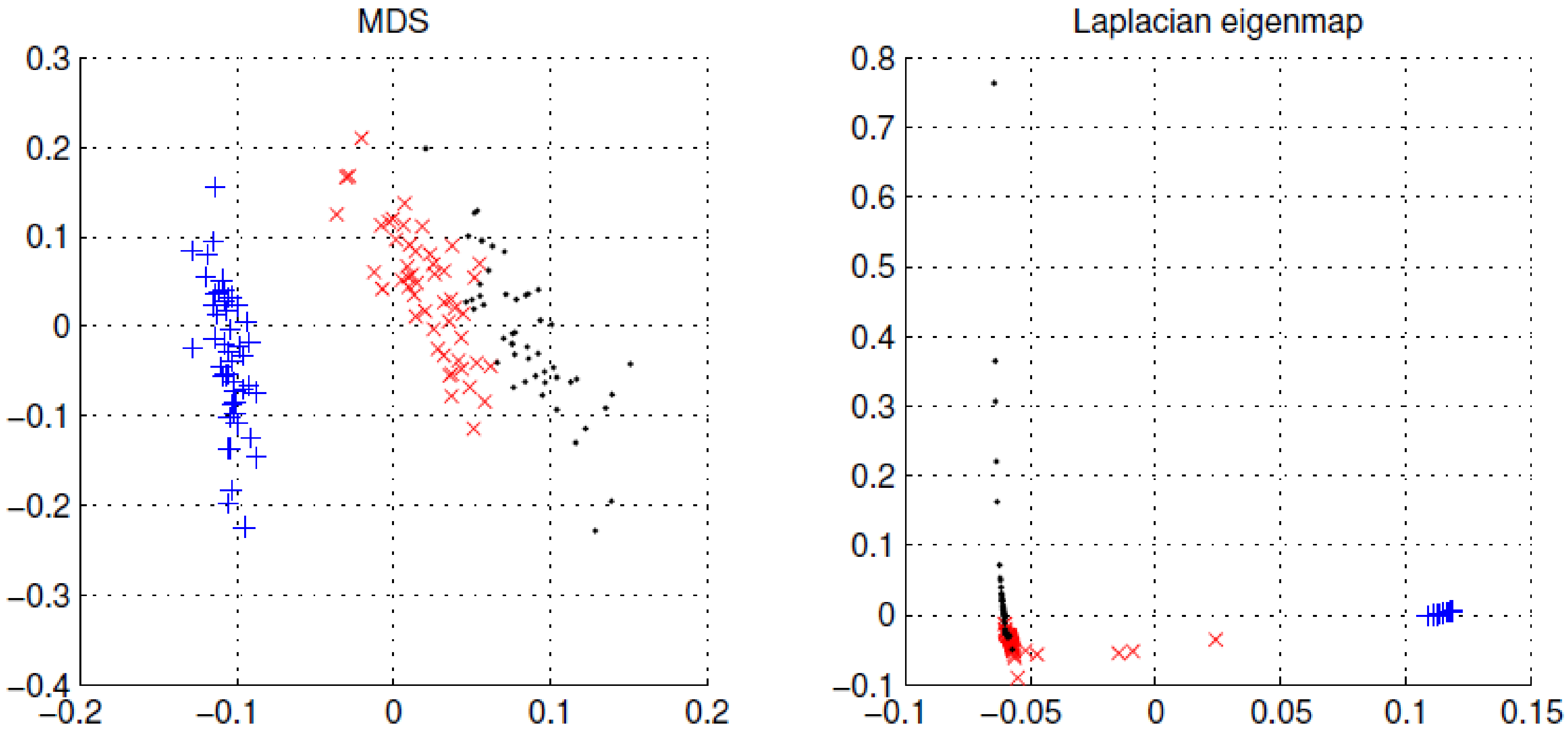
$$\min \sum_{r,s} \|\mathbf{z}^r - \mathbf{z}^s\|^2 B_{rs}$$

- B_{rs} može biti odredjeno u funkciji sličnosti u originalnom prostoru: 0 ako su $\boldsymbol{x}^r$ i $\boldsymbol{x}^s$ vrlo udaljeni, i u suprotnom

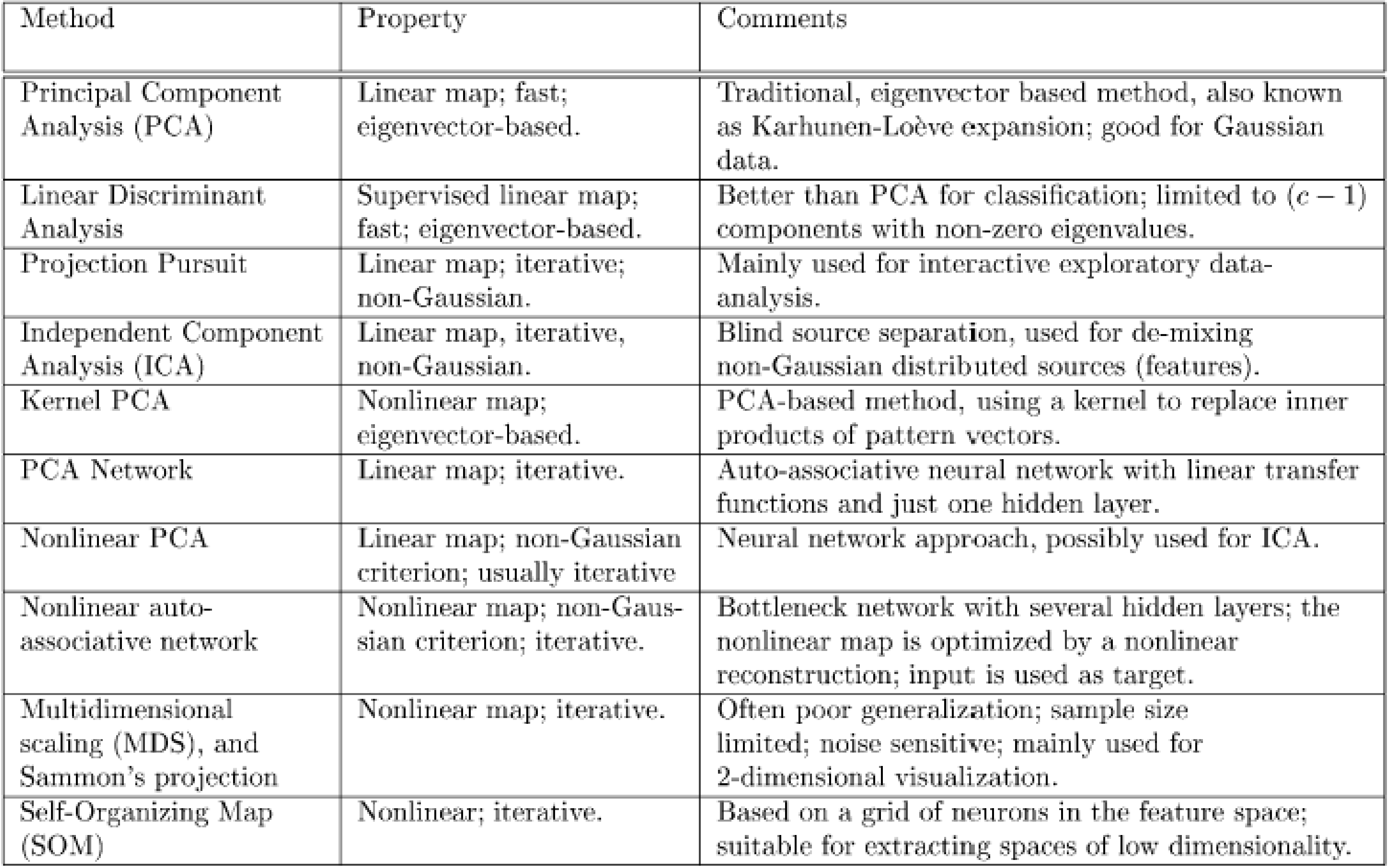
$$B_{rs} = \exp\left[-\frac{\|\mathbf{x}^r - \mathbf{x}^s\|^2}{2\sigma^2}\right]$$

- Ovim se može definisati Laplacian korespondirajućeg grafa, a postupak projektovanja originalnih obeležja daje $\boldsymbol{z}^r$.

Laplacian Eigenmaps na skupu Iris



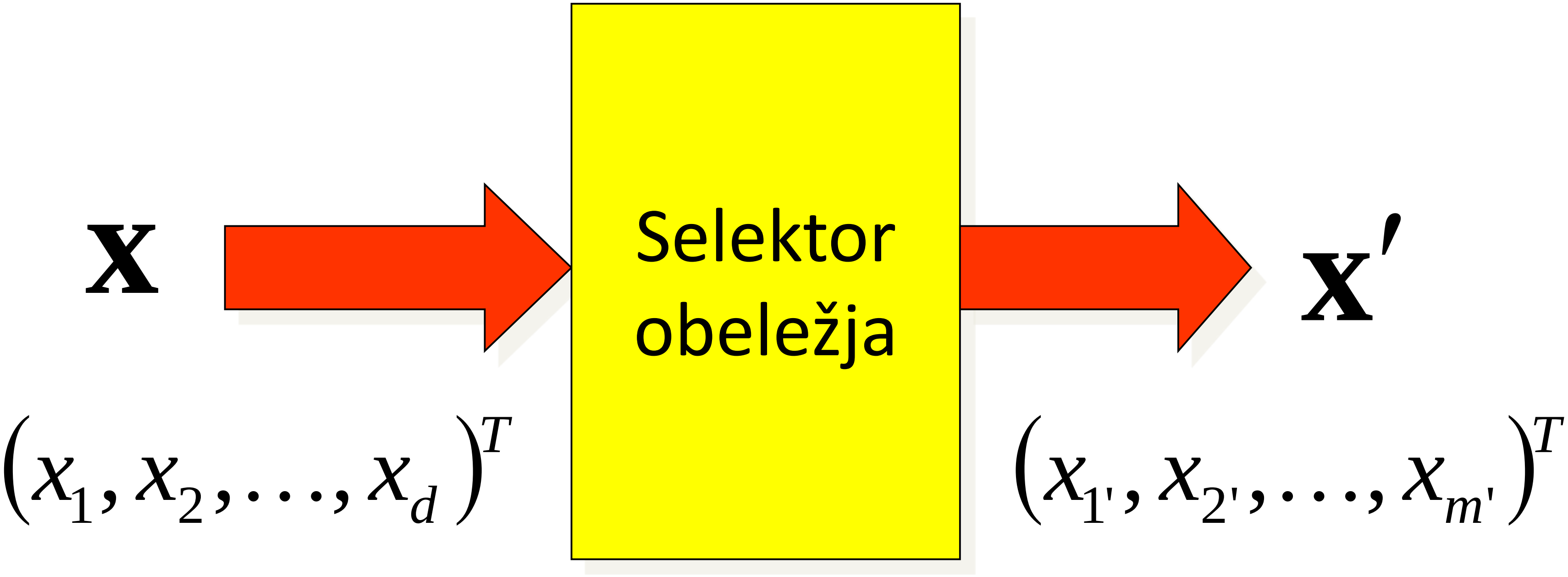
Rezime



Method	Property	Comments
Principal Component Analysis (PCA)	Linear map; fast; eigenvector-based.	Traditional, eigenvector based method, also known as Karhunen-Loève expansion; good for Gaussian data.
Linear Discriminant Analysis	Supervised linear map; fast; eigenvector-based.	Better than PCA for classification; limited to $(c-1)$ components with non-zero eigenvalues.
Projection Pursuit	Linear map; iterative; non-Gaussian.	Mainly used for interactive exploratory data-analysis.
Independent Component Analysis (ICA)	Linear map, iterative, non-Gaussian.	Blind source separation, used for de-mixing non-Gaussian distributed sources (features).
Kernel PCA	Nonlinear map; eigenvector-based.	PCA-based method, using a kernel to replace inner products of pattern vectors.
PCA Network	Linear map; iterative.	Auto-associative neural network with linear transfer functions and just one hidden layer.
Nonlinear PCA	Linear map; non-Gaussian criterion; usually iterative	Neural network approach, possibly used for ICA.
Nonlinear auto-associative network	Nonlinear map; non-Gaussian criterion; iterative.	Bottleneck network with several hidden layers; the nonlinear map is optimized by a nonlinear reconstruction; input is used as target.
Multidimensional scaling (MDS), and Sammon's projection	Nonlinear map; iterative.	Often poor generalization; sample size limited; noise sensitive; mainly used for 2-dimensional visualization.
Self-Organizing Map (SOM)	Nonlinear; iterative.	Based on a grid of neurons in the feature space; suitable for extracting spaces of low dimensionality.

Selekcija obeležja

# mogućih selekcija $\binom{d}{m}$



$\mathbf{x}$

Selektor obeležja

$\mathbf{x}'$

$(x_1, x_2, \ldots, x_d)^T$

$(x_{1'}, x_{2'}, \ldots, x_{m'})^T$

$m \leq d$, po pravilu

Selekcija obeležja – formulacija problema

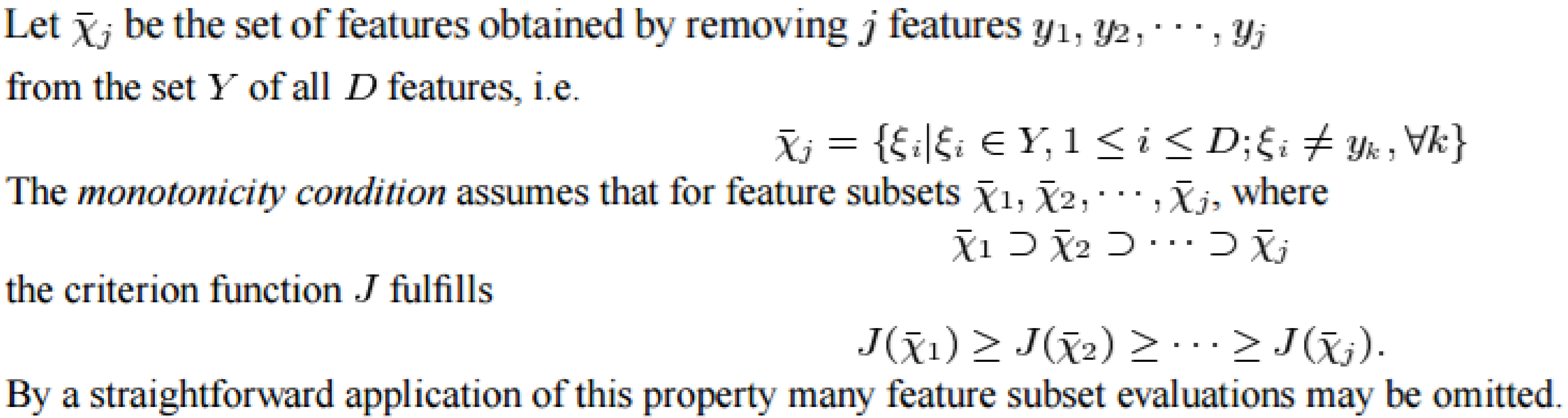
- Za zadati skup *d* obeležja, selektovati podskup dimenzije *m* koji daje najmanju grešku klasifikacije.
- Ne postoji nepotpuna procedura sekvencijalne selekcije obeležja, koja bi garantovala optimalni podskup obeležja. Drugim rečima, potpuna pretraga se ne može izbeći.

Optimalni metodi

- Potpuna pretraga
  - Evaluacija svih mogućih podskupova

- Algoritam grananja i ograničavanja (Branch-and-Bound)
  - Može se primeniti samo ako kriterijumska funkcija zadovoljava tzv. kriterijum monotonosti.

BB algoritam



Let $\bar{\chi}_j$ be the set of features obtained by removing j features $y_1, y_2, \cdots, y_j$ from the set Y of all D features, i.e.

$$\bar{\chi}_j = \{\xi_i | \xi_i \in Y, 1 \leq i \leq D; \xi_i \neq y_k, \forall k\}$$

The *monotonicity condition* assumes that for feature subsets $\bar{\chi}_1, \bar{\chi}_2, \cdots, \bar{\chi}_j$, where

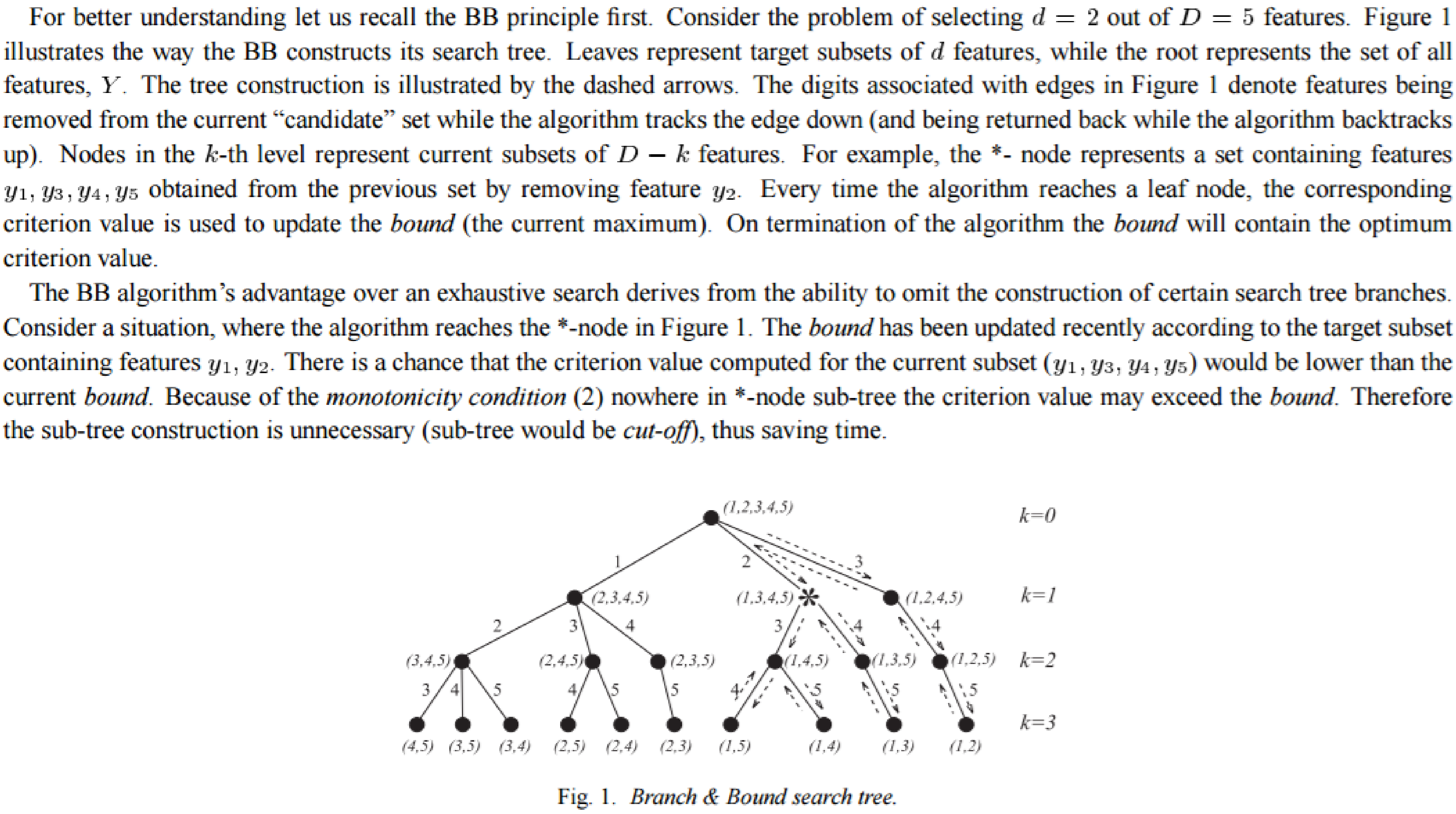
$$\bar{\chi}_1 \supset \bar{\chi}_2 \supset \cdots \supset \bar{\chi}_j$$

the criterion function J fulfills

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

By a straightforward application of this property many feature subset evaluations may be omitted.

BB algoritam - primer



For better understanding let us recall the BB principle first. Consider the problem of selecting $d = 2$ out of $D = 5$ features. Figure 1 illustrates the way the BB constructs its search tree. Leaves represent target subsets of d features, while the root represents the set of all features, Y. The tree construction is illustrated by the dashed arrows. The digits associated with edges in Figure 1 denote features being removed from the current "candidate" set while the algorithm tracks the edge down (and being returned back while the algorithm backtracks up). Nodes in the k-th level represent current subsets of $D - k$ features. For example, the *- node represents a set containing features y_1, y_3, y_4, y_5 obtained from the previous set by removing feature y_2. Every time the algorithm reaches a leaf node, the corresponding criterion value is used to update the *bound* (the current maximum). On termination of the algorithm the *bound* will contain the optimum criterion value.

The BB algorithm's advantage over an exhaustive search derives from the ability to omit the construction of certain search tree branches. Consider a situation, where the algorithm reaches the *-node in Figure 1. The *bound* has been updated recently according to the target subset containing features y_1, y_2. There is a chance that the criterion value computed for the current subset (y_1, y_3, y_4, y_5) would be lower than the current *bound*. Because of the *monotonicity condition* (2) nowhere in *-node sub-tree the criterion value may exceed the *bound*. Therefore the sub-tree construction is unnecessary (sub-tree would be *cut-off*), thus saving time.

Fig. 1. *Branch & Bound search tree.*

Suboptimalne metode

- Najbolja Individualna obeležja

- Sekvencijalna selekcija unapred (Sequential Forward Selection - SFS)
- Sekvencijalna selekcija unazad (Sequential Backward Selection - SBS)
- Selekcija „dodaj l i odbaci r“ (“Plus *l*-take away *r*” Selection)
- Sequential Forward Floating Search and Sequential Backward Floating Search

Naivna metoda sekvencijalnog izbora najinformativnijih obeležja

Naivan algoritam izbora M najboljih obeležja:

1. Evaluirati diskriminatornu informativnost svakog obeležja pojedinačno.
2. Sortirati pojedinačna obeležja po diskriminatornoj informativnosti
3. Izabrati prvih M sa spiska

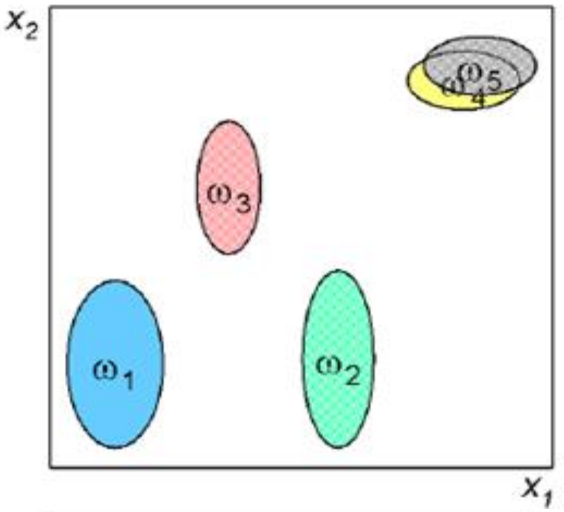
- Da li je ovo zaista i skup od M najboljih obeležja?
- Odgovor je ne, budući da u rangiranju nije uzeta u obzir medjusobna medjuzavisnost obeležja.

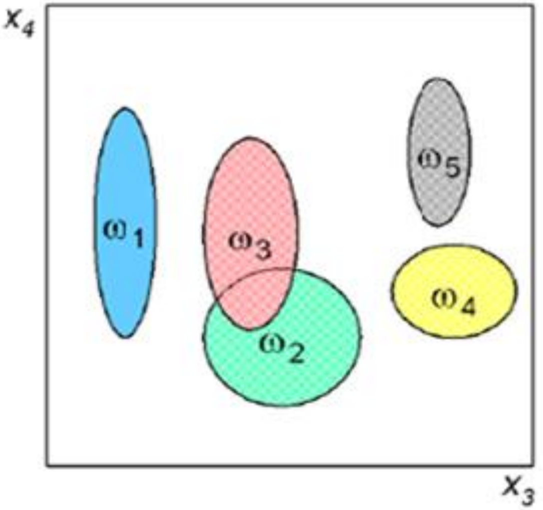
Kontra primer

- Razmotrimo 4 D prostor obeležja sa 5 klasa
- Svaki algoritam sa razumnom kriterijumskom funkcijom će izvršiti sledeće rangiranje

$$J(x_1) > J(x_2) \approx J(x_3) > J(x_4)$$

- x_1 je najbolje obeležje: ono separiše $\omega_1, \omega_2, \omega_3$ i $\{\omega_4, \omega_5\}$
- x_2 i x_3 su ekvivalentni i separišu klase u tri grupe
- x_4 je najgore obeležje, ono separiše samo ω_4 od ω_5





Kontra primer

- Optimalan podskup obeležja se ispostabvlja da je skup $\{x_1, x_4\}$ budući da x_4 jedino daje informaciju koja je potrebna x_1: diskriminacija izmedju ω_4 i ω_5
- Ako bi smo rangirali obeležja individualno na osnovu $J(x_k)$ sigurno bi izabrali x_1 a zatim x_2 ili x_3, što bi klase ω_4 i ω_5 ostavilo nerazdvojene.
- U suštini ova jednostavna strategija nije dobra zato što ne razmatra obeležja sa komplementarnim svojstvima.

Sekvencijalna selekcija unapred

FSS requires

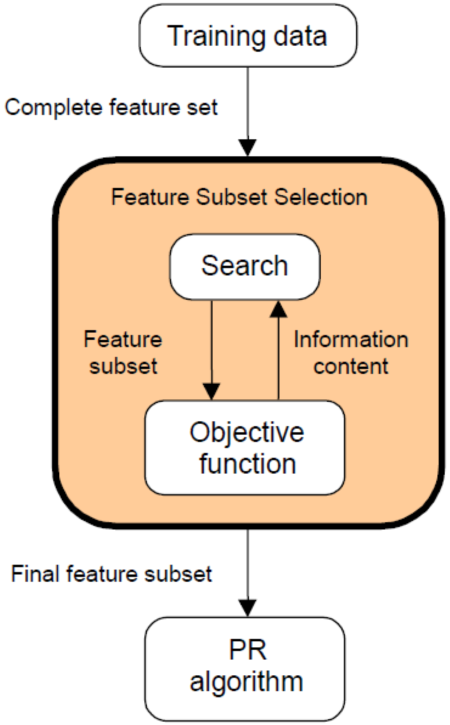
- A search strategy to select candidate subsets
- An objective function to evaluate these candidates

Search strategy

- Exhaustive evaluation of feature subsets involves $\binom{N}{M}$ combinations for a fixed value of M, and 2^N combinations if M must be optimized as well
  - This number of combinations is unfeasible, even for moderate values of M and N, so a search procedure must be used in practice
  - For example, exhaustive evaluation of 10 out of 20 features involves 184,756 feature subsets; exhaustive evaluation of 10 out of 100 involves more than 10^{13} feature subsets [Devijver and Kittler, 1982]
- A search strategy is therefore needed to direct the FSS process as it explores the space of all possible combination of features

Objective function

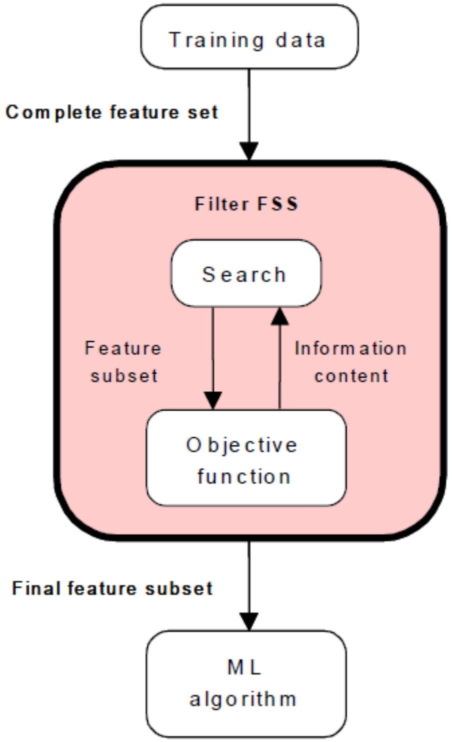
- The objective function evaluates candidate subsets and returns a measure of their "goodness", a feedback signal used by the search strategy to select new candidates

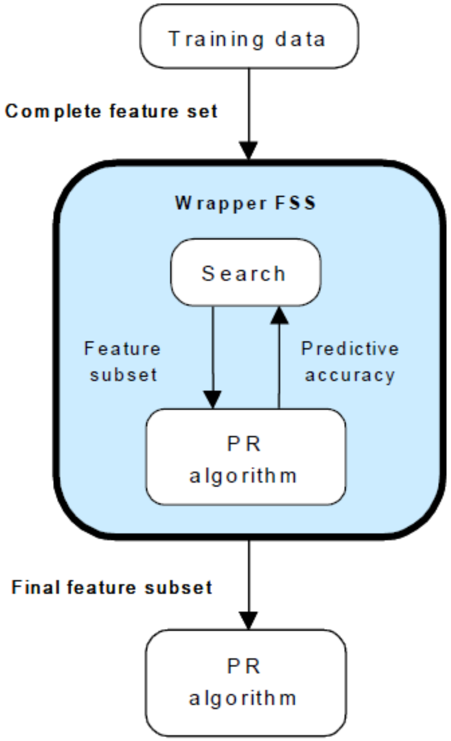


Kriterijumska funkcija

Objective functions are divided in two groups

- **Filters**: evaluate subsets by their information content, e.g., interclass distance, statistical dependence or information-theoretic measures
- **Wrappers**: use a classifier to evaluate subsets by their predictive accuracy (on test data) by statistical resampling or cross-validation





Filters vs. wrappers

Filters

- **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
- **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

Wrappers

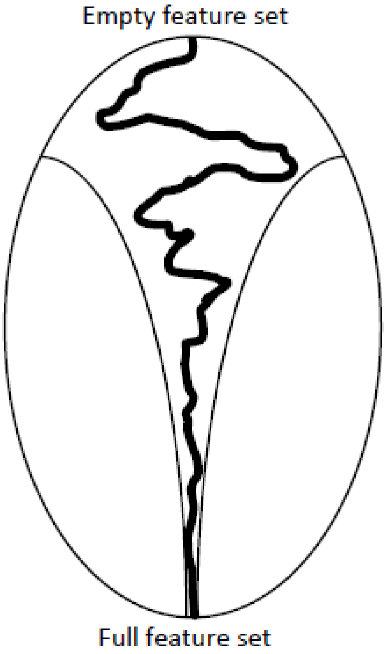
- **Accuracy (+)**: wrappers generally achieve better recognition rates than filters since they are tuned to the specific interactions between the classifier and the dataset
- **Ability to generalize (+)**: wrappers have a mechanism to avoid overfitting, since they typically use cross-validation measures of predictive accuracy
- **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

Sequential forward selection (SFS)

SFS is the simplest greedy search algorithm

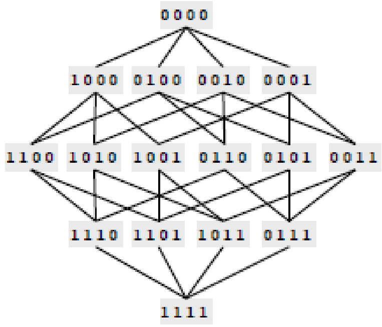
- Starting from the empty set, sequentially add the feature x^+ that maximizes $J(Y_k + x^+)$ when combined with the features Y_k that have already been selected

```
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
```



Notes

- SFS performs best when the optimal subset is small
  - When the search is near the empty set, a large number of states can be potentially evaluated
  - Towards the full set, the region examined by SFS is narrower since most features have already been selected
- The search space is drawn like an ellipse to emphasize the fact that there are fewer states towards the full or empty sets
  - The main disadvantage of SFS is that it is unable to remove features that become obsolete after the addition of other features

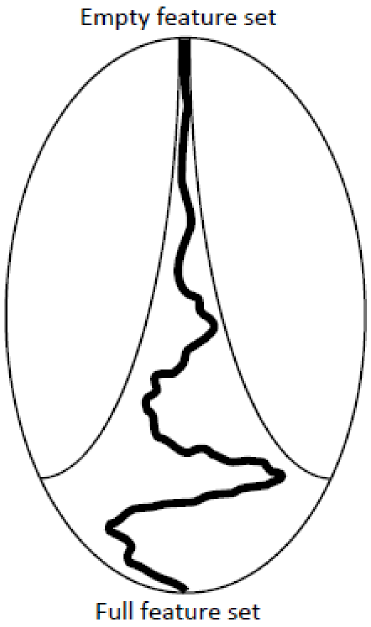


Sequential backward selection (SBS)

SBS works in the opposite direction of SFS

- Starting from the full set, sequentially remove the feature x^- that least reduces the value of the objective function $J(Y - x^-)$
  - Removing a feature may actually increase the objective function $J(Y_k - x^-) > J(Y_k)$; such functions are said to be non-monotonic (more on this when we cover Branch and Bound)

```
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
```



Notes

- SBS works best when the optimal feature subset is large, since SBS spends most of its time visiting large subsets
- The main limitation of SBS is its inability to reevaluate the usefulness of a feature after it has been discarded

Plus-L minus-R selection (LRS)

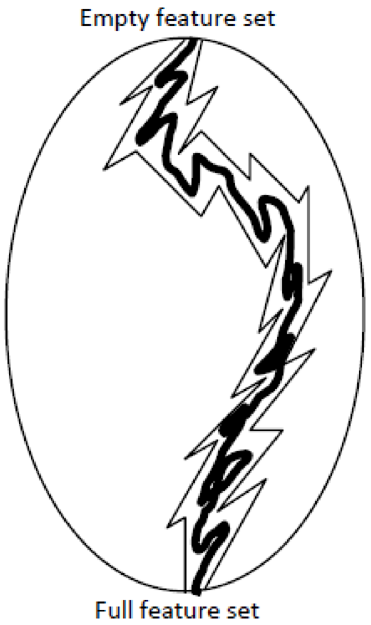
A generalization of SFS and SBS

- If L>R, LRS starts from the empty set and repeatedly adds L features and removes R features
- If L<R, LRS starts from the full set and repeatedly removes R features followed by L additions

```
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
```

Notes

- LRS attempts to compensate for the weaknesses of SFS and SBS with some backtracking capabilities
- Its main limitation is the lack of a theory to help predict the optimal values of L and R

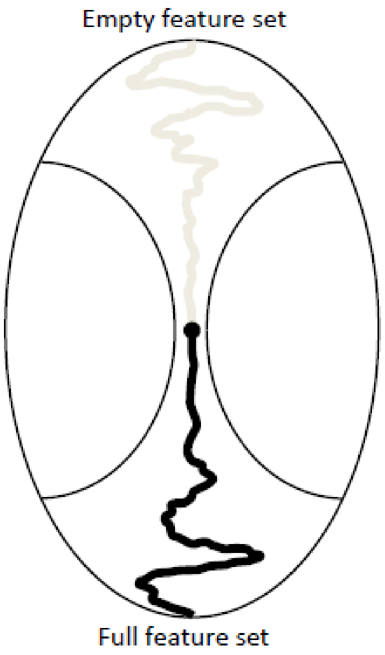


Bidirectional Search (BDS)

BDS is a parallel implementation of SFS and SBS

- 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 selected by SFS

```
1. Start SFS with $Y_F = \{\emptyset\}$
2. Start SBS with $Y_B = X$
3. Select the best feature
        $x^+ = \arg\max_{\substack{x \notin Y_{F_k} \\ x \in \mathrm{F}_{B_k}}} J\left(Y_{F_k} + x\right)$
        $Y_{F_{k+1}} = Y_{F_k} + x^+$
4. Remove the worst feature
        $x^- = \arg\max_{\substack{x \in Y_{B_k} \\ x \notin Y_{F_{k+1}}}} J\left(Y_{B_k} - x\right)$
        $Y_{B_{k+1}} = Y_{B_k} - x^-;\ k = k + 1$
5. Go to 2
```



Filter types

Distance or separability measures

- These methods measure class separability using metrics such as
  - Distance between classes: Euclidean, Mahalanobis, etc.
  - Determinant of $S_W^{-1} S_B$ (LDA eigenvalues)

Correlation and information-theoretic measures

- These methods are based on the rationale that good feature subsets contain features highly correlated with (predictive of) the class, yet uncorrelated with (not predictive of) each other
- **Linear relation measures**
  - Linear relationship between variables can be measured using the correlation coefficient

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

  - Where ρ_{ic} is the correlation coefficient between feature i and the class label and ρ_{ij} is the correlation coefficient between features i and j

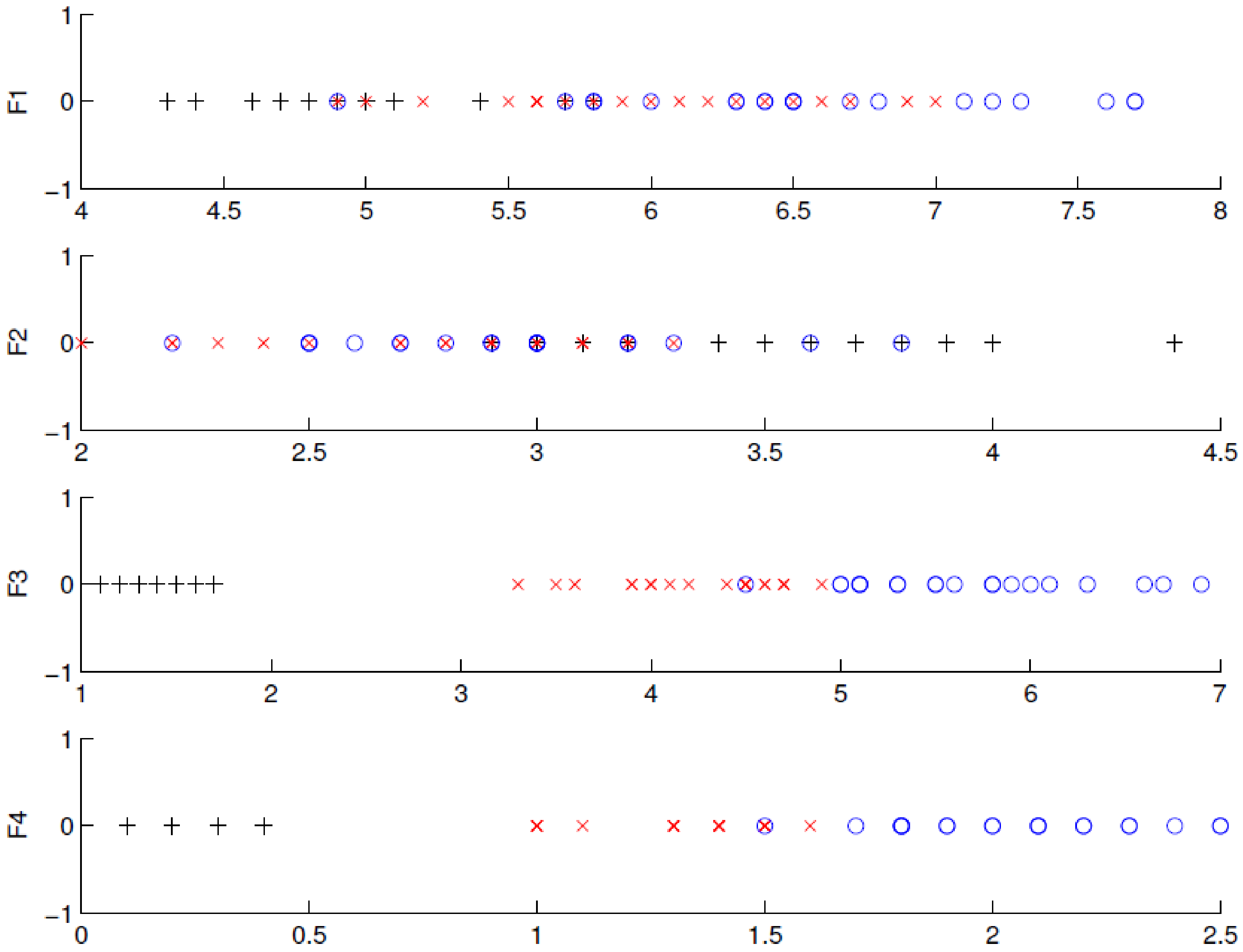
- **Non-linear relation measures**
  - Correlation is only capable of measuring linear dependence
  - A more powerful measure is the mutual information $I(Y_k; C)$

$$J(Y_M) = I(Y_M; C) = H(C) - H(C|Y_M) = \sum_{c=1}^{C} \int_{Y_M} p(Y_M, \omega_c) log \frac{p(Y_M, \omega_c)}{p(Y_M)P(\omega_c)} dx$$

  - The mutual information between the feature vector and the class label $I(Y_M; C)$ measures the amount by which the uncertainty in the class $H(C)$ is decreased by knowledge of the feature vector $H(C|Y_M)$, where $H(\cdot)$ is the entropy function
  - Note that mutual information requires the computation of the multivariate densities $p(Y_M)$ and $p(Y_M, \omega_c)$, which is ill-posed for high-dimensional spaces
  - In practice [Battiti, 1994], mutual information is replaced by a heuristic such as

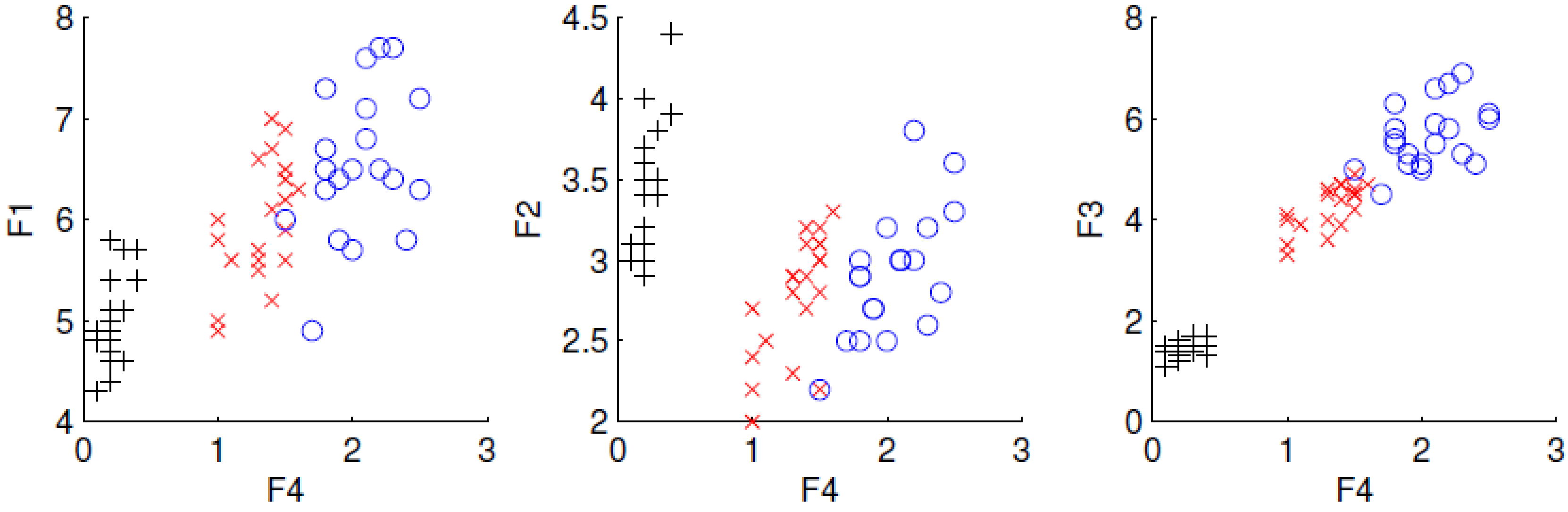
$$J(Y_M) = \sum_{m=1}^{M} I(x_{i_m}; C) - \beta \sum_{m=1}^{M} \sum_{n=m+1}^{M} I(x_{i_m}; x_{i_n})$$

Iris: Jedno obeležja



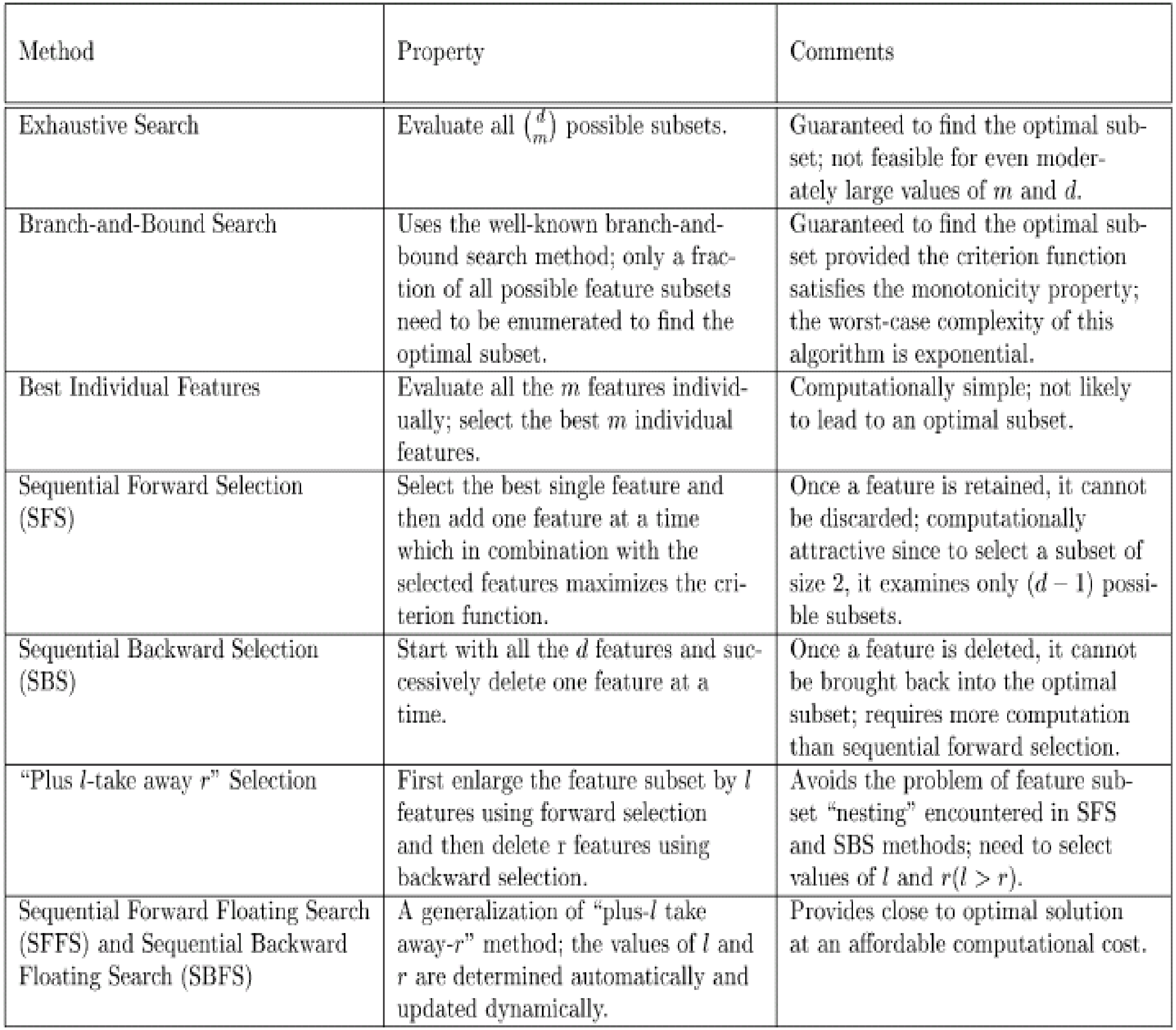
Izabran

Iris: Dodavanje narednog obeležja uz Izabrano obeležje F4



Izabran

Rezime



Method	Property	Comments
Exhaustive Search	Evaluate all $\binom{d}{m}$ possible subsets.	Guaranteed to find the optimal subset; not feasible for even moderately large values of m and d.
Branch-and-Bound Search	Uses the well-known branch-and-bound search method; only a fraction of all possible feature subsets need to be enumerated to find the optimal subset.	Guaranteed to find the optimal subset provided the criterion function satisfies the monotonicity property; the worst-case complexity of this algorithm is exponential.
Best Individual Features	Evaluate all the m features individually; select the best m individual features.	Computationally simple; not likely to lead to an optimal subset.
Sequential Forward Selection (SFS)	Select the best single feature and then add one feature at a time which in combination with the selected features maximizes the criterion function.	Once a feature is retained, it cannot be discarded; computationally attractive since to select a subset of size 2, it examines only $(d-1)$ possible subsets.
Sequential Backward Selection (SBS)	Start with all the d features and successively delete one feature at a time.	Once a feature is deleted, it cannot be brought back into the optimal subset; requires more computation than sequential forward selection.
"Plus l-take away r" Selection	First enlarge the feature subset by l features using forward selection and then delete r features using backward selection.	Avoids the problem of feature subset "nesting" encountered in SFS and SBS methods; need to select values of l and $r (l > r)$.
Sequential Forward Floating Search (SFFS) and Sequential Backward Floating Search (SBFS)	A generalization of "plus-l take away-r" method; the values of l and r are determined automatically and updated dynamically.	Provides close to optimal solution at an affordable computational cost.

Optimalni

Suboptimalni