# Classical Data Analysis


**Master in Big Data Solutions 2017-2018**

Francisco Gutierres
francisco.gutierres@bts.tech

Sara Hajian
sara.hajian@bts.tech

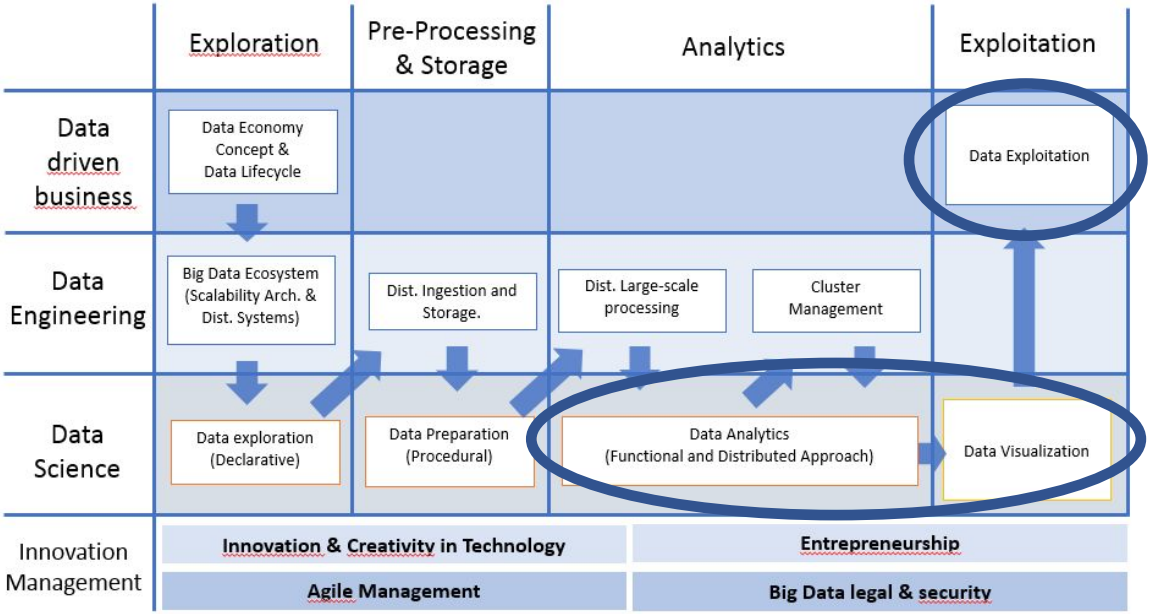
# Session 2.2.3 - Classification
# Decision Trees II

Sara Hajian

# What we will learn

**Session 2.2.3: Classification**

Decision Trees

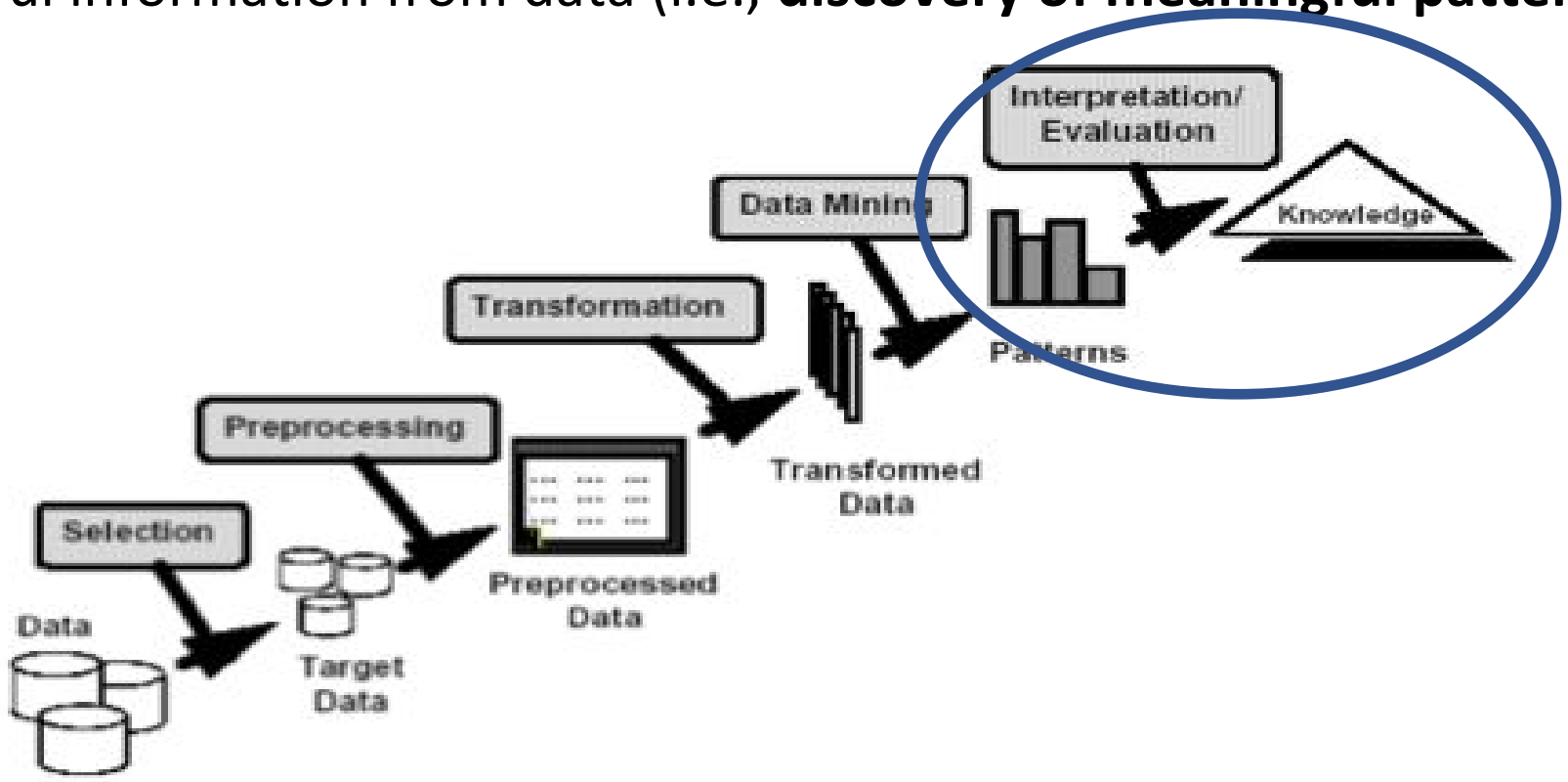



We will learn how to:

- Model overfitting
- Estimation of generalization errors
- Handling overfitting in decision tree induction
- Evaluating the performance of a classifier
- Methods for comparing classifiers

# What is data mining

Non-trivial extraction of implicit, previously unknown and potentially useful information from data (i.e., **discovery of meaningful patterns**)




Tan, Pang-Ning. *Introduction to data mining*. Pearson Education India, 2006.

# Data mining algorithms

**Supervised**: Classification and regression

**Unsupervised**: Clustering

# Classical Data analysis: Curriculum

**Part I: Regression**

Linear regression

Logistic regression

**Part II: Classification**

Supervised learning algorithms

Support vector machines (SVM)

Decision trees

Ensemble methods

**Clustering**

# Classification: definition

Given a collection of records (*training set* )

- Each record contains a set of *attributes*, one of the attributes is the *class*.

Find a *model* for class attribute as a function of the values of other attributes.

Goal: previously unseen records should be assigned a class as accurately as possible.

- A *test set* is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

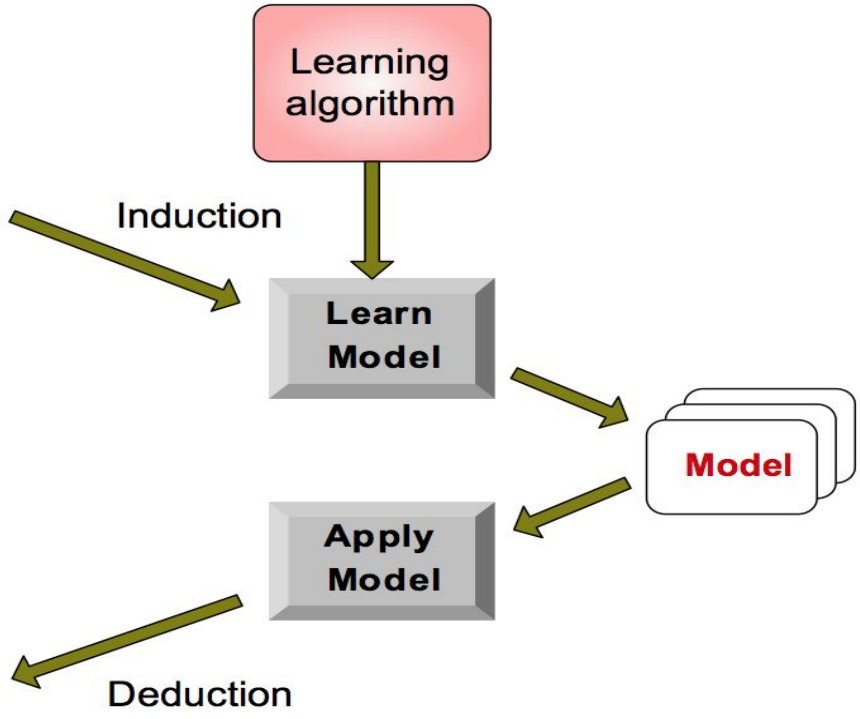
# Classification task

| Tid | Attrib1 | Attrib2 | Attrib3 | Class |
|---|---|---|---|---|
| 1 | Yes | Large | 125K | No |
| 2 | No | Medium | 100K | No |
| 3 | No | Small | 70K | No |
| 4 | Yes | Medium | 120K | No |
| 5 | No | Large | 95K | Yes |
| 6 | No | Medium | 60K | No |
| 7 | Yes | Large | 220K | No |
| 8 | No | Small | 85K | Yes |
| 9 | No | Medium | 75K | No |
| 10 | No | Small | 90K | Yes |

Training Set

| Tid | Attrib1 | Attrib2 | Attrib3 | Class |
|---|---|---|---|---|
| 11 | No | Small | 55K | ? |
| 12 | Yes | Medium | 80K | ? |
| 13 | Yes | Large | 110K | ? |
| 14 | No | Small | 95K | ? |
| 15 | No | Large | 67K | ? |

Test Set




Tan, Pang-Ning. *Introduction to data mining*. Pearson Education India, 2006.

# Classification: applications

**Health**

- Goal: Predicting tumor cells as benign or malignant

**Fraud Detection**

- Goal: Predict fraudulent cases in credit card transactions.

**News:**

- Goal: Categorizing news stories as finance, weather, entertainment, sports, etc

# Classification: algorithms

**Support vector machines (SVM)**

**Decision trees**

**Ensemble methods**

Rule-based Methods

Neural Networks

Bayesian algorithms such as Naïve Bayes

Instance-based algorithms such kNN

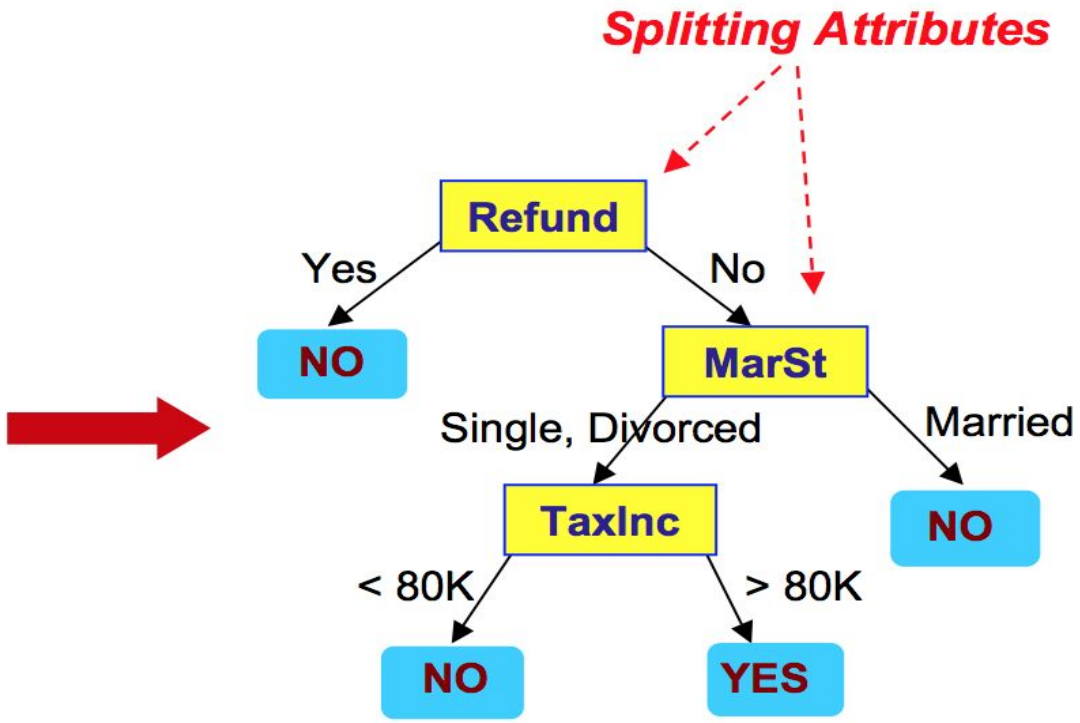
Deep learning

# A Decision Tree: Example

| Tid | Refund | Marital Status | Taxable Income | Cheat |
|---|---|---|---|---|
| | | | | |
| 1 | Yes | Single | 125K | No |
| 2 | No | Married | 100K | No |
| 3 | No | Single | 70K | No |
| 4 | Yes | Married | 120K | No |
| 5 | No | Divorced | 95K | Yes |
| 6 | No | Married | 60K | No |
| 7 | Yes | Divorced | 220K | No |
| 8 | No | Single | 85K | Yes |
| 9 | No | Married | 75K | No |
| 10 | No | Single | 90K | Yes |

categorical categorical continuous class

Training Data




Model: Decision Tree

# Decision tree induction

**Greedy strategy**

Split the records based on an attribute test that optimizes certain criterion.

**Issues**

Determine how to split the records

- How to specify the attribute test condition?
- How to determine the best split?

Determine when to stop splitting

# Decision tree induction

**Greedy strategy**

Split the records based on an attribute test that optimizes certain criterion.

**Issues**

Determine how to split the records

- How to specify the attribute test condition?
- How to determine the best split?

Determine when to stop splitting

# Model overfitting and underfitting

**Training error:** it the number of misclassification errors committed on training records

**Generalization error**: it is the expected error of the model in previously unseen records

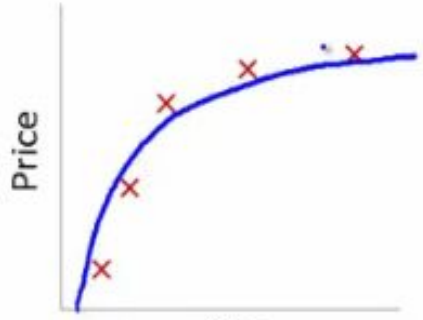
**What is a good regression or classification model?**

# Model overfitting and underfitting

## Regression

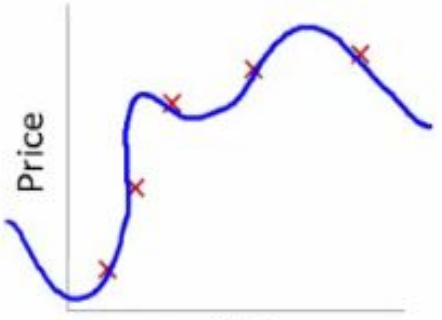

$\theta_0 + \theta_1 x$

High bias
(underfit)




$\theta_0 + \theta_1 x + \theta_2 x^2$

"Just right"

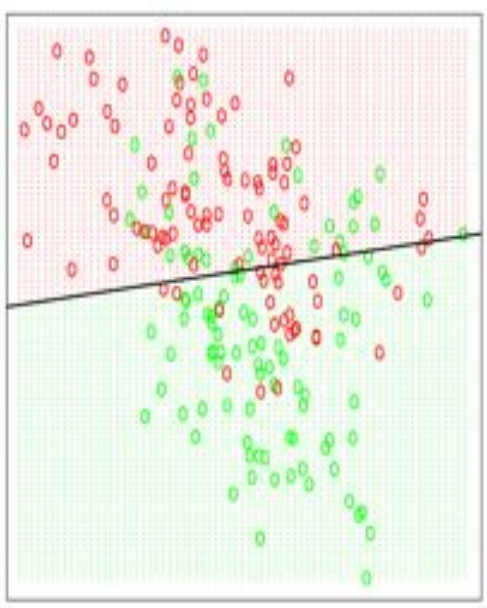


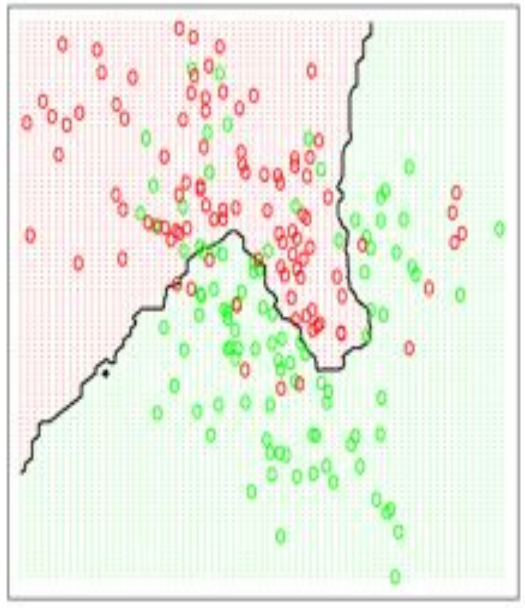

$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

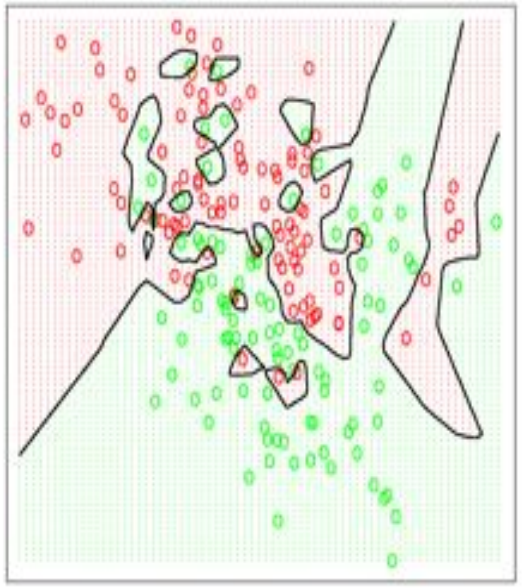
High variance
(overfit)

# Model overfitting and underfitting

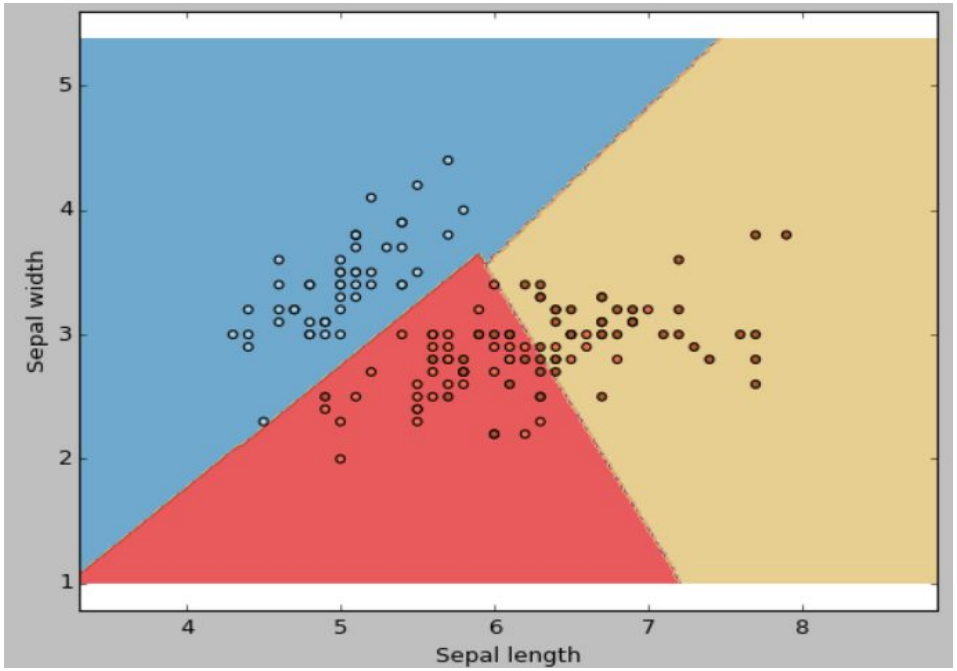
## Classification

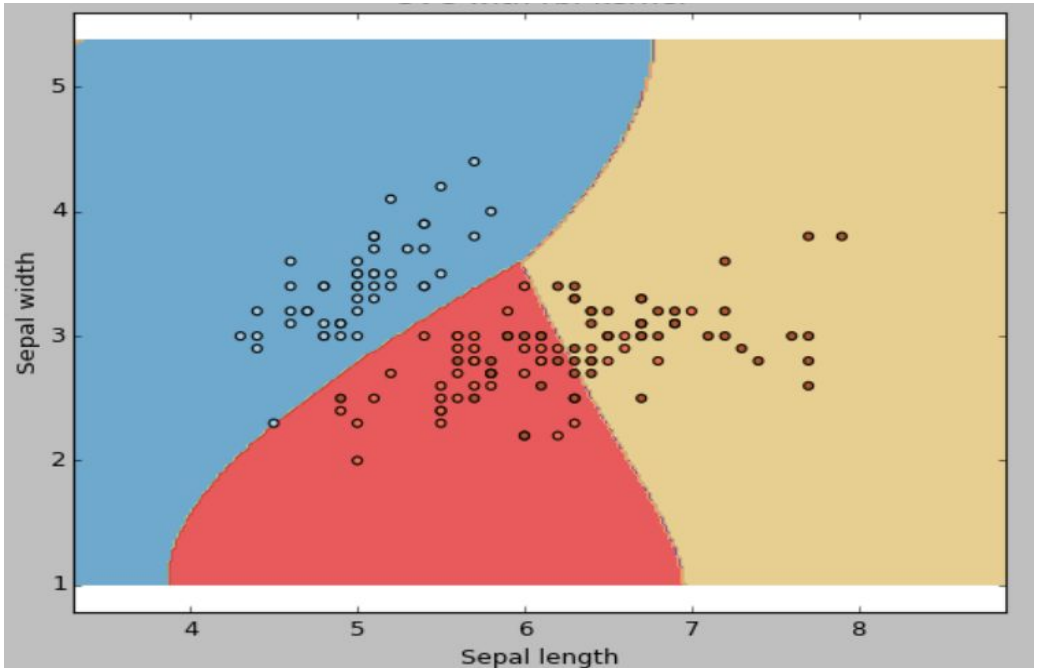






# Model overfitting and underfitting: SVM



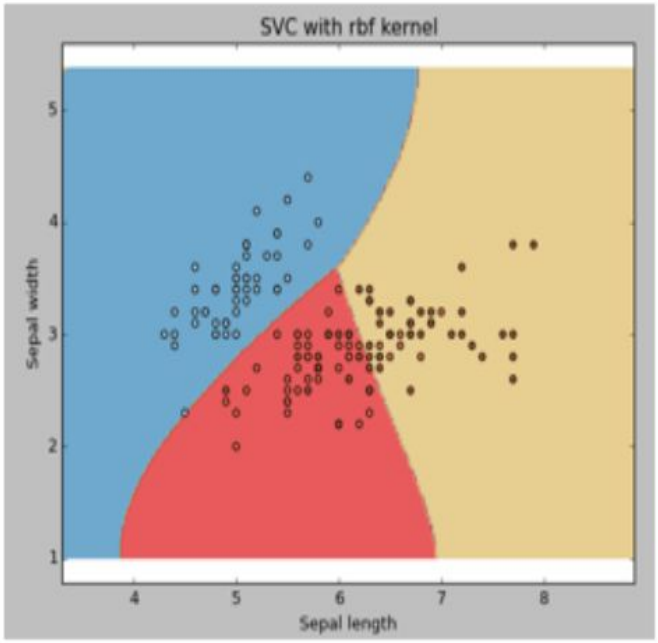



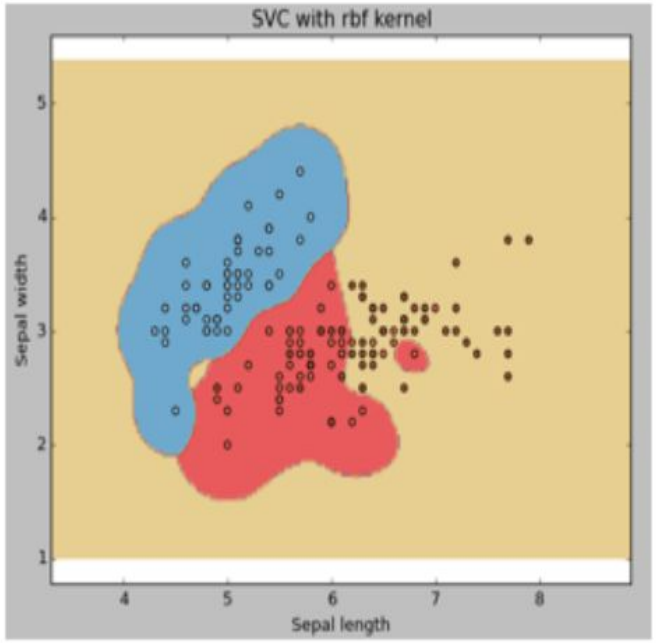

Which kernel? linear or not?

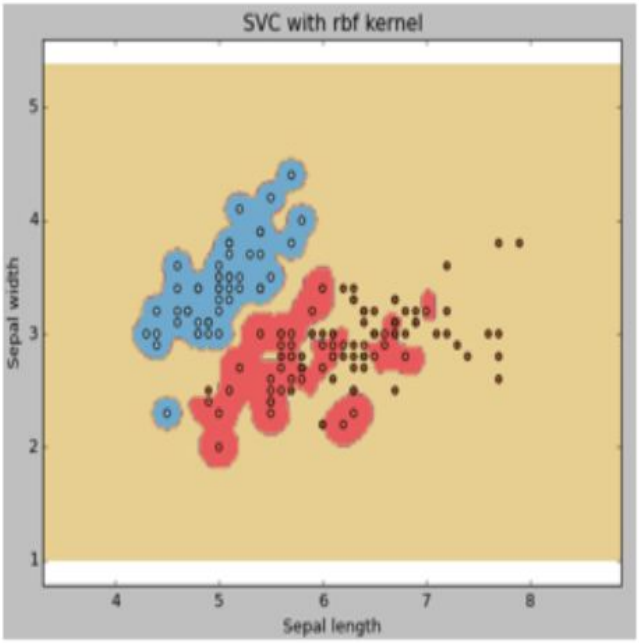
BTS | Barcelona Technology School

eurecat Centre Tecnològic de Catalunya

# Model overfitting and underfitting: SVM



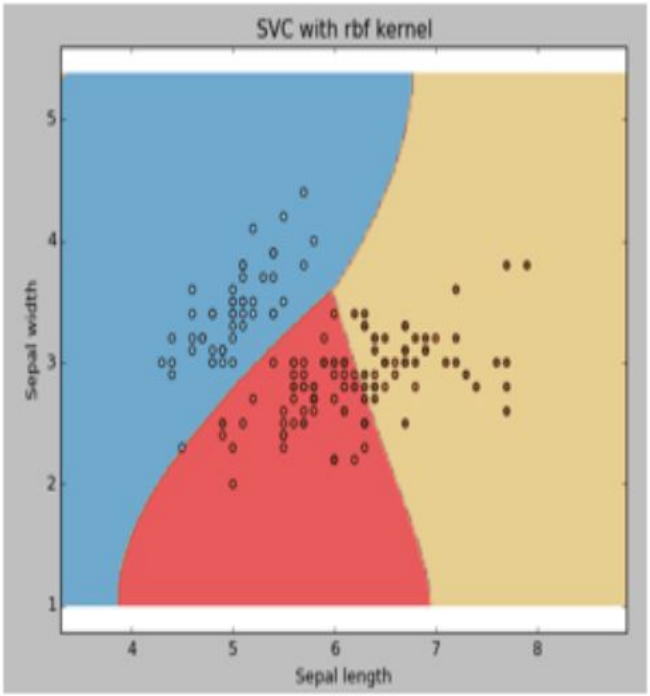



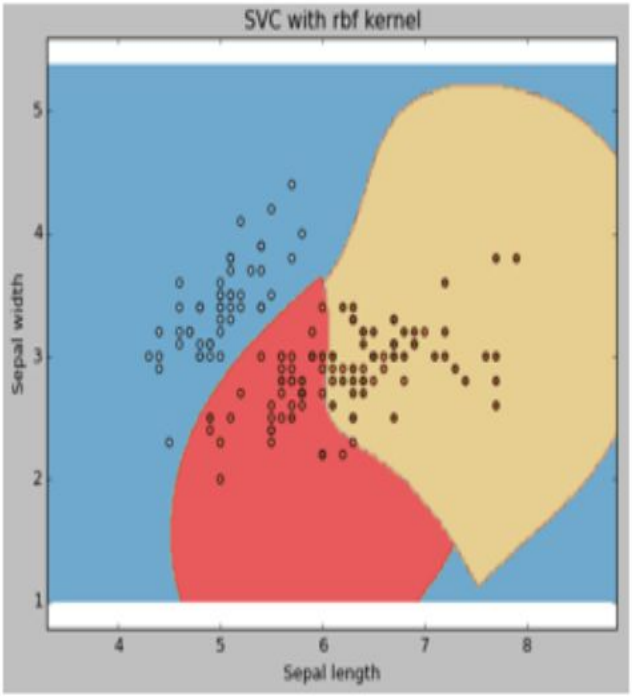



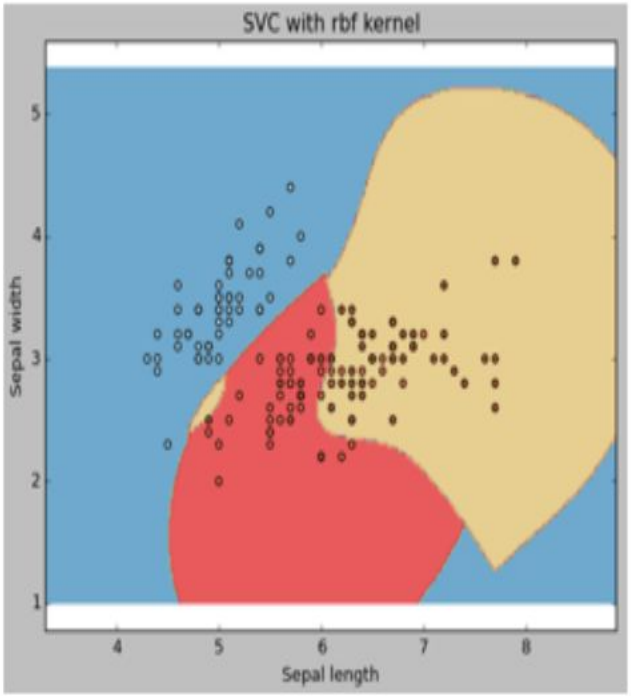

Which gamma (kernel coefficient)?

0, 10, 100

# Model overfitting and underfitting: SVM



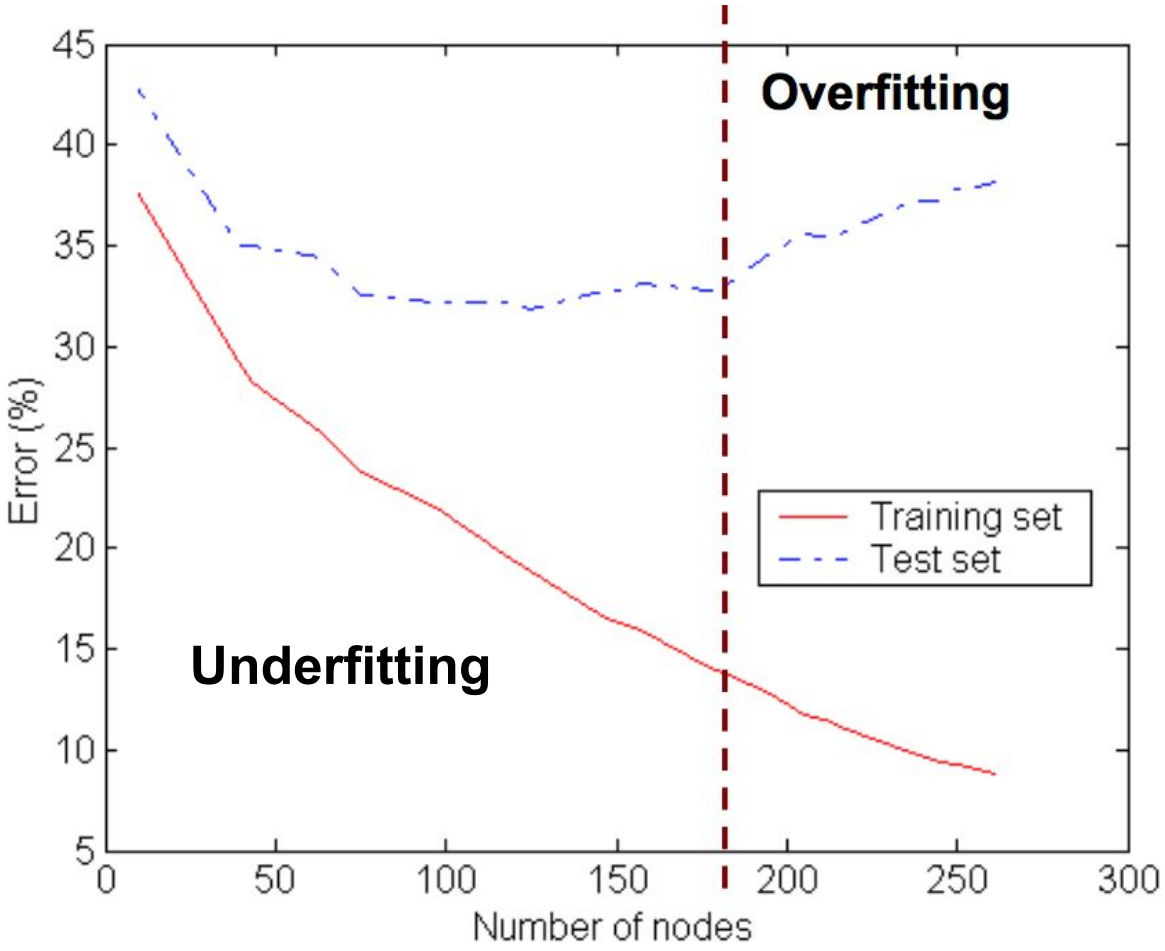




Which C (penalty parameter)?

1, 100, 1000

# Model overfitting and underfitting



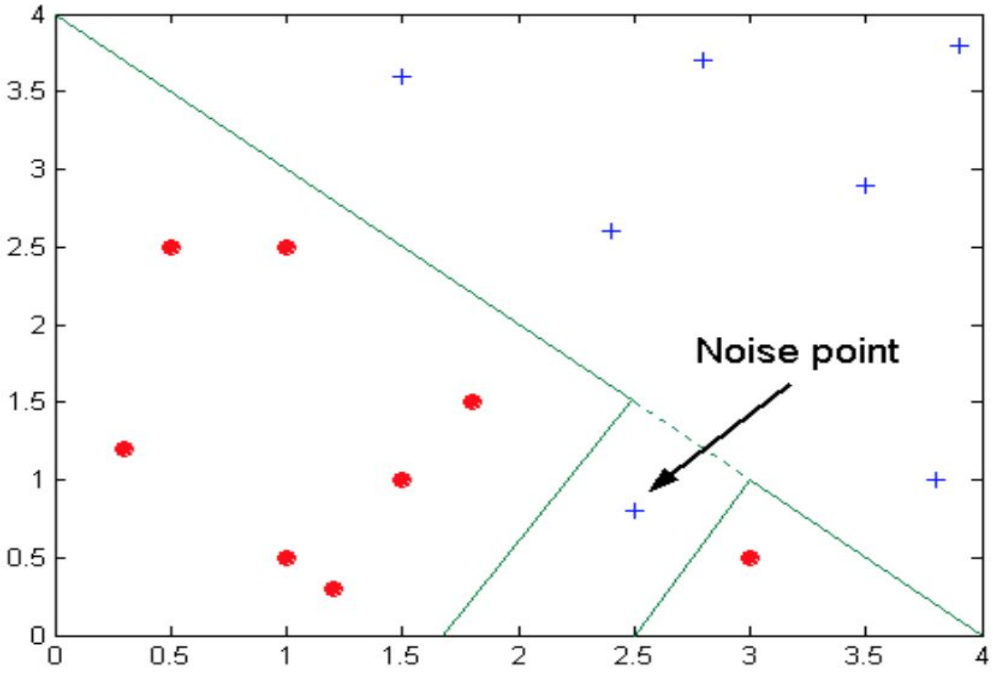

Model complexity

# Model overfitting

Why does it happen?

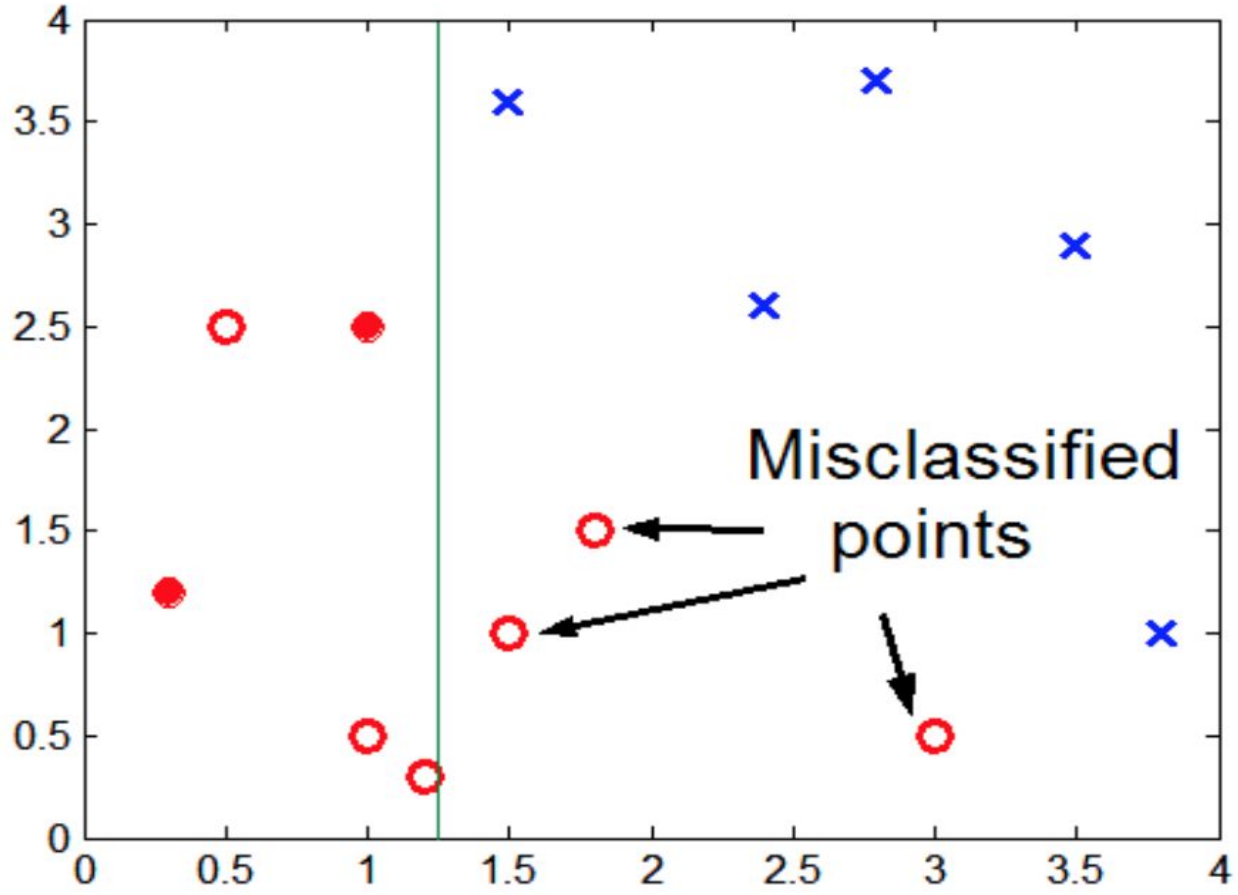
How to detect and prevent it?

# Model overfitting due to Noise




**Decision boundary is distorted by noise point**

# Model overfitting due to insufficient examples



# Overfitting in decision trees

Overfitting results in decision trees that are more complex than necessary

Training error no longer provides a good estimate of how well the tree will perform on previously unseen records

Need new ways for estimating errors

# Estimating generalization errors

- Re-substitution errors: error on training ($\Sigma$ e(t) )
- Generalization errors: error on testing ($\Sigma$ e'(t))
- Methods for estimating generalization errors:
  - Optimistic approach: e'(t) = e(t)
  - Pessimistic approach:
    - For each leaf node: e'(t) = (e(t)+0.5)
    - Total errors: e'(T) = e(T) + N $\times$ 0.5 (N: number of leaf nodes)
    - For a tree with 30 leaf nodes and 10 errors on training (out of 1000 instances):
      Training error = 10/1000 = 1%
      Generalization error = (10 + 30$\times$0.5)/1000 = 2.5%
  - Reduced error pruning (REP):
    - uses validation data set to estimate generalization error

# Occam's Razor

- Given two models of similar generalization errors, one should prefer the simpler model over the more complex model

- For complex models, there is a greater chance that it was fitted accidentally by errors in data

- Therefore, one should include model complexity when evaluating a model

# How to address overfitting in decision trees

- Pre-Pruning (Early Stopping Rule)
  - Stop the algorithm before it becomes a fully-grown tree
  - Typical stopping conditions for a node:
    - Stop if all instances belong to the same class
    - Stop if all the attribute values are the same
  - More restrictive conditions:
    - Stop if number of instances is less than some user-specified threshold
    - Stop if class distribution of instances are independent of the available features (e.g., using $\chi^2$ test)
    - Stop if expanding the current node does not improve impurity measures (e.g., Gini or information gain).

# How to address overfitting in decision trees

- Post-pruning
  - Grow decision tree to its entirety
  - Trim the nodes of the decision tree in a bottom-up fashion
  - If generalization error improves after trimming, replace sub-tree by a leaf node.
  - Class label of leaf node is determined from majority class of instances in the sub-tree

# Model overfitting: How to detect it?

- If our model does much better on the training set than on the test set, then we're likely overfitting.
- Another tip is to start with a very simple model to serve as a benchmark.

# Model overfitting: How to prevent it?

- Cross-validation
- Train with more data
- Remove features
- Early stopping
- Regularization
- Ensembling

# Model evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

# Model evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

# Metrics for Performance Evaluation

- Focus on the predictive capability of a model
  - Rather than how fast it takes to classify or build models, scalability, etc.
- Confusion Matrix:

| | | PREDICTED CLASS | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | a | b |
| | Class=No | c | d |

**a: TP (true positive)**

**b: FN (false negative)**

**c: FP (false positive)**

**d: TN (true negative)**

# Metrics for Performance Evaluation

| | | PREDICTED CLASS | |
|---|---|---|---|
| | | Class=Yes | Class=No |
| ACTUAL CLASS | Class=Yes | a (TP) | b (FN) |
| | Class=No | c (FP) | d (TN) |

- Most widely-used metric:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

# Limitations of accuracy

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10

- If model predicts everything to be class 0, accuracy is 9990/10000 = 99.9 %
  - Accuracy is misleading because model does not detect any class 1 example

# Cost matrix

| | PREDICTED CLASS | | |
|---|---|---|---|
| ACTUAL CLASS | C(i\|j) | **Class=Yes** | **Class=No** |
| | **Class=Yes** | C(Yes\|Yes) | C(No\|Yes) |
| | **Class=No** | C(Yes\|No) | C(No\|No) |

C(i|j): Cost of misclassifying class j example as class i

# Computing cost of classification

| Cost Matrix | PREDICTED CLASS | | |
|---|---|---|---|
| | C(i\|j) | + | - |
| ACTUAL CLASS | + | -1 | 100 |
| | - | 1 | 0 |

| Model $M_1$ | PREDICTED CLASS | | |
|---|---|---|---|
| | | + | - |
| ACTUAL CLASS | + | 150 | 40 |
| | - | 60 | 250 |

Accuracy = 80%

Cost = 3910

| Model $M_2$ | PREDICTED CLASS | | |
|---|---|---|---|
| | | + | - |
| ACTUAL CLASS | + | 250 | 45 |
| | - | 5 | 200 |

Accuracy = 90%

Cost = 4255

# Cost-sensitive evaluation metrics

$$\text{Precision (p)} = \frac{a}{a+c}$$

$$\text{Recall (r)} = \frac{a}{a+b}$$

$$\text{F-measure (F)} = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

- Precision is biased towards C(Yes|Yes) & C(Yes|No)
- Recall is biased towards C(Yes|Yes) & C(No|Yes)
- F-measure is biased towards all except C(No|No)

$$\text{Weighted Accuracy} = \frac{w_1 a + w_4 d}{w_1 a + w_2 b + w_3 c + w_4 d}$$

# Precision versus Recall

When a **search engine** returns 30 pages only 20 of which were relevant while failing to return 40 additional relevant pages, its precision is 20/30 = 2/3 while its recall is 20/60 = 1/3. So, in this case, **precision** is "how useful the search results are", and **recall** is "how complete the results are"

# Precision versus Recall

**Precision**

(Of all patients where we predicted, what fraction actually has cancer?)

**Recall**

(Of all patients that actually have cancer, what fraction did we correctly detect as having cancer?)

# Model evaluation

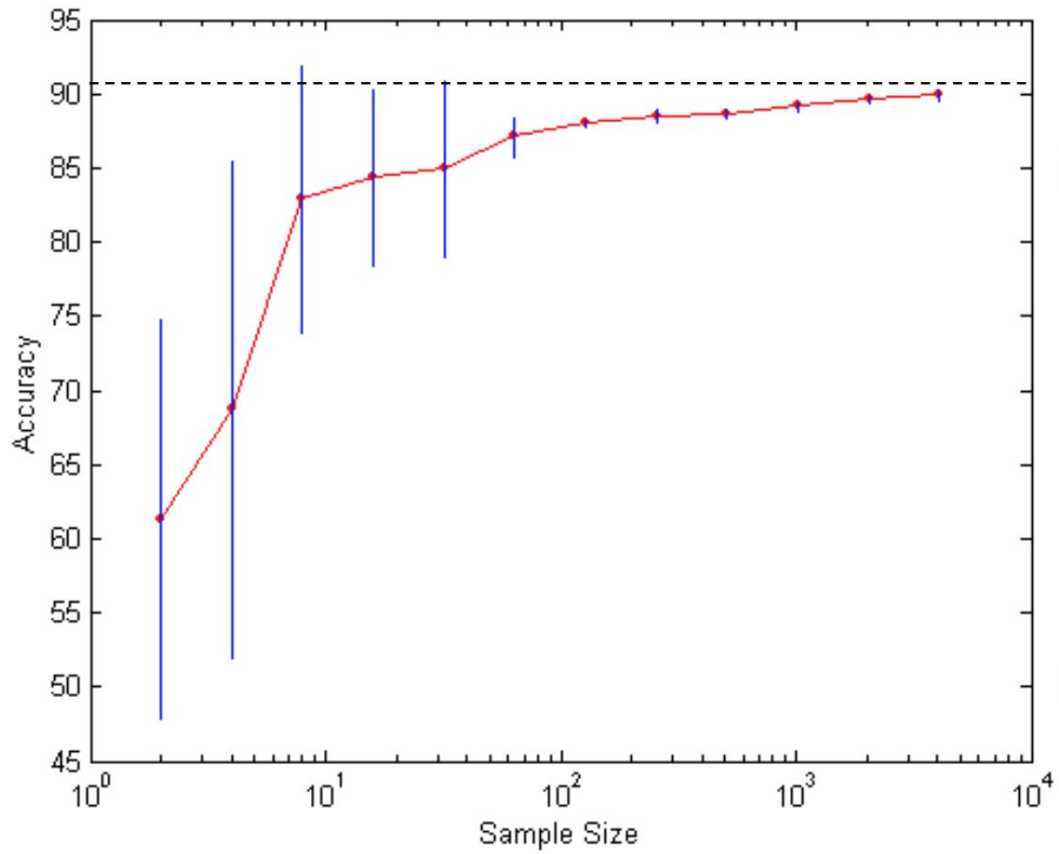
- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

# Methods for Performance Evaluation

- How to obtain a reliable estimate of performance?

- Performance of a model may depend on other factors besides the learning algorithm:
  - Class distribution
  - Cost of misclassification
  - Size of training and test sets

# Learning curve




- Learning curve shows how accuracy changes with varying sample size
- Requires a sampling schedule for creating learning curve:
  - Arithmetic sampling (Langley, et al)
  - Geometric sampling (Provost et al)

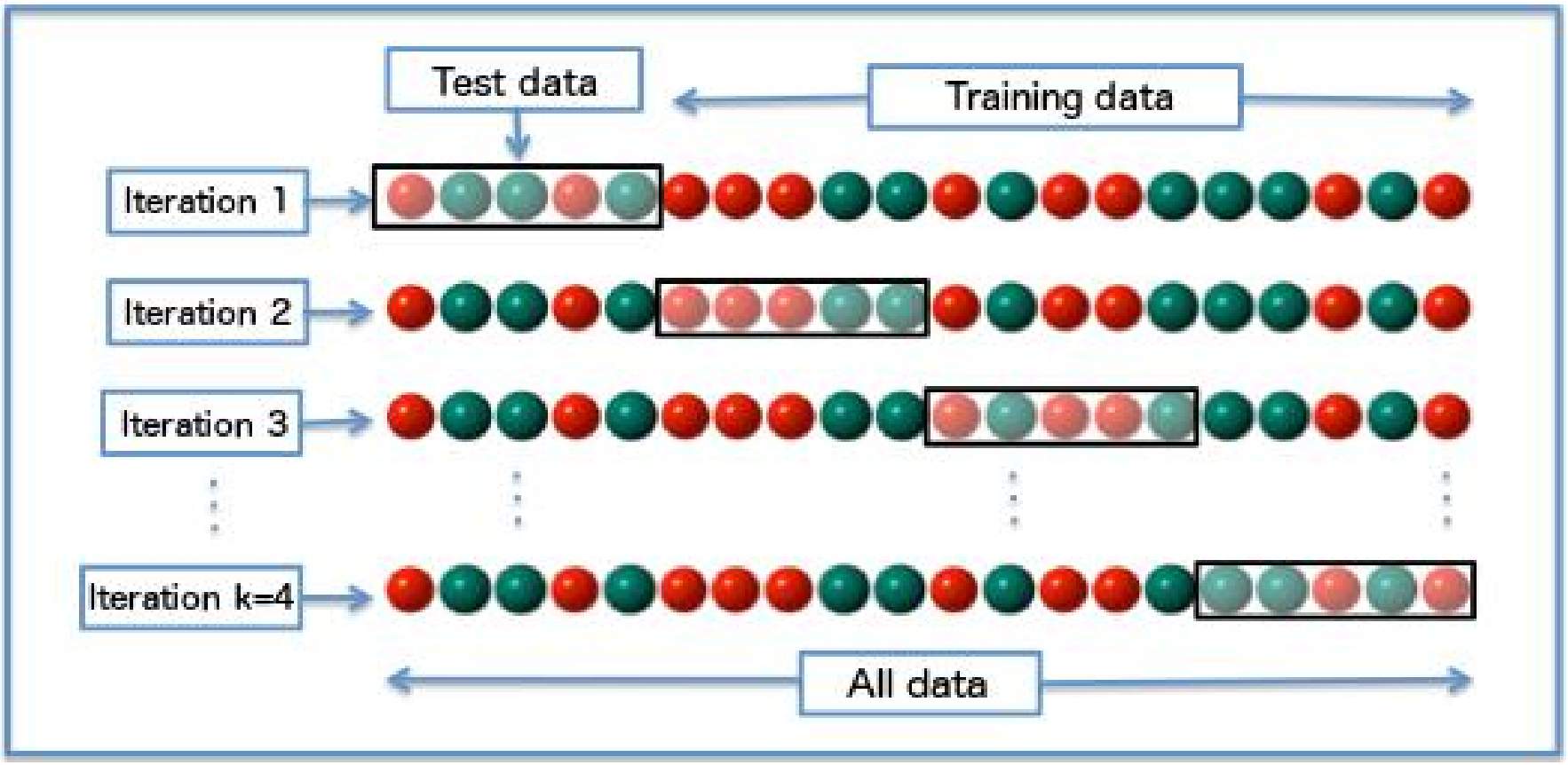
Effect of small sample size:

- Bias in the estimate
- Variance of estimate

# Methods of estimation

- Holdout
  - Reserve 2/3 for training and 1/3 for testing
- Random subsampling
  - Repeated holdout
- Cross validation
  - Partition data into k disjoint subsets
  - k-fold: train on k-1 partitions, test on the remaining one
  - Leave-one-out: k=n
- Stratified sampling
  - oversampling vs undersampling
- Bootstrap
  - Sampling with replacement

# Cross-validation



# Model evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

# ROC (Receiver Operating Characteristic)

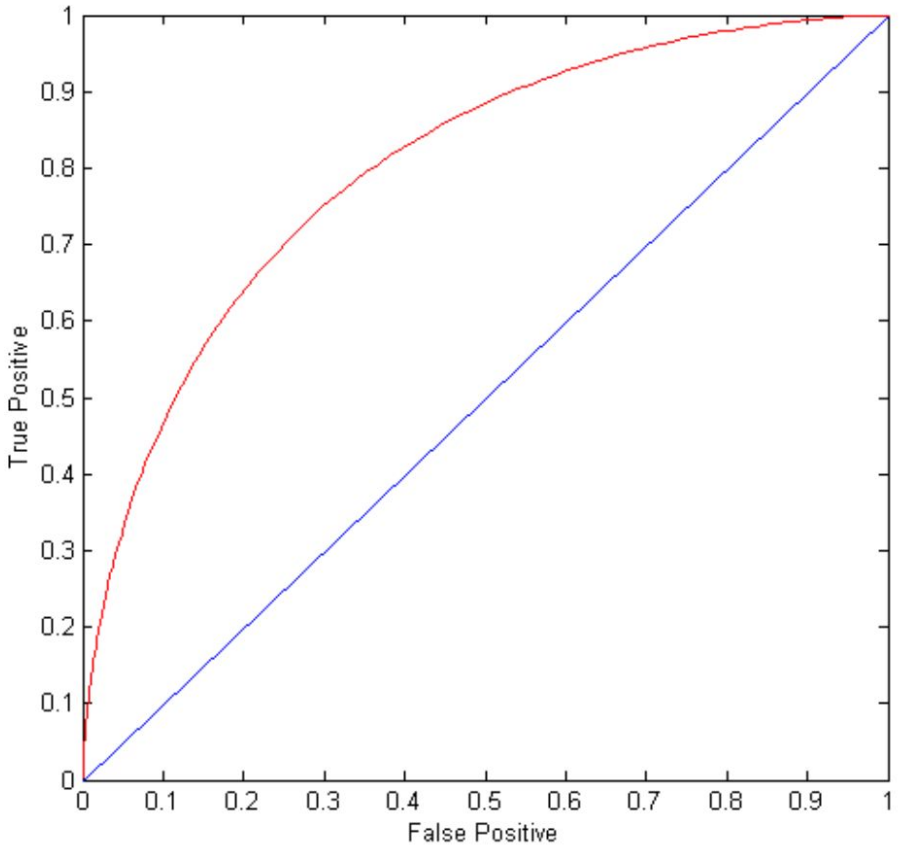
- ROC curve plots TP (on the y-axis) against FP (on the x-axis)
- Performance of each classifier represented as a point on the ROC curve
  - changing the threshold of algorithm, sample distribution or cost matrix changes the location of the point

# ROC curve

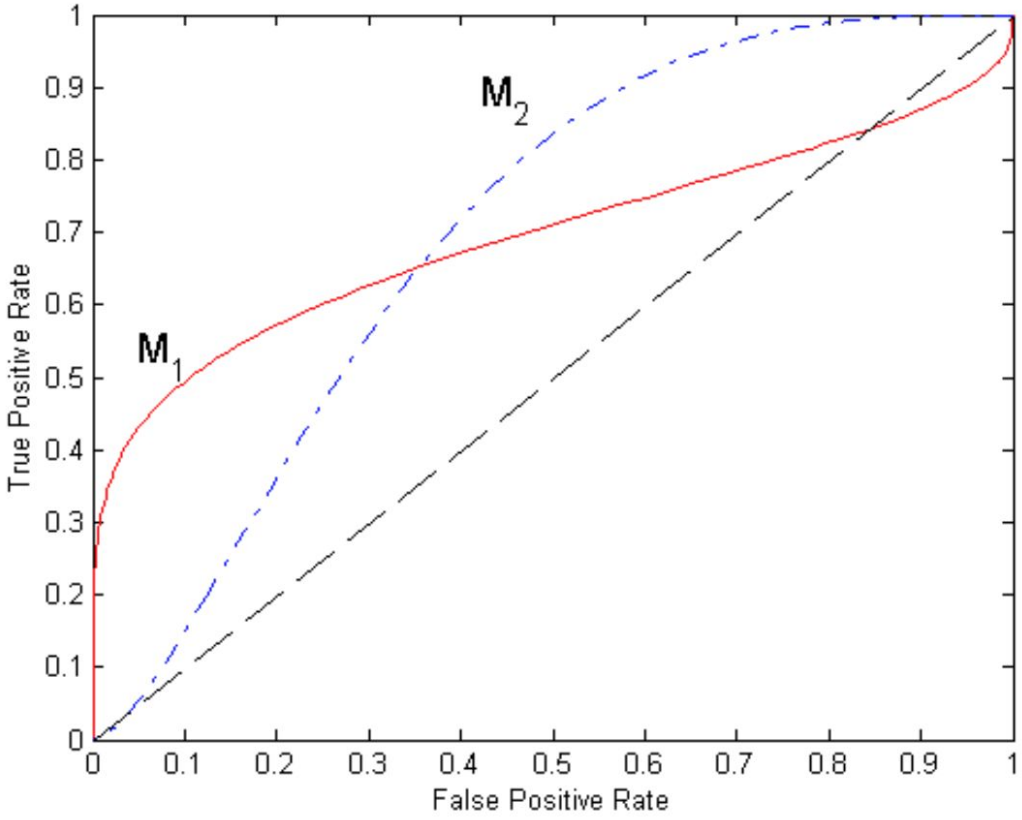
(TP,FP):

- (0,0): declare everything to be negative class
- (1,1): declare everything to be positive class
- (1,0): ideal

- Diagonal line:
  - Random guessing
  - Below diagonal line:
    - prediction is opposite of the true class

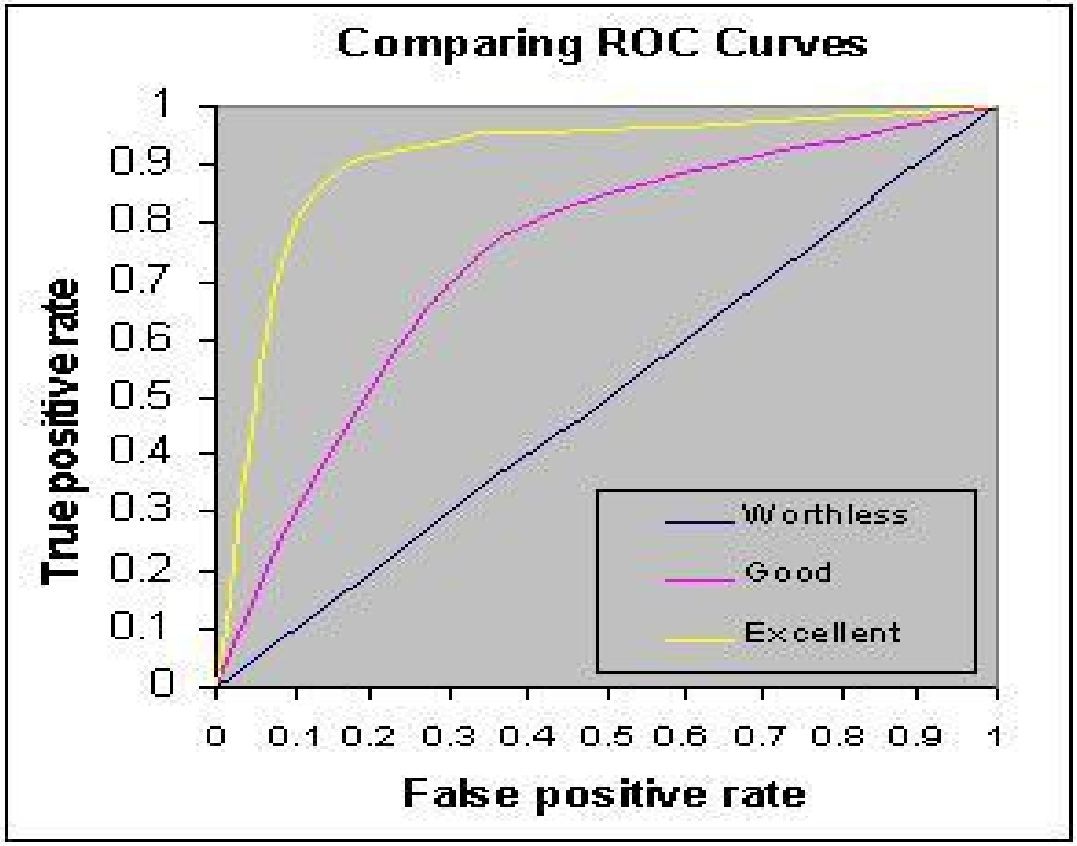


# Using Roc for model comparison




- No model consistently outperform the other
  - $M_1$ is better for small FPR
  - $M_2$ is better for large FPR
- Area Under the ROC curve
  - Ideal:
    - Area = 1
  - Random guess:
    - Area = 0.5

# Using Roc for model comparison



# How to construct a ROC curve

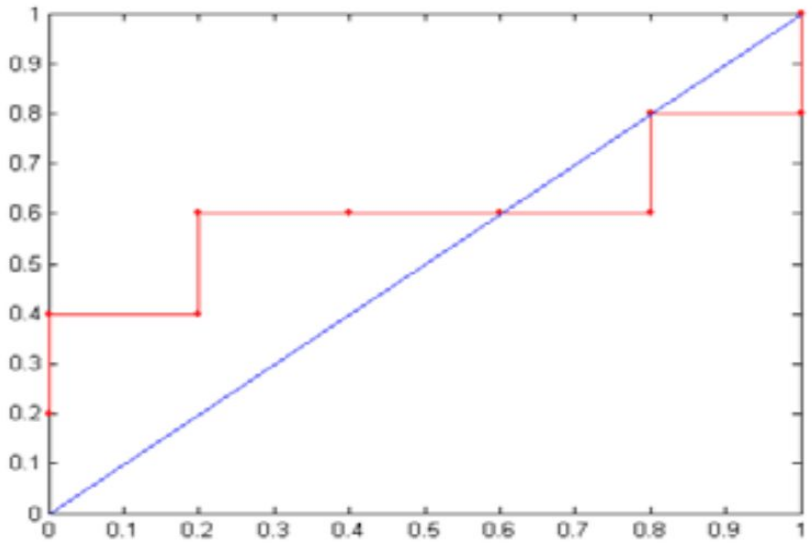
| Instance | P(+\|A) | True Class |
|---|---|---|
| 1 | 0.95 | + |
| 2 | 0.93 | + |
| 3 | 0.87 | - |
| 4 | 0.85 | - |
| 5 | 0.85 | - |
| 6 | 0.85 | + |
| 7 | 0.76 | - |
| 8 | 0.53 | + |
| 9 | 0.43 | - |
| 10 | 0.25 | + |

- Use classifier that produces posterior probability for each test instance P(+|A)
- Sort the instances according to P(+|A) in decreasing order
- Apply threshold at each unique value of P(+|A)
- Count the number of TP, FP, TN, FN at each threshold
- TP rate, TPR = TP/(TP+FN)
- FP rate, FPR = FP/(FP + TN)

# How to construct a ROC curve

| | Class | + | - | + | - | - | - | + | - | + | + | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Threshold >= | | 0.25 | 0.43 | 0.53 | 0.76 | 0.85 | 0.85 | 0.85 | 0.87 | 0.93 | 0.95 | 1.00 |
| | TP | 5 | 4 | 4 | 3 | 3 | 3 | 3 | 2 | 2 | 1 | 0 |
| | FP | 5 | 5 | 4 | 4 | 3 | 2 | 1 | 1 | 0 | 0 | 0 |
| | TN | 0 | 0 | 1 | 1 | 2 | 3 | 4 | 4 | 5 | 5 | 5 |
| | FN | 0 | 1 | 1 | 2 | 2 | 2 | 2 | 3 | 3 | 4 | 5 |
| → | TPR | 1 | 0.8 | 0.8 | 0.6 | 0.6 | 0.6 | 0.6 | 0.4 | 0.4 | 0.2 | 0 |
| → | FPR | 1 | 1 | 0.8 | 0.8 | 0.6 | 0.4 | 0.2 | 0.2 | 0 | 0 | 0 |

**ROC Curve:**



# What you do when you have large error?

- Get more training examples
- Try smaller sets of features
- Try additional features
- Try adding polynomial feature (complexity of model)
- Try decreasing or increasing your learning parameters
- Try different learning

# Individual assignment

**Decision tree evaluation with Python**

Try different learning parameters (gini index and information gain as criterion) to train varying decision trees from the given dataset.

Evaluate the accuracy of the decision trees trained on random sample with varying size (i.e, plot the learning curve)

save the trained model for the future use without having to retrain the decision tree model

use 10-fold cross validation for the evaluation of the decision trees

# Next class

**Ensemble methods:**

- general introduction to ensemble methods
- methods for building an ensemble classifier: bagging and boosting, random forests
- comparison among different classification methods

**Random forest with Python:**

- get the data
- split the data into a training and a test set
- train a random forest model from the training set
- evaluate the random forest model

# Resources


- Tan, Pang-Ning. *Introduction to data mining*. Pearson Education India, 2006.

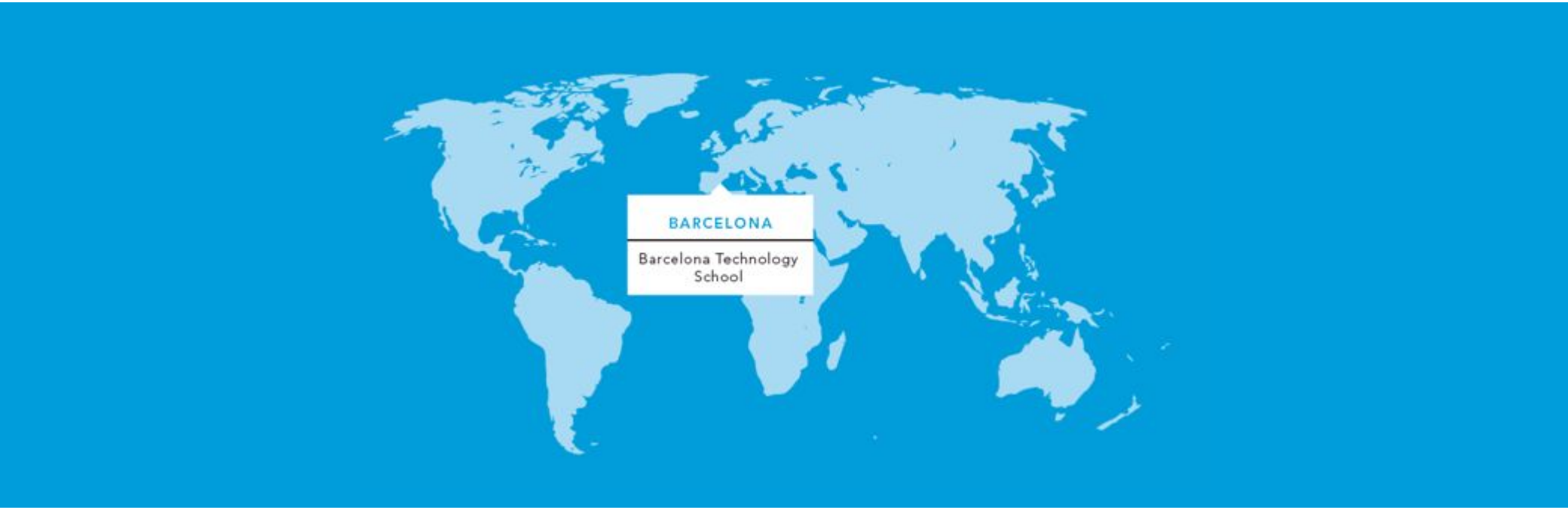
- Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. New York: Springer series in statistics, 2001.

- Examples of SVM with different learning parameters

https://chrisalbon.com/machine-learning/svc_parameters_using_rbf_kernel.html

BTS | Barcelona Technology School

eurecat Centre Tecnològic de Catalunya




# Thank you

Barcelona, 2017