LING 506 - TOPICS IN COMPUTATIONAL LINGUISTICS

Introductory Machine Learning

Yan Tang

Department of Linguistics, UIUC

Week 7

Last week…

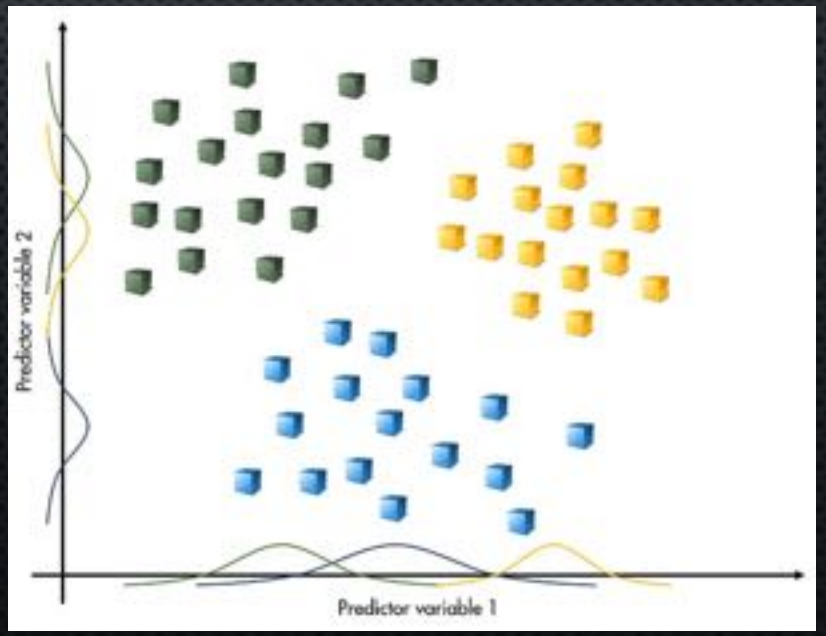
- K-nearest neighbours (KNN)
  - The simplest algorithm
    - An instance-based non-parametric approach
  - Using voting to assign the new observation to a class that has majority members in the neighbourhood constituting K observations
    - Rules to break tie when voting
  - Works well for data with low dimensions; not high dimensions
  - Does not work for data with mixed variable types

Last week…

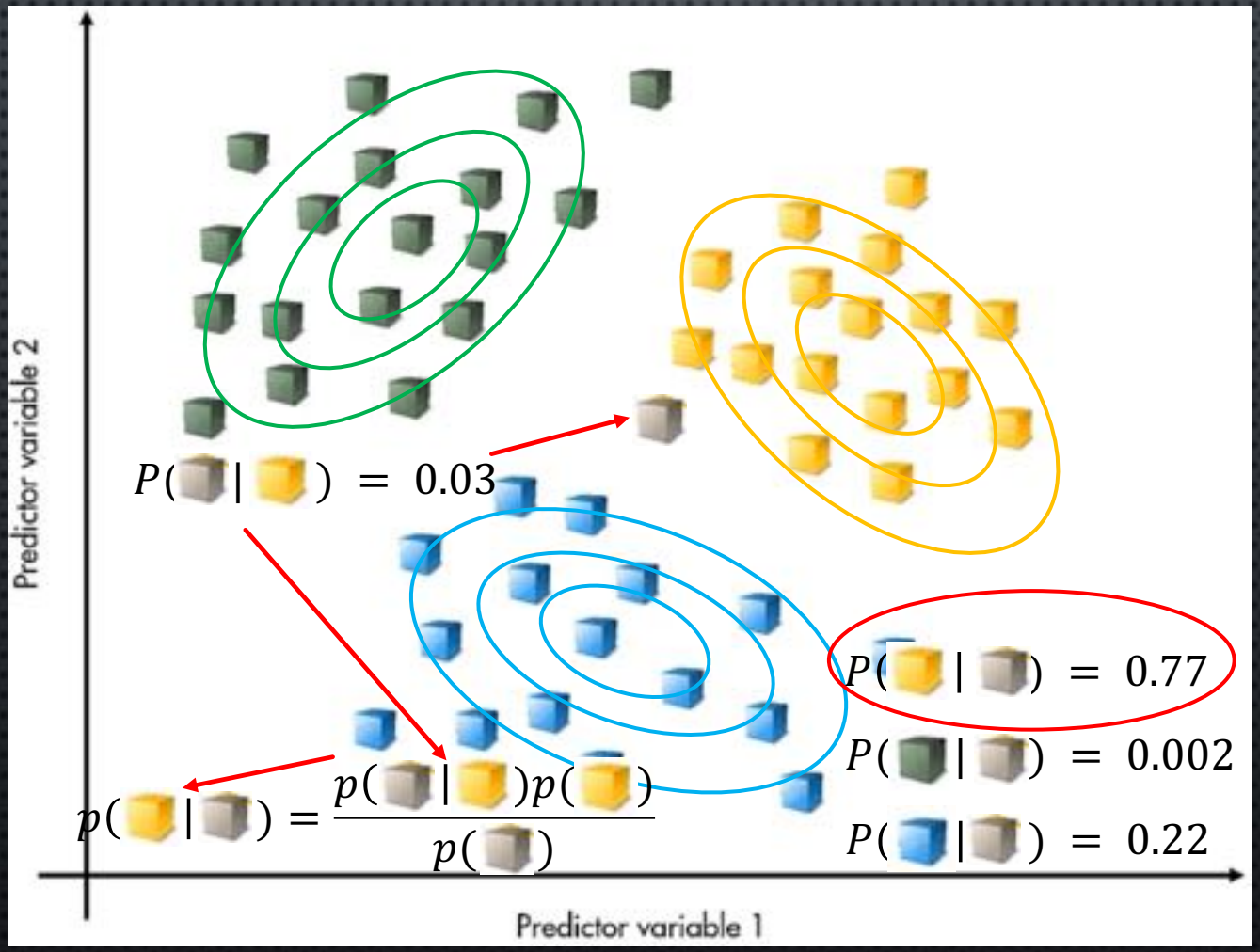
- Decision trees (DTs)
  - Classic technique for classification and regression
    - Non-parametric algorithm; further construct random forests
  - Builds a tree from all observations, down to individual brunches (classes) which contain minimal number of heterogenous members
  - Requires little pre-processing and works with mixed variable types
  - Prone to overfitting; sensitive to variations in data

Naïve Bayes classification

- Assumes the data comes from a certain underlying distribution
  - Each observation is a statistical sample
  - Reduce the influence of outliers



Naïve Bayes classification



Naïve Bayes: principle

Assumption: predictors are independent on each other

- i.e. a presence of of variable in a class is unrelated to the presence of any other variables

Bayes theorem: a way to calculate posteriori probability for an observation in a class, c, given its features $X = [x_1, \dots, x_n]$:

$$P(c|X) = \frac{P(X|c)P(c)}{P(X)}$$

- $P(c)$: prior probability of a class
- $P(X|c)$: probability of features given a class
- $P(X)$: prior probability of features

Naïve Bayes: principle

With the naïve assumption:

$$P(c|X) = \frac{P(X|c)P(c)}{P(X)}$$

$$P(c|X) = \frac{P(c)\prod_{i=1}^{n} P(x_i|c)}{\prod_{i=1}^{n} P(x_i)}$$

$\prod_{i=1}^{n} P(x_i)$ is constant given a set of X:

$$\therefore P(c|X) \propto P(c)\prod_{i=1}^{n} P(x_i|c)$$

The observation is assigned to the class in which it has highest probability

$$\hat{c} = \arg\max_{c} p(c)\prod_{i=1}^{n} P(x_i|c)$$

Naïve Bayes classification: example:

Text	Class
Very good	Positive
Pretty bad	Negative
Very exciting	Positive
Not exciting	Negative
Very bad	Negative
Pretty good	Positive

"Very pretty" ?

Naïve Bayes classification

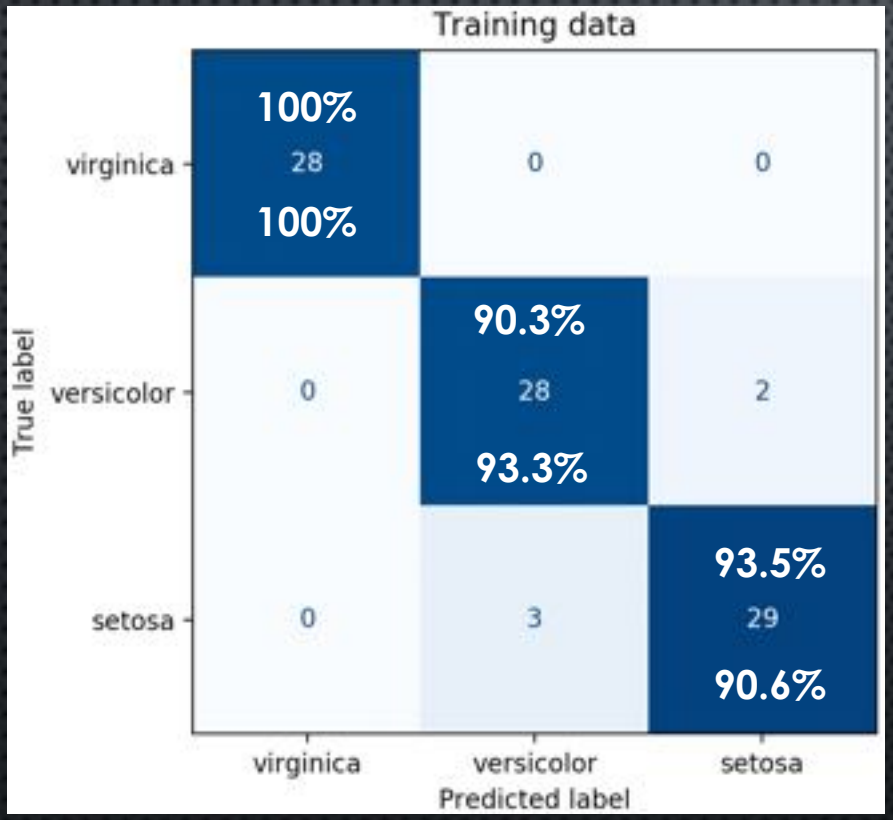
- Naïve Bayes classifiers:
  - Gaussian Naïve Bayes
    - Any continuous data
  - Multinomial Naive Bayes (MNB)
    - Suitable for discrete data, e.g. integer count of an incident (word occurrence in document)
  - Bernoulli Naïve Bayes
    - Suitable for binary data
  - …
- Main difference: the assumptions they make regarding the distribution of $P(x_i|c)$

Naïve Bayes classification: Iris data

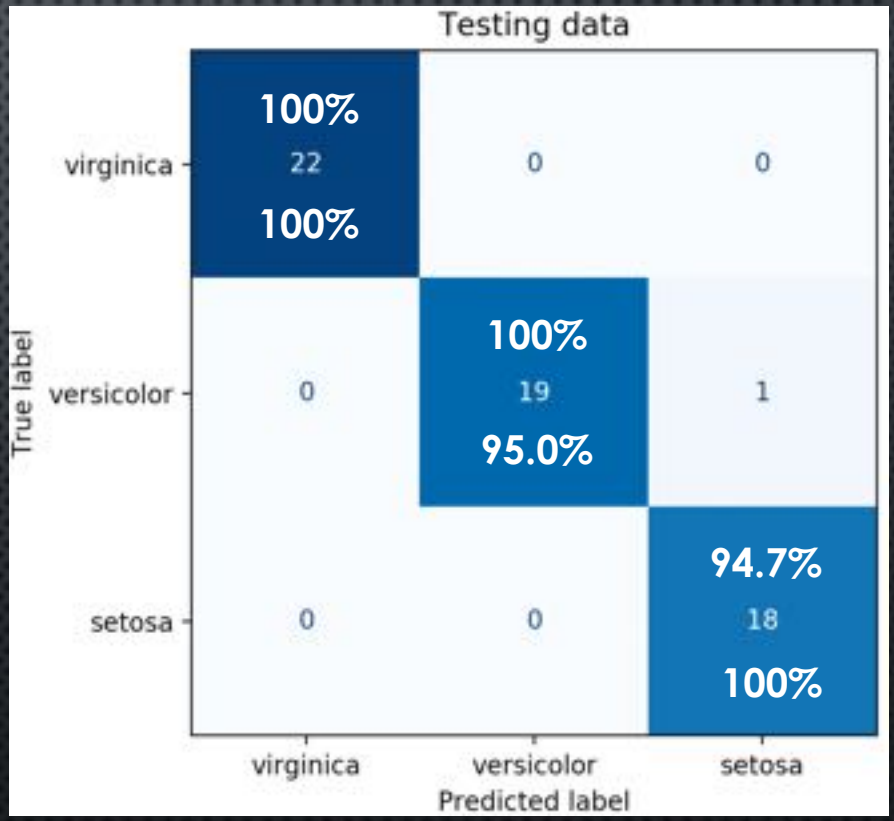
- No. of samples = 150
- Classes = {"*setosa*", "*versicolor*", "*virginica*"}
- Training : testing = 0.6 : 0.4
- Algorithm: Gaussian Naïve Bayes

Naïve Bayes classification: Iris data

Error rate: 5.6%



Error rate: 1.7%

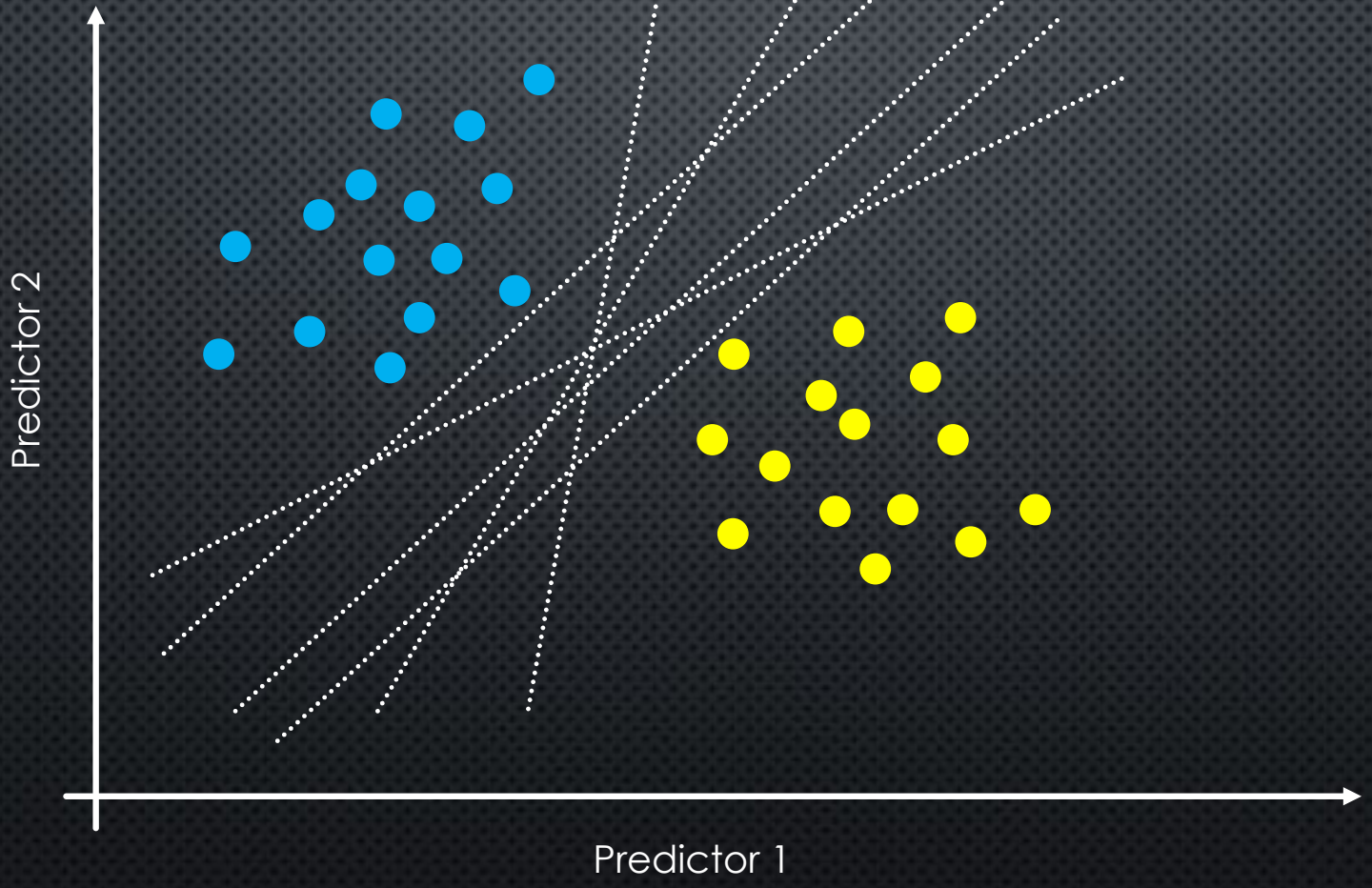


Precision
Count
Recall

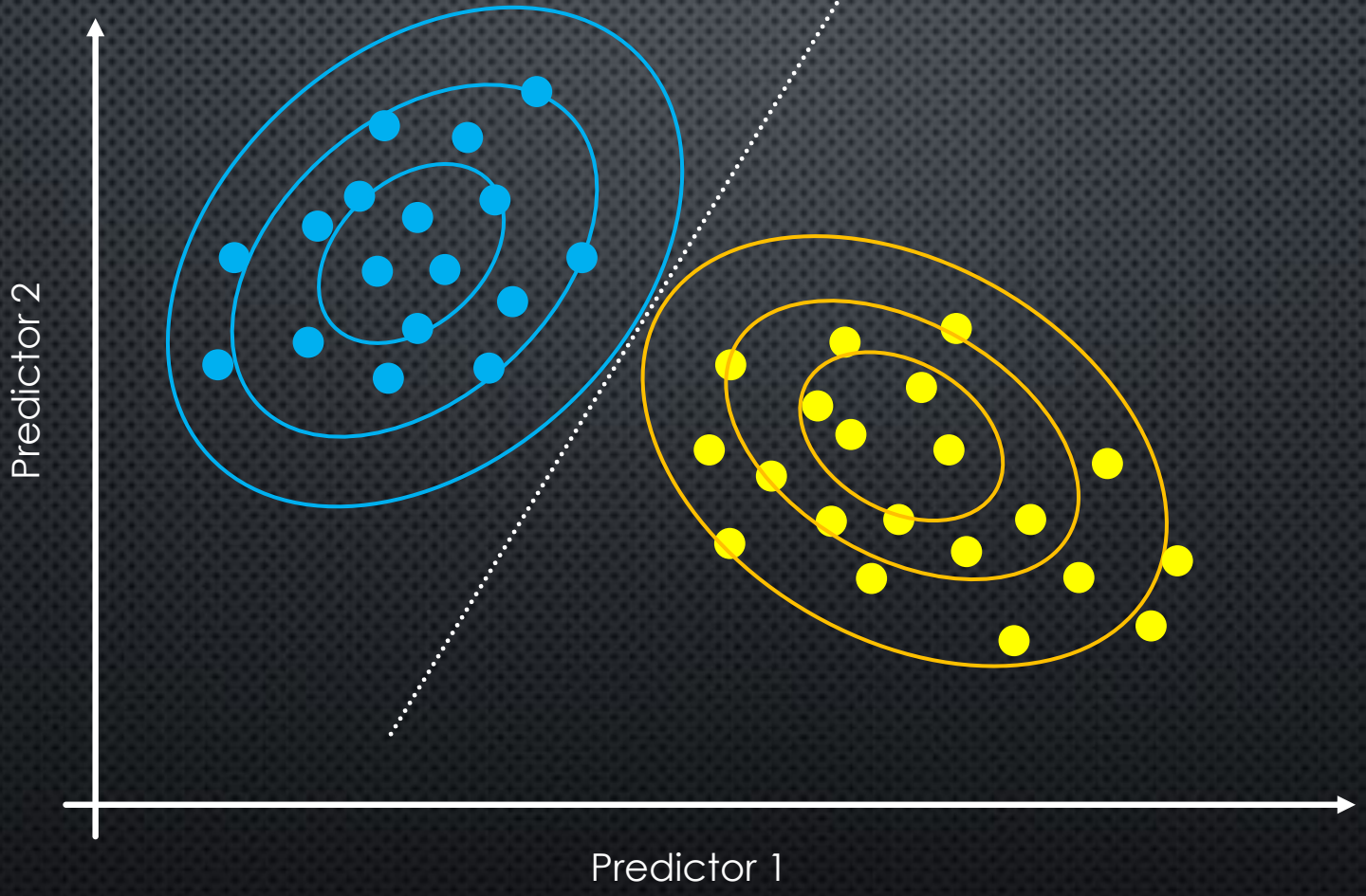
Naïve Bayes classification: pros and cons

- Fast training (requiring less data) and prediction making
- Works well with high-dimensional sparse data; robust to parameters
- Good as baseline model for multiclass prediction
- The fundamental assumption is too "naïve"
- "Zero frequency"

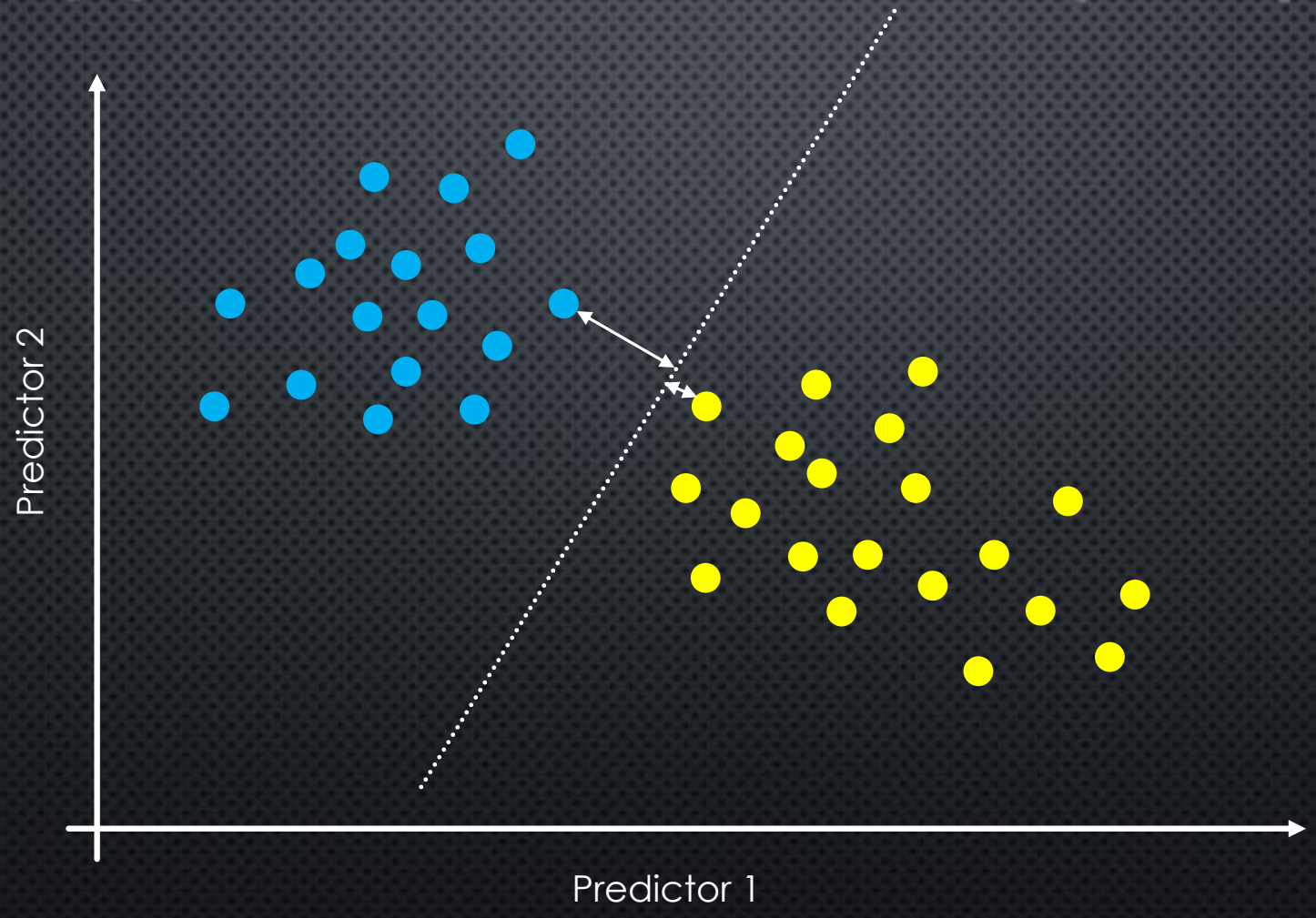
Support vector machines (SVM)



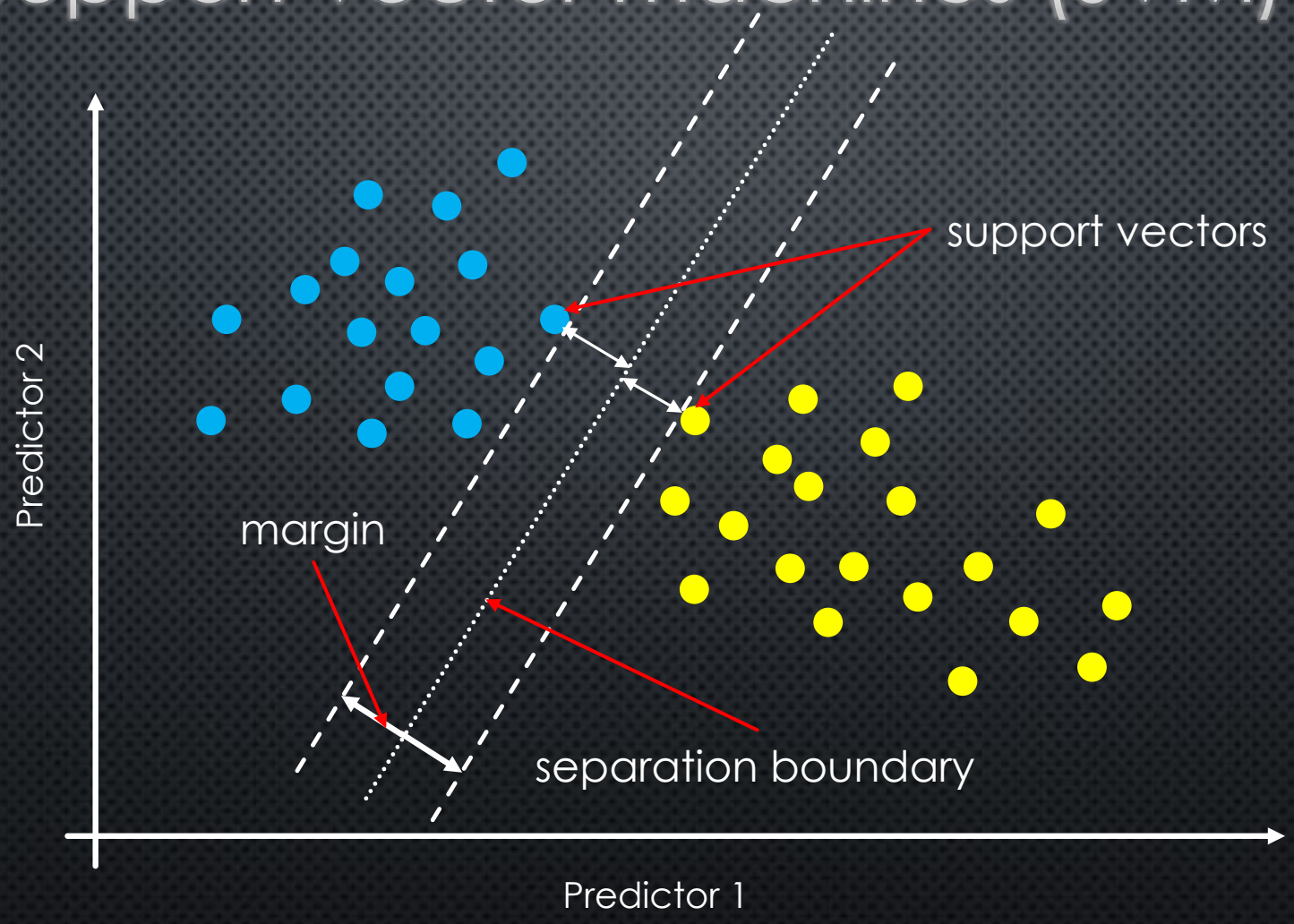
Support vector machines (SVM)



Support vector machines (SVM)

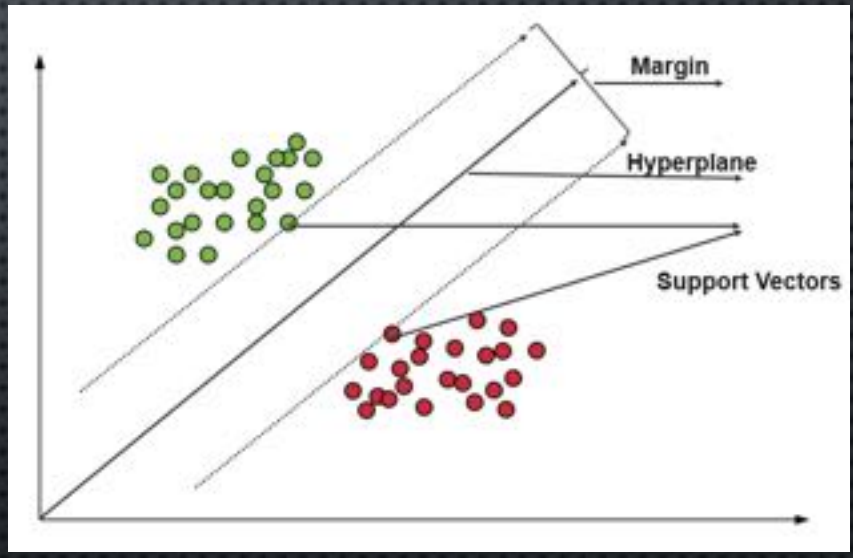


Support vector machines (SVM)



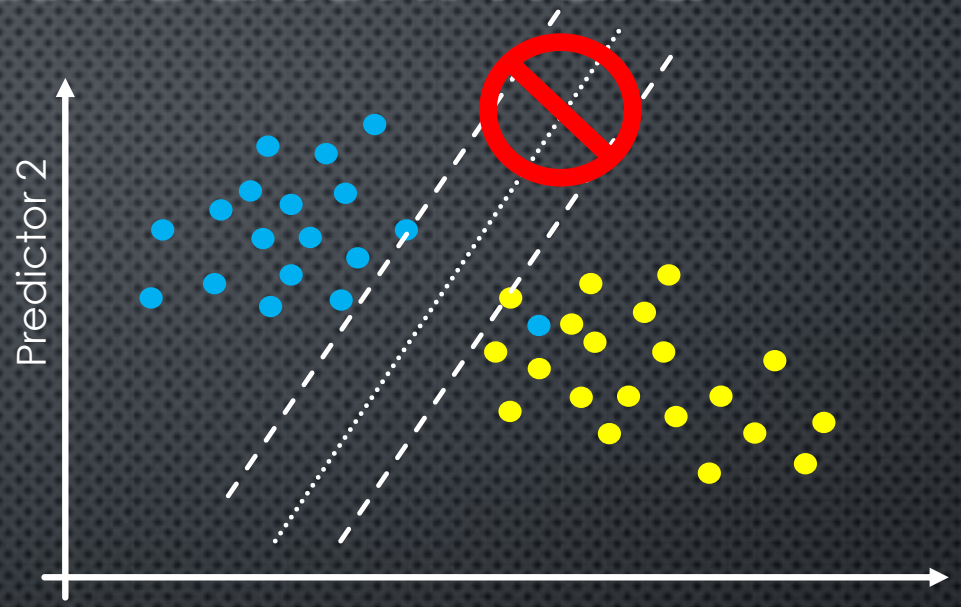
Linear SVM classification

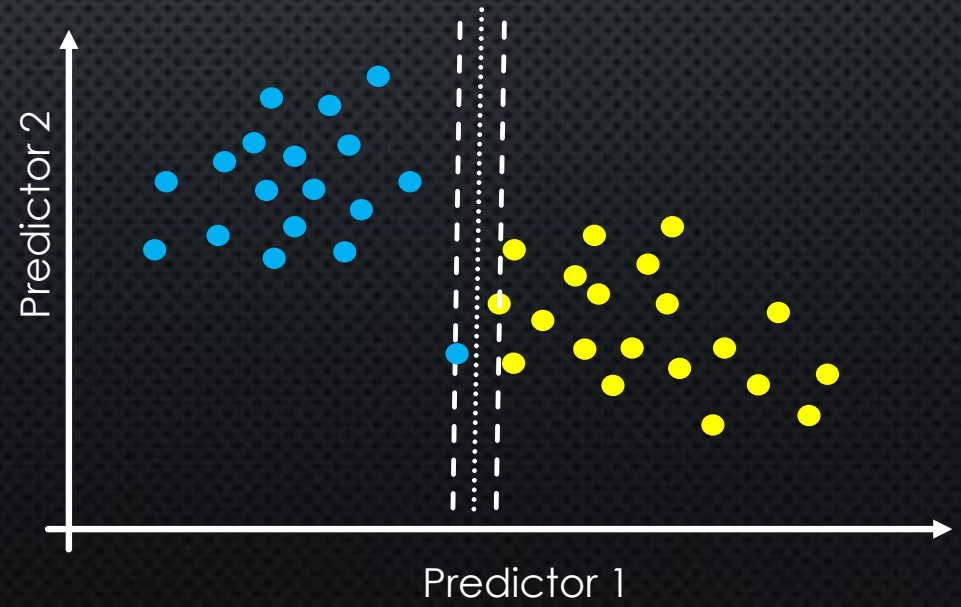
- Think of SVM classifier as fitting the widest possible path between the classes (support observations!!)
- Observations on the margin are support vectors
- The location of decision boundary is independent on off-margin observations, i.e. non-support vectors



Linear SVM classification: hard margin

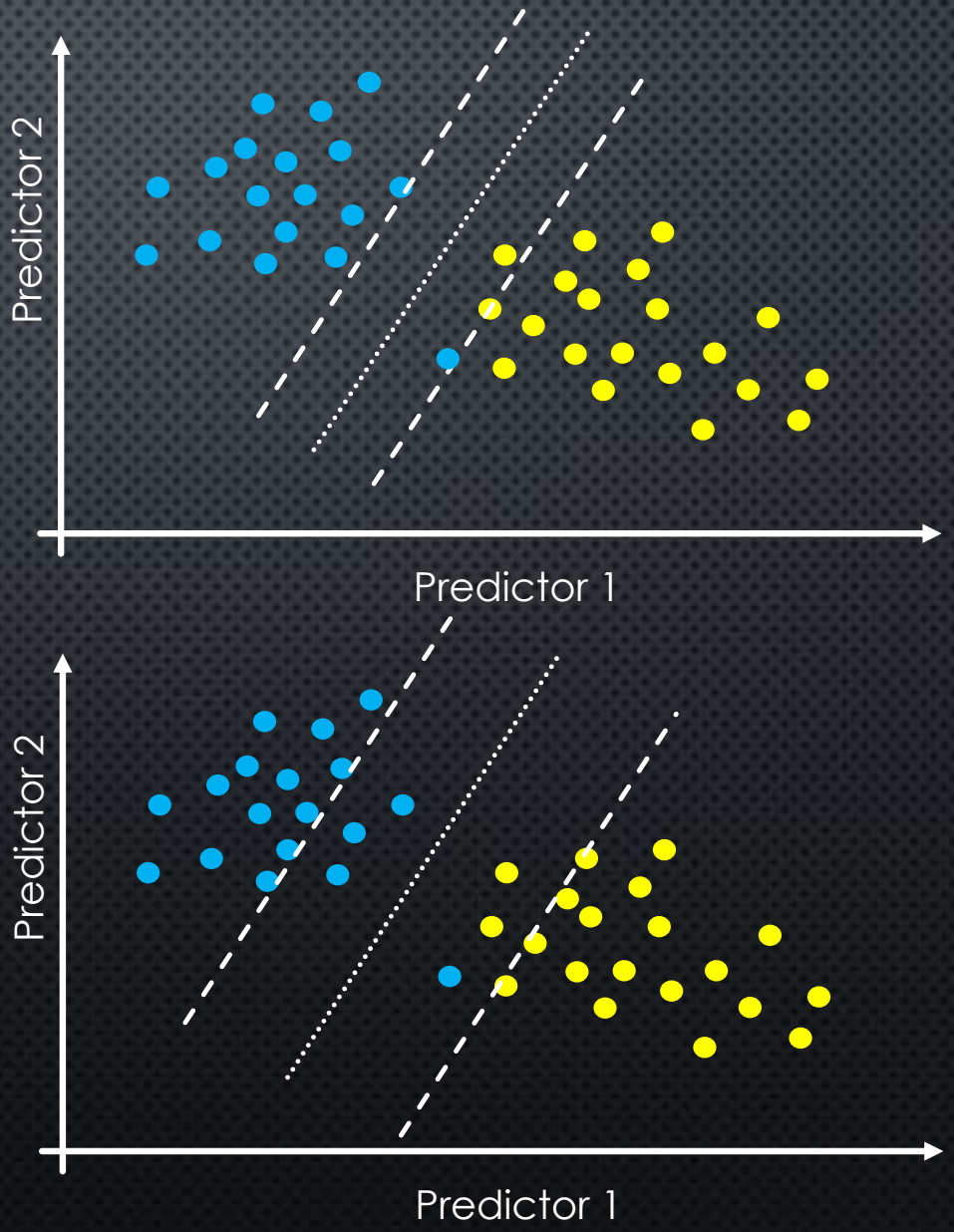
- All observations must be behind the margin on their side
- Issues:
  - Only works if the data is linearly separable
  - Sensitive to outliners





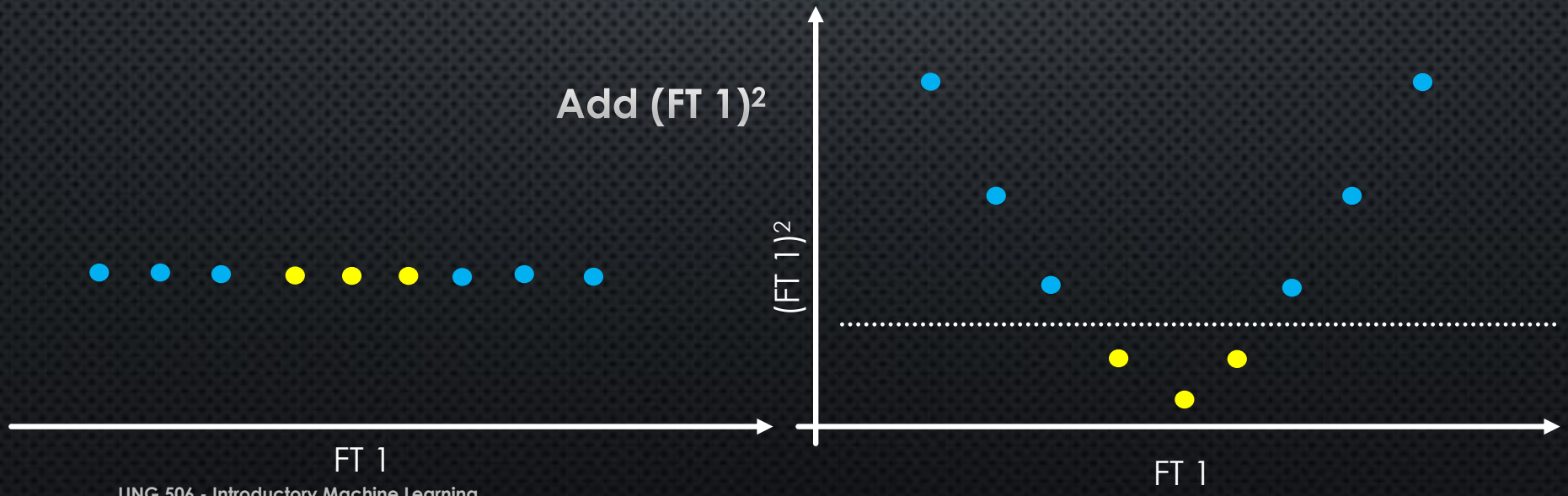
Linear SVM classification: soft margin

- Good balance between keeping margin as large as possible and limiting the number of observations crossing the margin or boundary
- Hyperparameter C
  - Larger C: smaller margin, less margin violation
  - Smaller C: larger margin, more margin violation



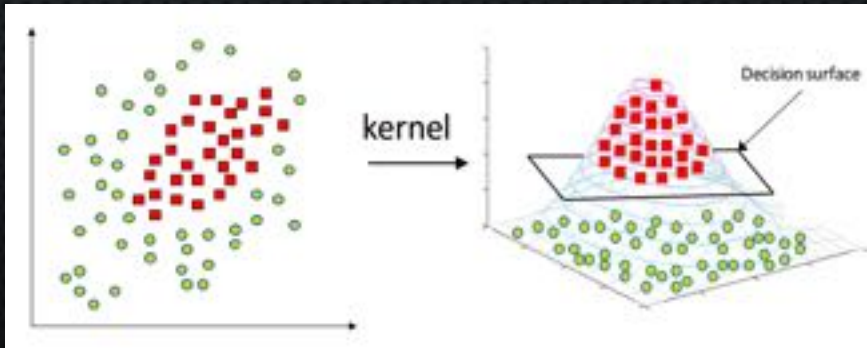
Non-linear SVM classification

- Dataset most of time are not linearly separable
  - What can be done?
  - Adding features to make it or close to be linearly separable
  - Non-linear features: polynomial features



Non-linear SVM classification

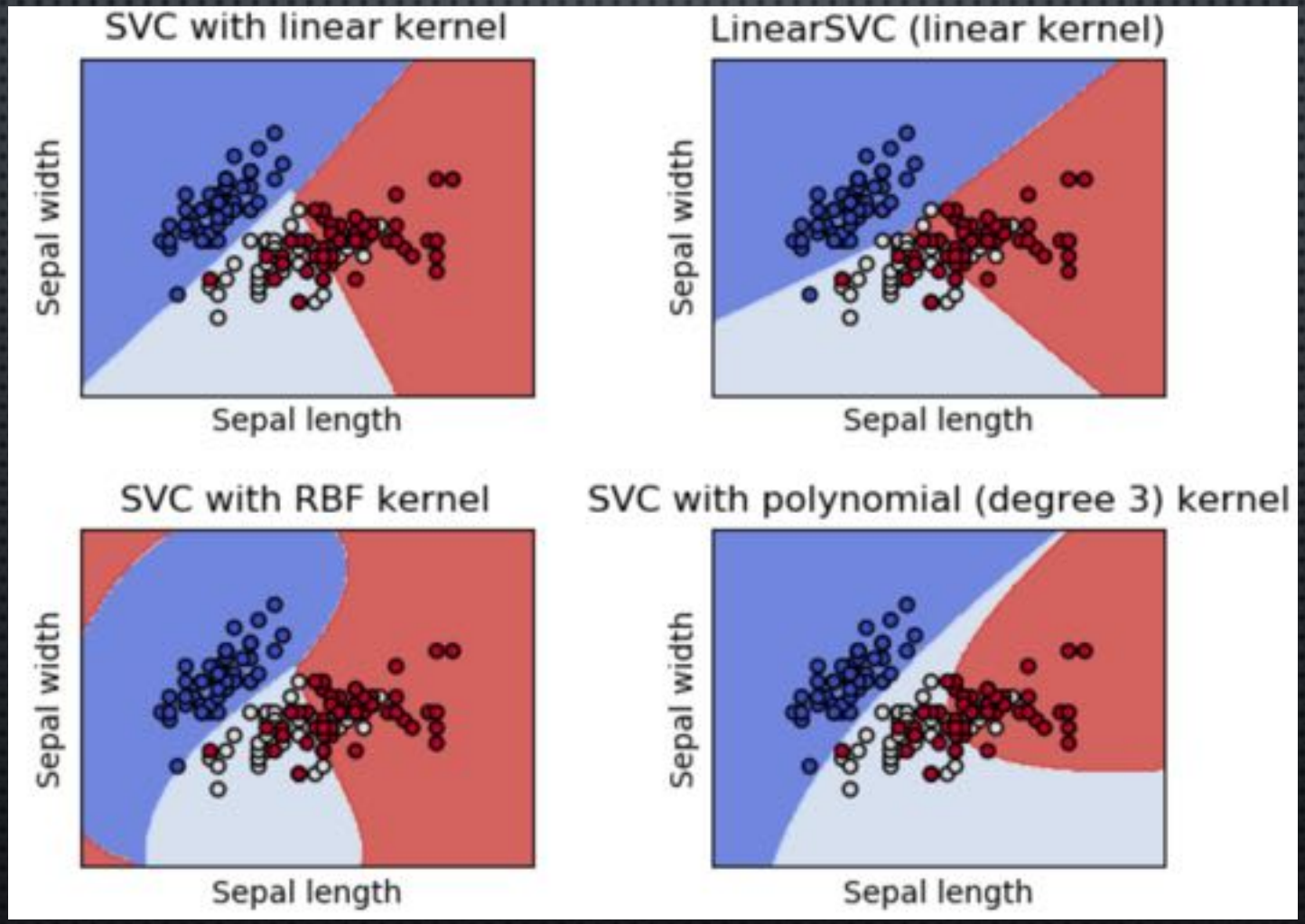
- Limitations of adding polynomial features
  - Cannot deal with very complex datasets
  - Introduce too many features when polynomial degree is high
- Kernel tricks:
  - Use distance or similarity of the data for higher dimensional feature representation
  - Do not introduce extra features



Non-linear SVM classification

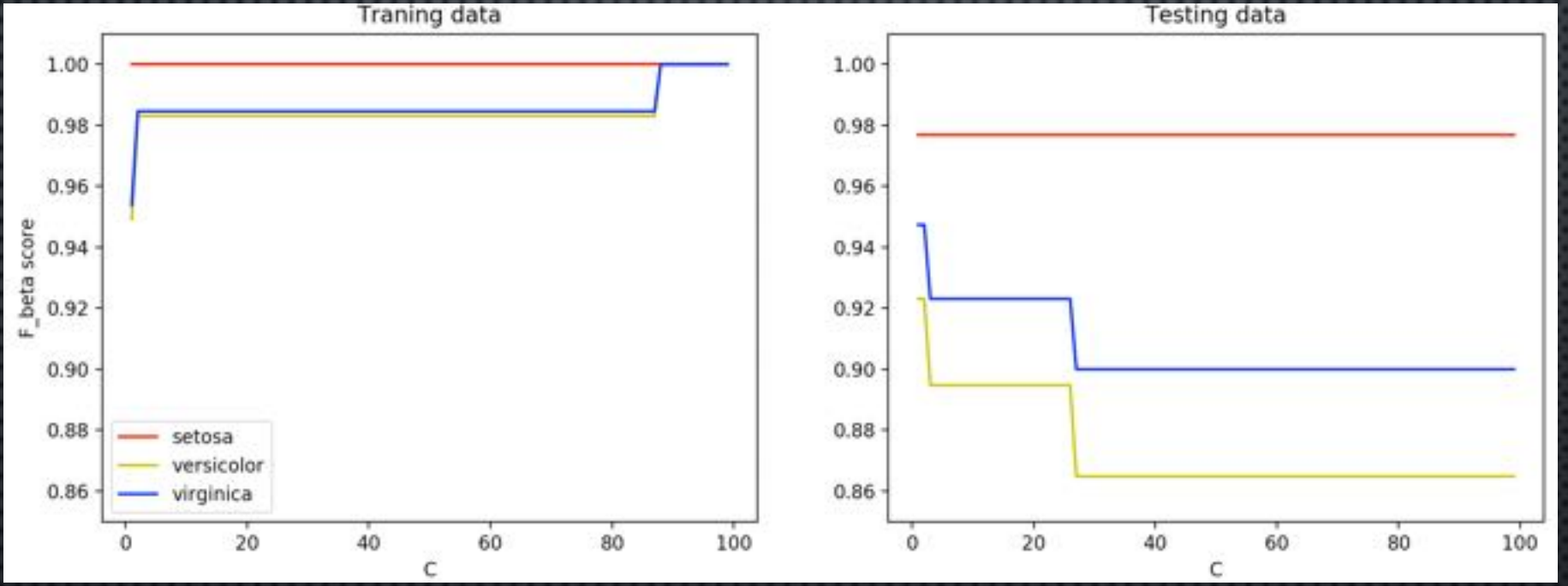
- Kernel functions:
  - Linear kernel
  - Polynomial kernel
  - Gaussian Radial Basic Function (RBF)
  - Sigmoid kernel
  - etc

SVM: Iris data



https://scikit-learn.org/stable/auto_examples/svm/plot_iris_svc.html#sphx-glr-auto-examples-svm-plot-iris-svc-py

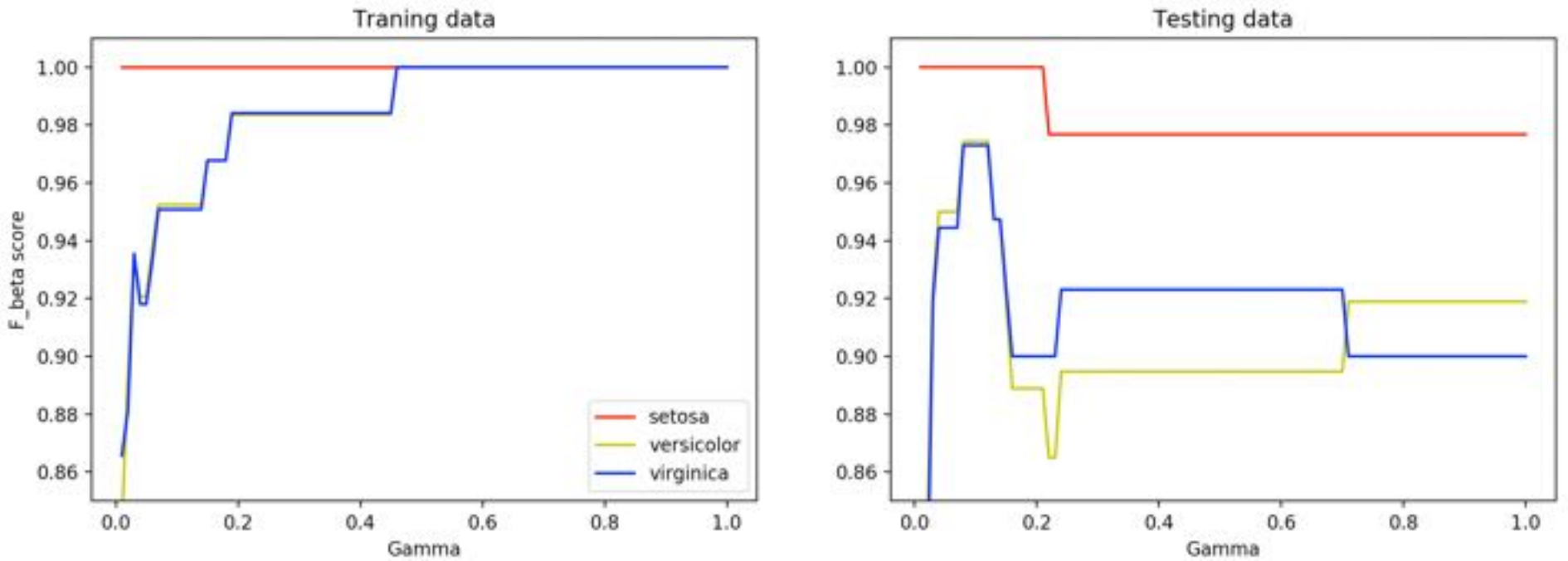
SVM: Iris data



Algorithm: Linear SVM

C = [1, 99]

SVM: Iris data



Algorithm: Kernelised SVB

Kernel function: Gaussian RBF

Gamma = [0.1, 1]

https://scikit-learn.org/stable/auto_examples/svm/plot_iris_svc.html#sphx-glr-auto-examples-svm-plot-iris-svc-py

SVM: summary

- Advantages:
  - Works well on low- and high dimensional data
  - Allows complex decision boundaries with limited number of features
  - Provides different type of kernel functions, as well as custom kernel functions for problems
- Disadvantages:
  - Sensitive to feature scales
  - Performance may rely on hyperparameter tuning, e.g C and gamma
  - Less good scalability