Lecture 6

Dimensionality reduction

(Advanced) Machine Learning

Ivan Smetannikov

12.10.2016

Lecture plan

- Dimensionality Reduction
- Feature Selection
- Feature Extraction

Lecture plan

- Dimensionality Reduction
- Feature Selection
- Feature Extraction

Dimensionality Reduction

Why should we look at dimensionality reduction?

- Speeds up algorithms
- Reduces space used by data for them

Dimensionality Reduction

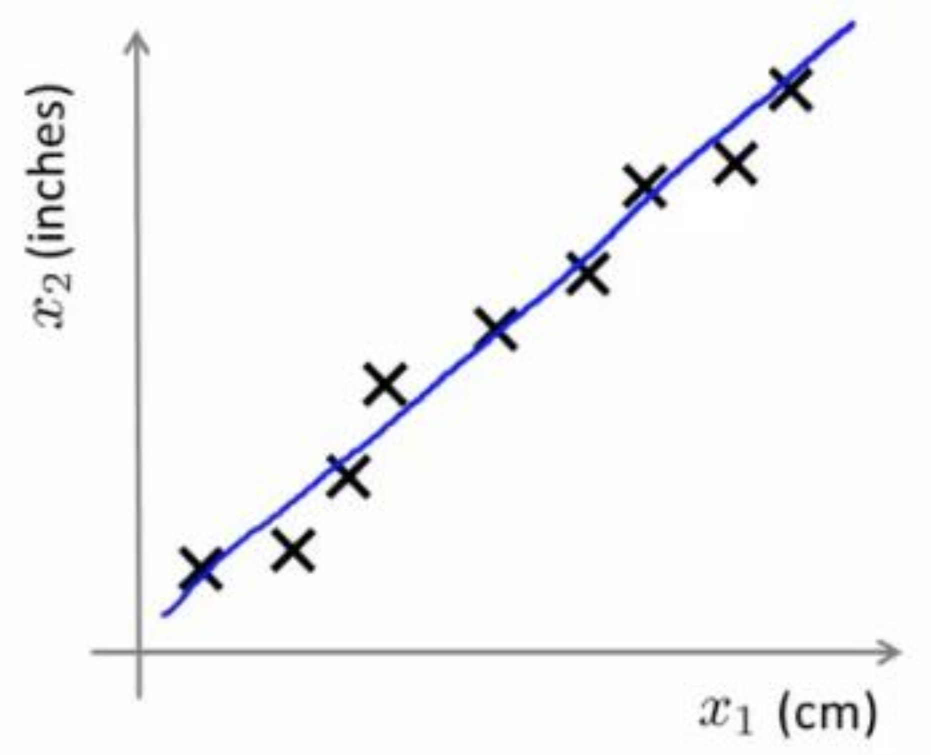
What is dimensionality reduction?

- You’ve collected many features – maybe more than you need. Can you ”simply” your data set in a rational and useful way?

Dimensionality Reduction

Example:

- Redundant data set – different units for same attribute
- Reduce data to 1D (2D -> 1D)



Dimensionality Reduction

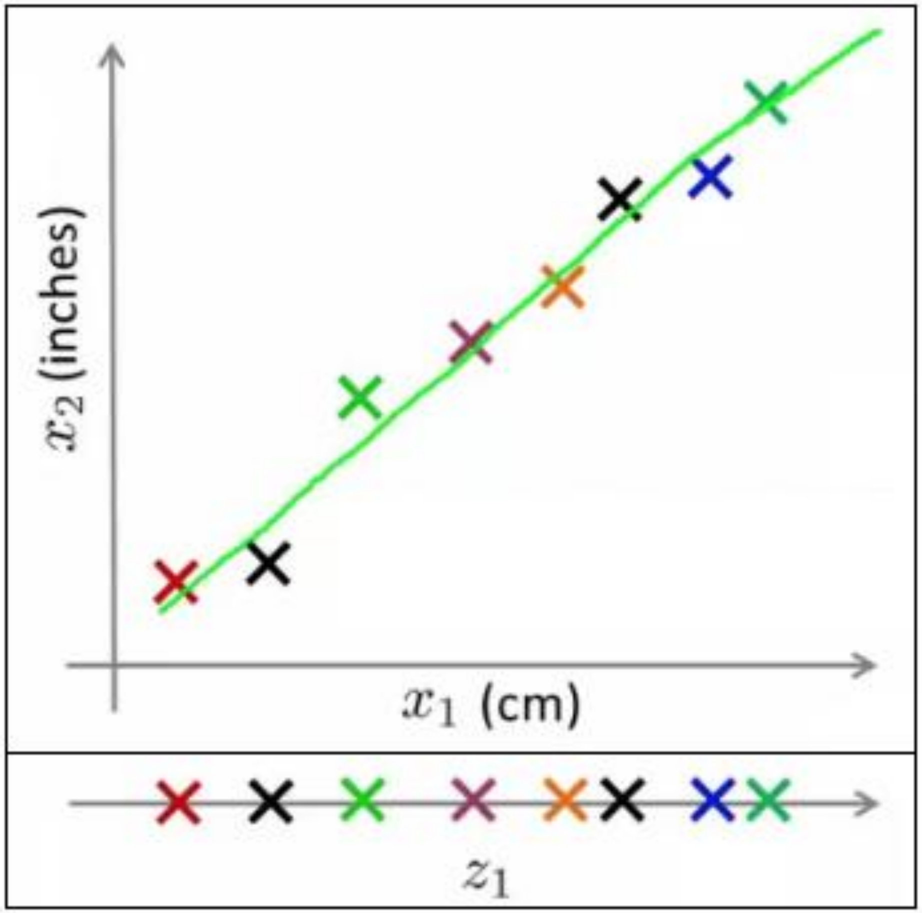
Another Example

- Helicopter flying - do a survey of pilots (x1 = skill, x2 = pilot enjoyment) These features may be highly correlated
- This correlation can be combined into a single attribute called aptitude (for example)

Dimensionality Reduction

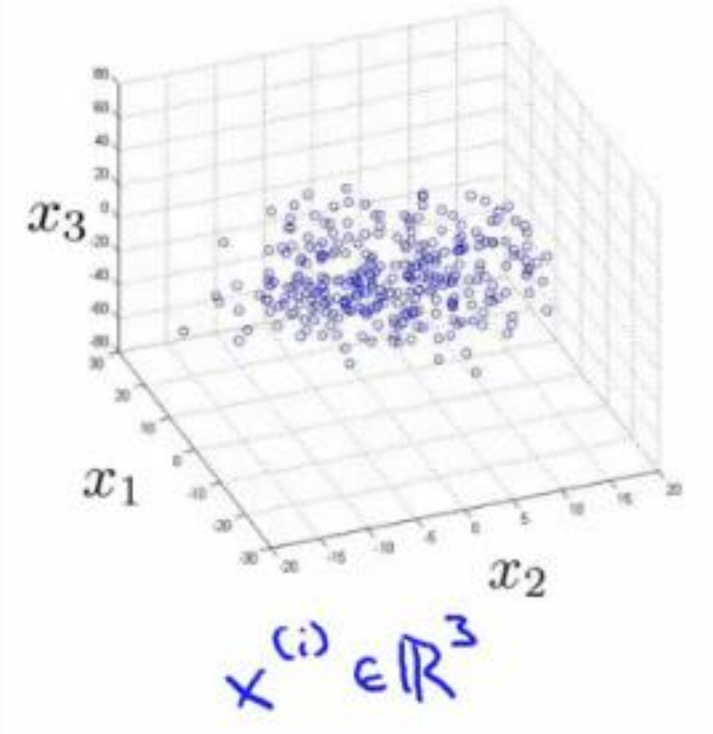
So what does dimensionality reduction mean?

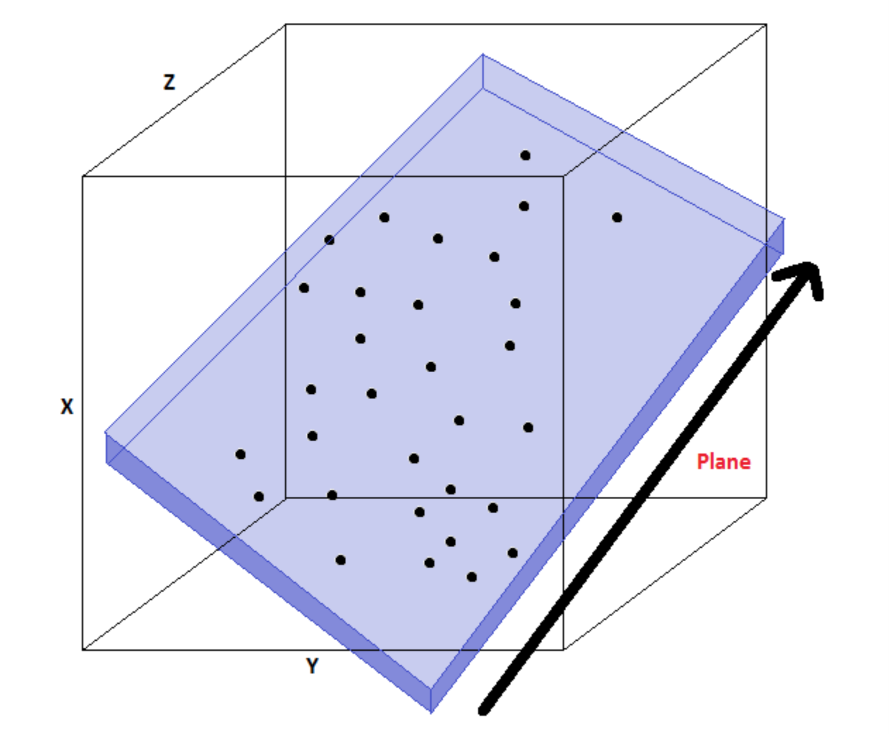
- Let plot a line
- Take exact example and record position on that line
- So we can present x^1 as 1D number



Dimensionality Reduction

Another example 3D -> 2D





Dimensionality Reduction

Motivation:

Collect a large data set (50 dimensions)

Country	GDP (trillions of US$)	Per capita GDP (thousands of intl. $)	Human Develop-ment Index	Life expectancy	Poverty Index (Gini as percentage)	Mean household income (thousands of US$)	...
Canada	1.577	39.17	0.908	80.7	32.6	67.293	...
China	5.878	7.54	0.687	73	46.9	10.22	...
India	1.632	3.41	0.547	64.7	36.8	0.735	...
Russia	1.48	19.84	0.755	65.5	39.9	0.72	...
Singapore	0.223	56.69	0.866	80	42.5	67.1	...
USA	14.527	46.86	0.91	78.3	40.8	84.3	...
...	...	...	...	...	...	...	

Dimensionality Reduction

Using dimensionality reduction come up with a different feature representation

Country	z_1	z_2
Canada	1.6	1.2
China	1.7	0.3
India	1.6	0.2
Russia	1.4	0.5
Singapore	0.5	1.7
USA	2	1.5
...	...	...

Lecture plan

- Dimensionality Reduction
- Feature Selection
- Feature Extraction

Feature Selection

Goals of feature selection:

- Avoiding overfitting and improving the quality of classification
- Best understanding of models
- Boosting of classifying models

Feature Selection

Type of elected attributes:

- Redundant attributes - do not carry any additional information
- Irrelevant attributes - are not generally informative

Feature Selection

Evaluation methods of feature selection:

- At various datasets
- With different classifiers (if possible)
- By adding to datasets noise and target vectors

Feature Selection

Feature selection types:

- Filter methods
  a. Univariate
  b. Multivariate
- Wrapper methods
  a. Deterministic
  b. Randomized
- Embedded methods

Feature Selection

Filter methods:

Evaluate the quality of certain attributes and remove the worst of them.

+ Simple to compute, easy to scale

- Ignore the relationships between attributes or features used by classifier

Feature Selection

Examples of filter methods:

- Univariate:
    - o Euclidian distance
    - o Information gain
    - Spearman corellation coefficient
- oMultivariate:
    - o CFS
    - o MBF

Feature Selection

Spearman corellation coefficient

$$\rho = \frac{\sum_{ij}(x_{ij}-\bar{x}_j)(y_i-\bar{y})}{\sqrt{\sum_{ij}(x_{ij}-\bar{x}_j)^2 \sum_i (y_i-\bar{y})^2}}$$

$\rho \in [-1; 1]$

$\rho \to 0$

Python SciPy:

```
scipy.stats.pearsonr(x, y)
```

Parameters:	**x** : *(N,) array_like* Input **y** : *(N,) array_like* Input
Returns:	*(Pearson's correlation coefficient,* 2-tailed p-value)

Feature Selection

Weka:

```
ASEvaluation evaluator = new CorrelationAttributeEval();
Ranker ranker = new Ranker();

// ranker.setThreshold(0.05); or ranker.setNumToSelect(10);

AttributeSelection selection = new AttributeSelection();

selection.setInputFormat(heavyInstances);

selection.setEvaluator(evaluator);
selection.setSearch(ranker);
Instances lightInstances = Filter.useFilter(heavyInstances, selection);
```

Feature Selection

Wrapper methods:

Get a subset of attributes of the source

+ Higher accuracy than Filtering

+ Consider the relationships between attributes

+ Direct interaction with the classifier

- Long computing time

- The probability of overfitting

Feature Selection

Examples of Wrapper methods:

- Deterministic:
  - SFS (sequential forward selection?)
  - SBE (sequential backward elimination?)
  - SVM-RFE
- Randomized:
  - Randomized Hill Climbing
  - Genetic Algorithms

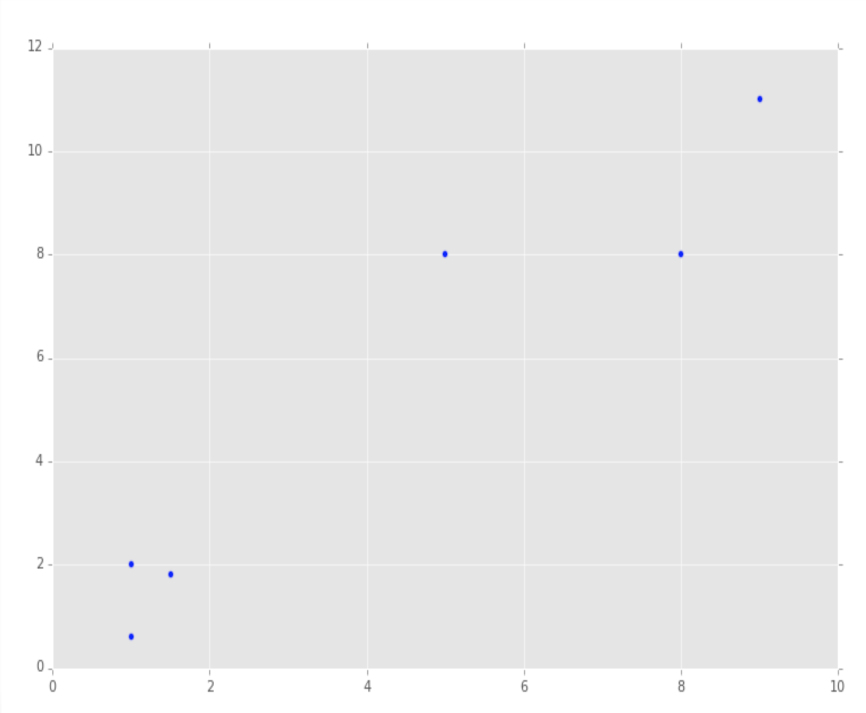
Feature Selection

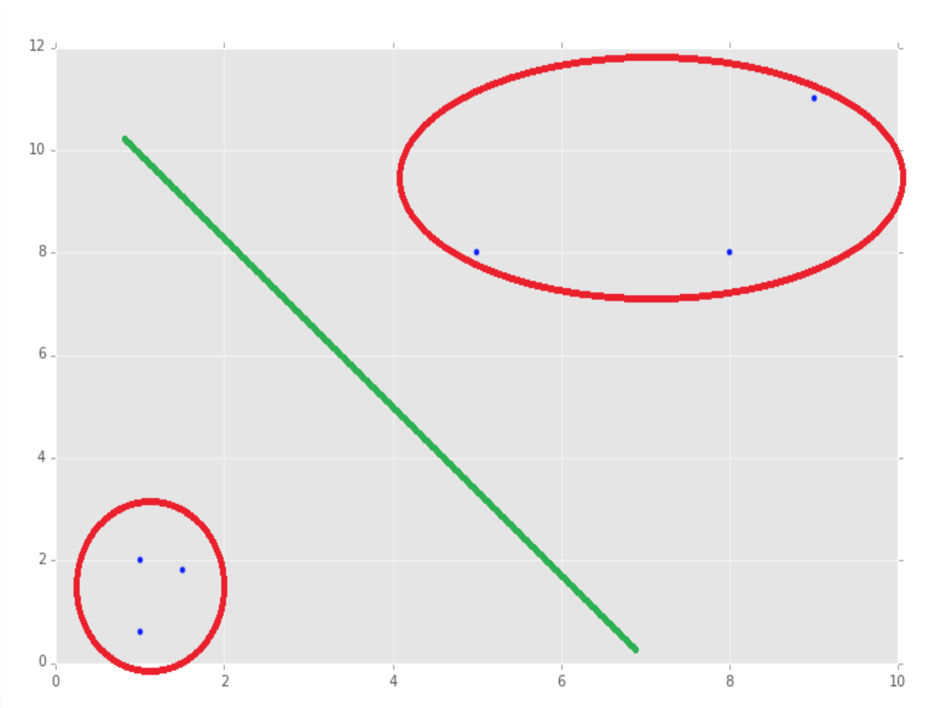
SVM-RFE

- Train SVM on training subset
- Rank features by received weights
- Throw out last features
- Repeat until the necessary amount of features will left

SVM-RFE (Python example)

```
x = [1, 5, 1.5, 8, 1, 9]
y = [2, 8, 1.8, 8, 0.6, 11]
```





Feature Selection

SVM-RFE (Python example)

```
X = np.array([[1,2], [5,8], [1.5,1.8], [8,8],
[1,0.6], [9,11]])
y = [0,1,0,1,0,1]
```

Let use SVM:

```
clf = svm.SVC(kernel='linear', C = 1.0)
```

Let fit our model:

```
clf.fit(X,y)
```

Let predict predict something:

```
print(clf.predict([0.58,0.76]))
```

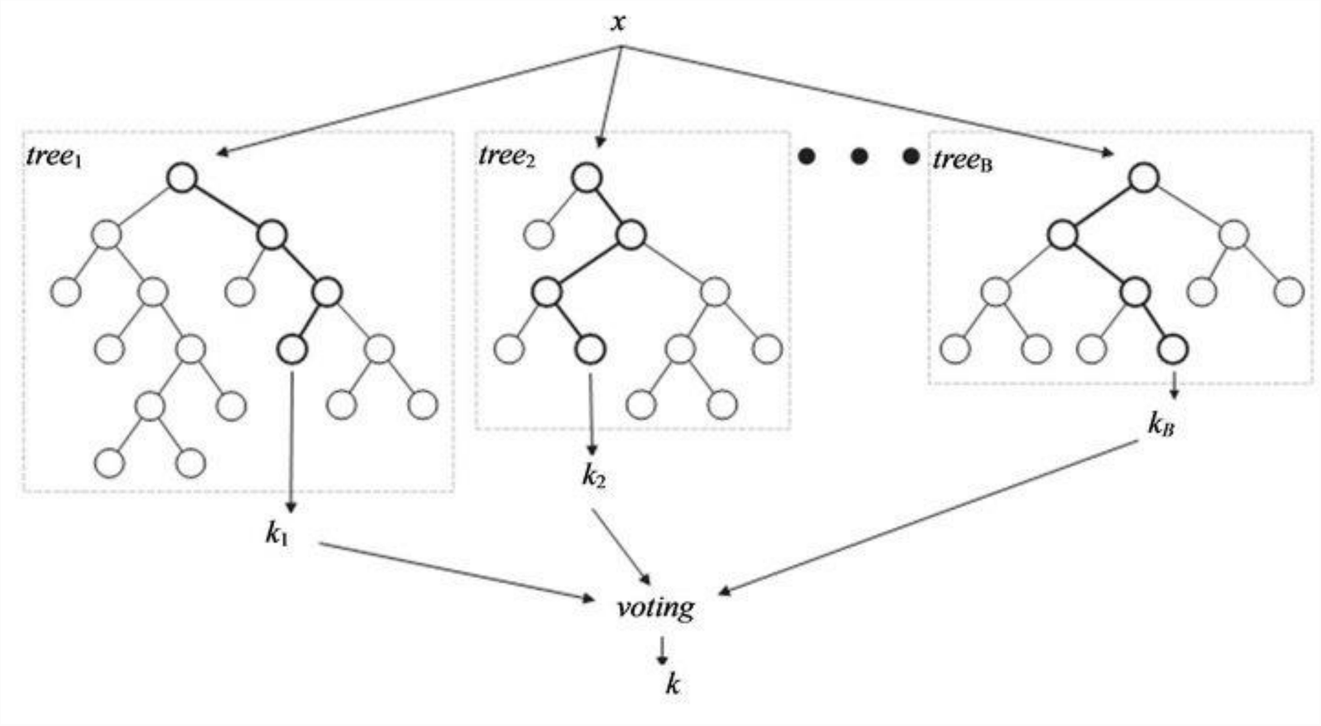
Feature Selection

Embedded

- Take into account the particular classifier
- Use individual method for each classifier

Feature Selection

Random Forest:



Feature Selection

Random Forest:

- Select a subsample of size N for each tree with replacement
- Build decision trees. To select next feature to split the $m \approx \sqrt{M}$ considered
- Choose the best for a given criteria

Random Forest (Python example):

```
# Import the random forest package
from sklearn.ensemble import RandomForestClassifier
# Create the random forest object which will include all the
parameters for the fit
forest = RandomForestClassifier(n_estimators = 100)
# Fit the training data to the Survived labels and create the
decision trees
forest = forest.fir(train_data[0::, 1::],
train_data[0::, 0])
# Take the same decision trees and run it on the test data
output = forest.predict(test_data)
```

Feature Selection

Random Forest (Weka):

```
int numFolds = 10;
br = new BufferedReader(new FileReader(“data.arff"));

      Instances trainData = new Instances(br);
      trainData.setClassIndex(trainData.numAttributes() - 1);

      RandomForest rf = new RandomForest();
      rf.setNumTrees(100);

      rf.buildClassifier(trainData);
      Evaluation evaluation = new Evaluation(trainData);
      evaluation.crossValidateModel(rf, trainData, numFolds, new Random(1));

      System.out.println("F-measure=  " + evaluation.fMeasure(0));
```

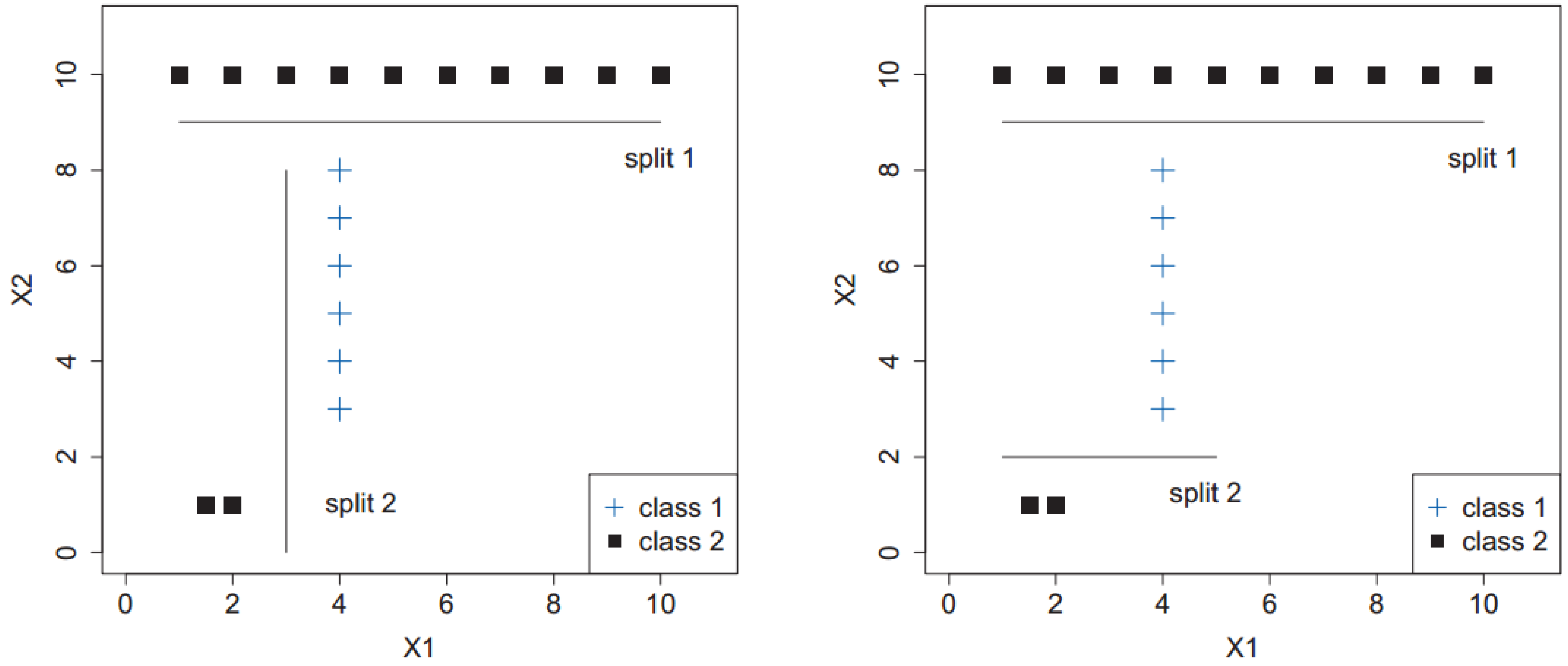
Feature Selection

IG and IG

$$gain(T) = -\sum_{i=1}^{k} p(c_i)\log_2\big(p(c_i)\big) + \sum_{i=1}^{n} p(t_i)\sum_{j=1}^{k} p(c_j|t_i)\log_2\big(p(c_j|t_i)\big).$$

$$gini(T) = 1 - \sum_{i=1}^{k}\big(p(c_i)\big)^2 - \sum_{i=1}^{n} p(t_i)\sum_{j=1}^{k} p(c_j|t_i)\big(1 - p(c_j|t_i)\big).$$

Redundancy



split 1
split 2
class 1
class 2
X1
X2
0
2
4
6
8
10
split 1
split 2
class 1
class 2
X1
X2

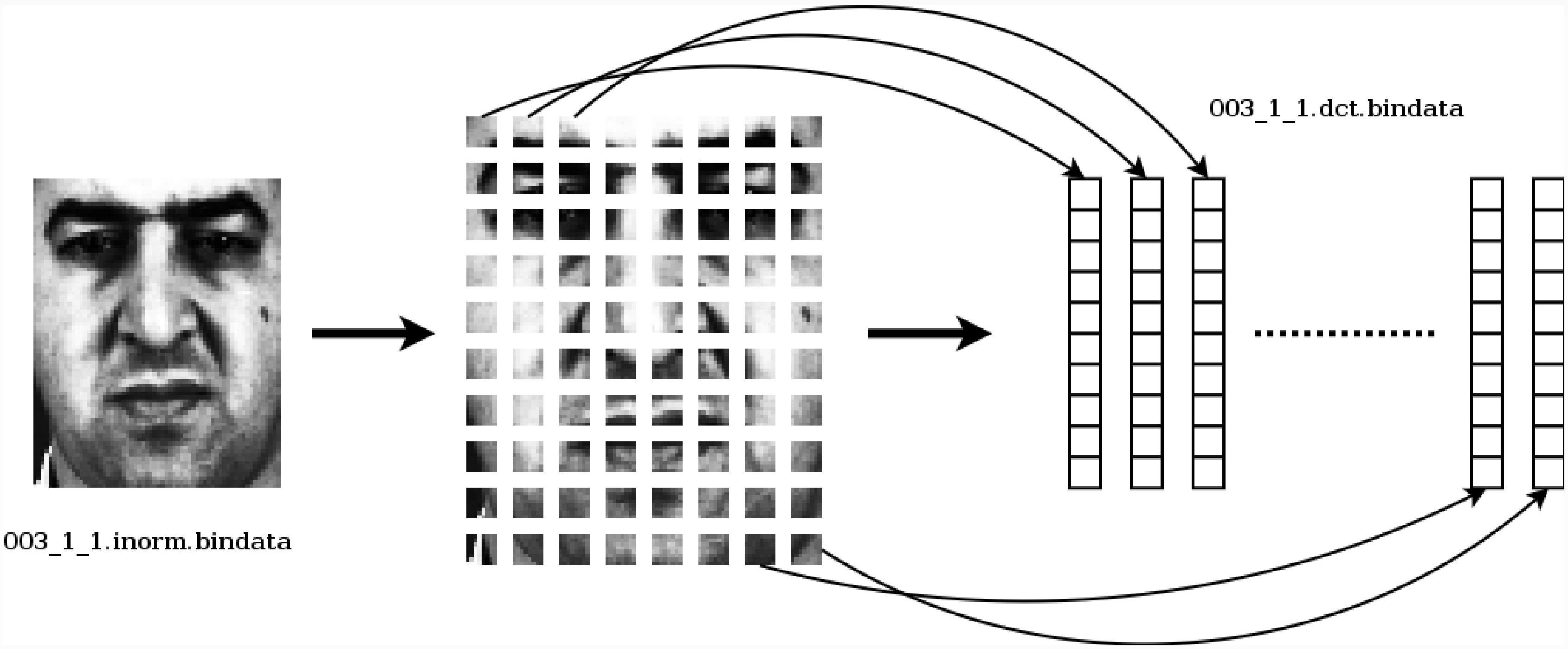
Regularization

$$gain_R(X_j) = \begin{cases} \lambda \cdot gain(X_j) & X_j \notin F \\ gain(X_i) & X_j \in F \end{cases}$$

Lecture plan

- Dimensionality Reduction
- Feature Selection
- Feature Extraction

Feature Extraction



Feature Extraction

Feature Extraction

- Reducing the amount of resources required to describe a large set of data
- New features
- Linear and nonlinear

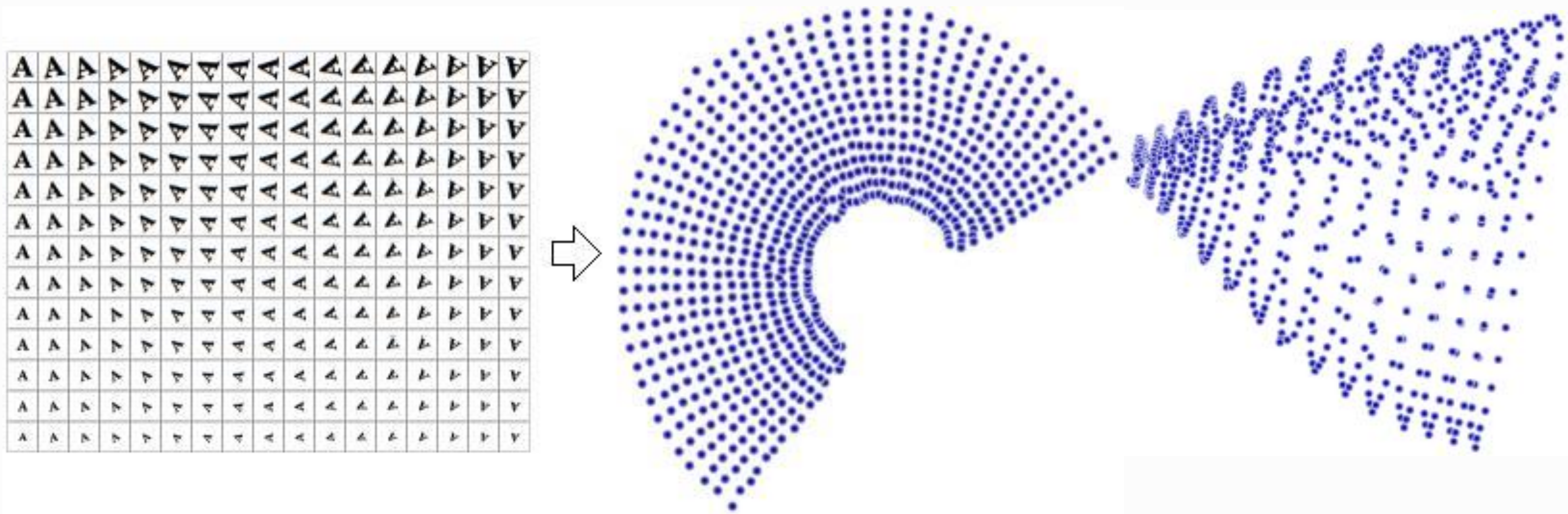
Feature Extraction

Feature Extraction

- Reducing the amount of resources required to describe a large set of data
- New features
- Linear and nonlinear

Feature Extraction

Linear and nonlinear

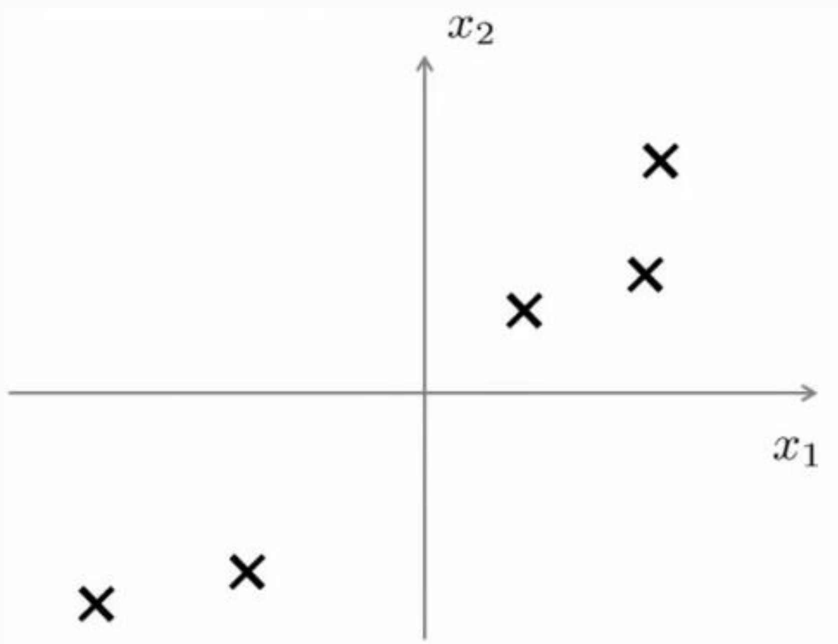


Maniford Sculpting PCA

Feature Extraction

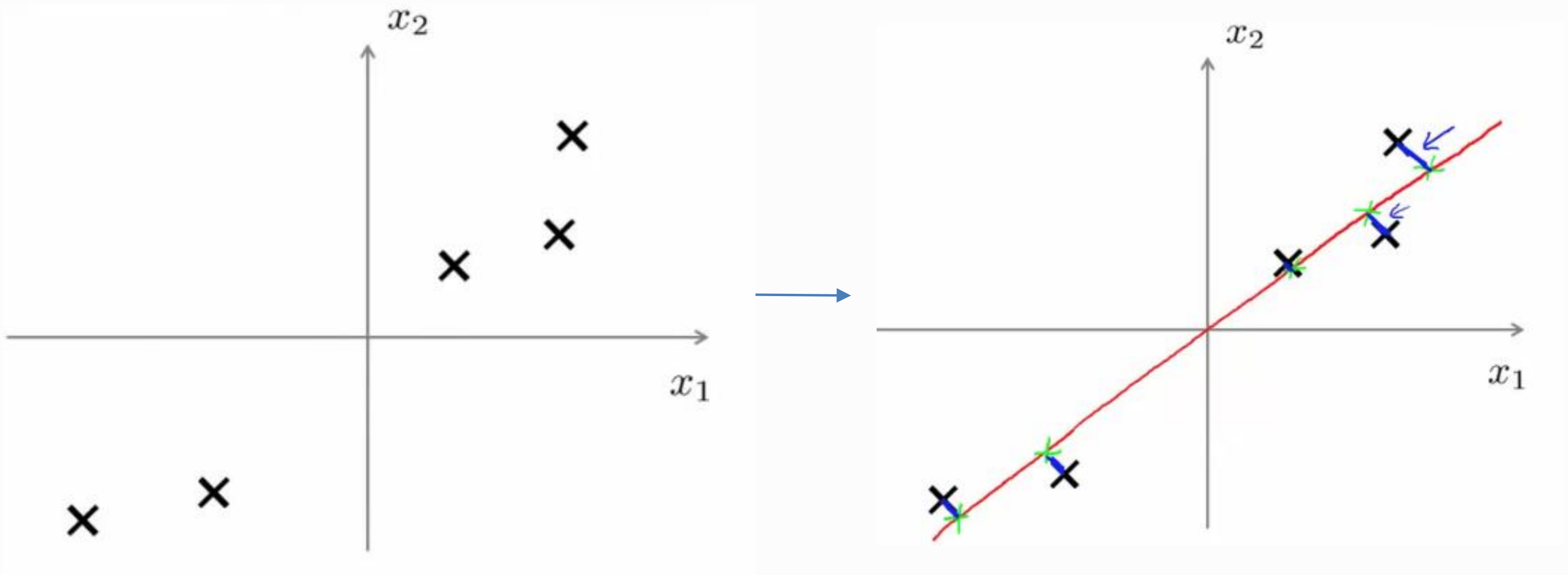
PCA

We have 2D dataset which we wish to reduce to 1D



Feature Extraction

PCA tries to find the surface (a straight line in this case) which has
the minimum projection error



Feature Extraction

PCA (Python example)

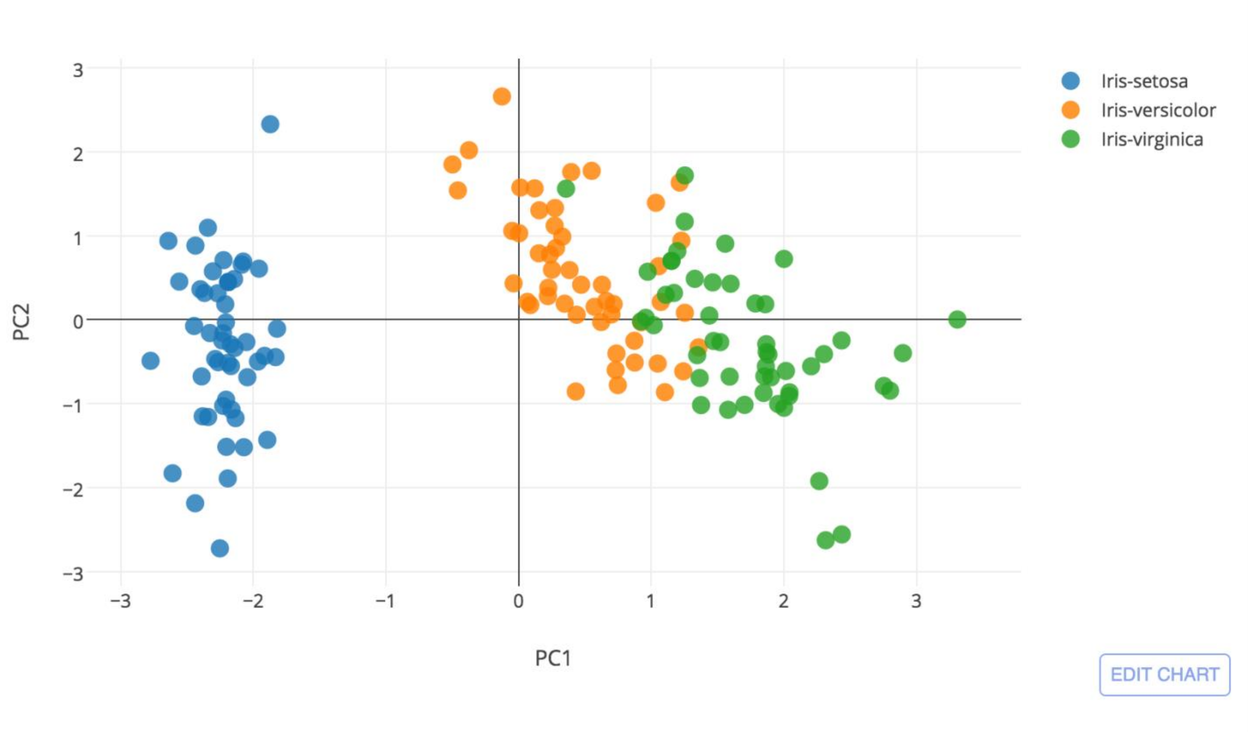
Let use Iris-data and import PCA

```
from sklearn.decomposition import PCA as sklearnPCA
sklearn_pca = sklearnPCA(n_components=2)
Y_sklearn = sklearn_pca.fit_transform(X_std)
```

Feature Extraction

PCA (Python example)

Let plot PCA-results



Feature Extraction

PCA (Weka)

```
PrincipalComponents pca = new PrincipalComponents();

pca.setInputFormat(trainingData);
pca.setMaximumAttributes(100);
newData = Filter.useFilter(newData, pca);
```