Performing Kernel Approximations

Janani Ravi
CO-FOUNDER, LOONYCORN

www.loonycorn.com

Overview

Understanding the need for kernel approximations

Feature mappings to approximate specific kernels

Nystroem method

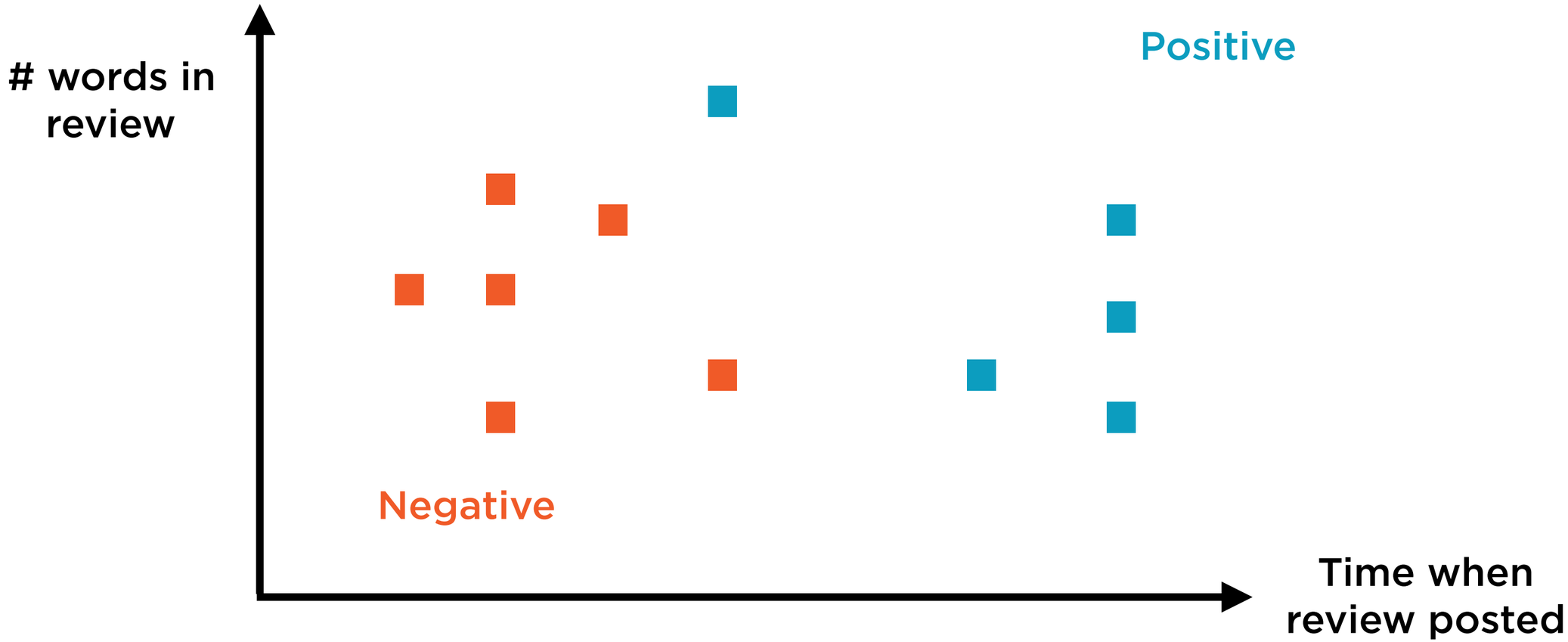
Radial Basis Function (RBF) method

Contrasting SVMs with full kernel to simpler kernels with approximation

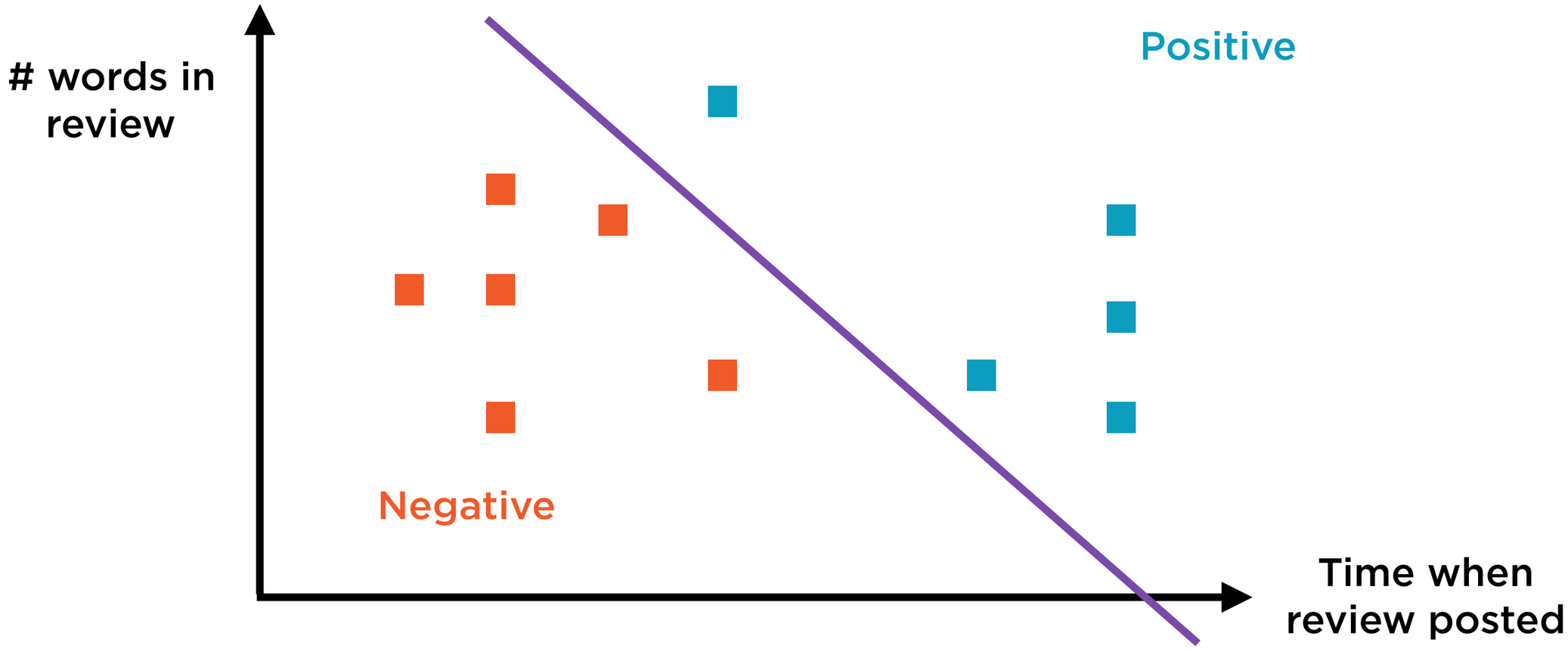
Classification using Support Vector Machines

Data in Two Dimensions

Data in Two Dimensions

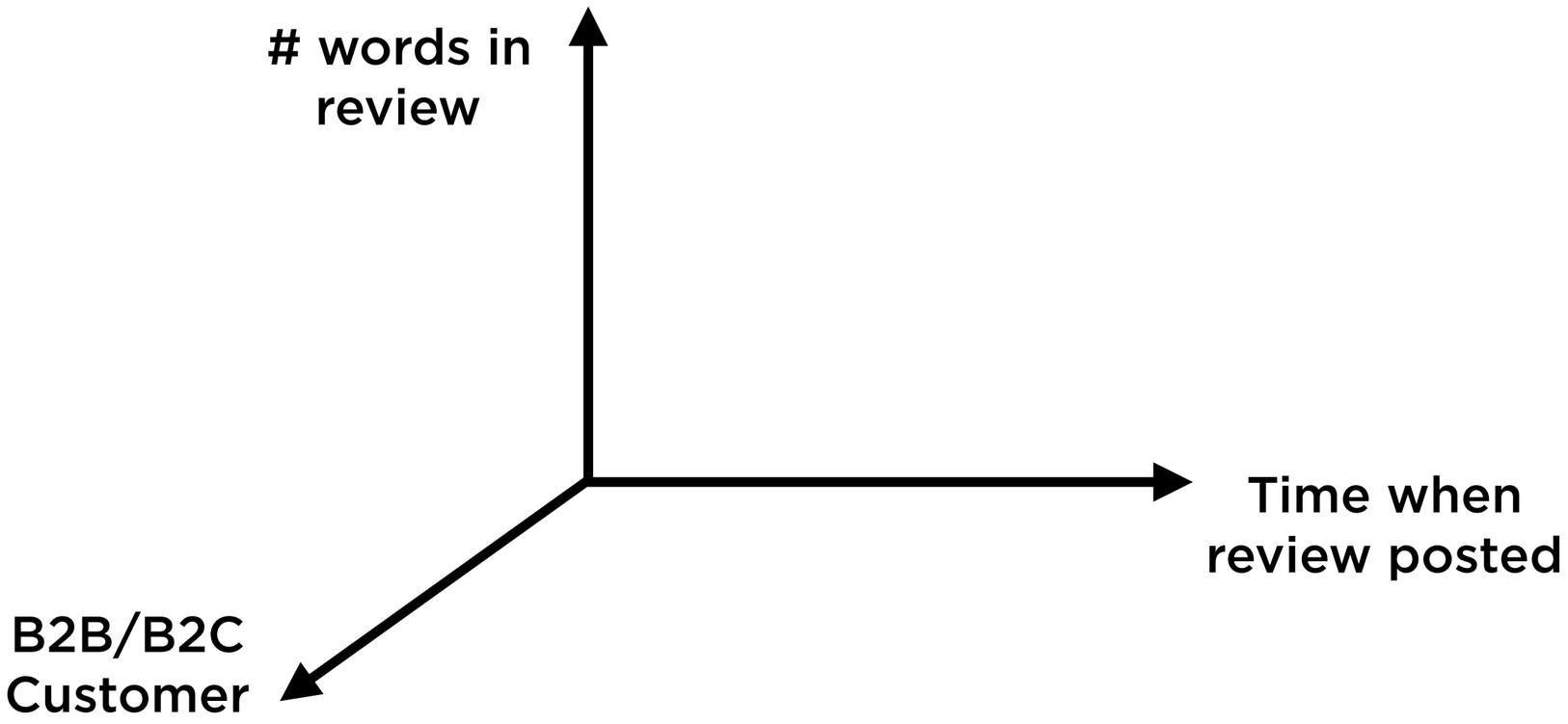


Data in Two Dimensions



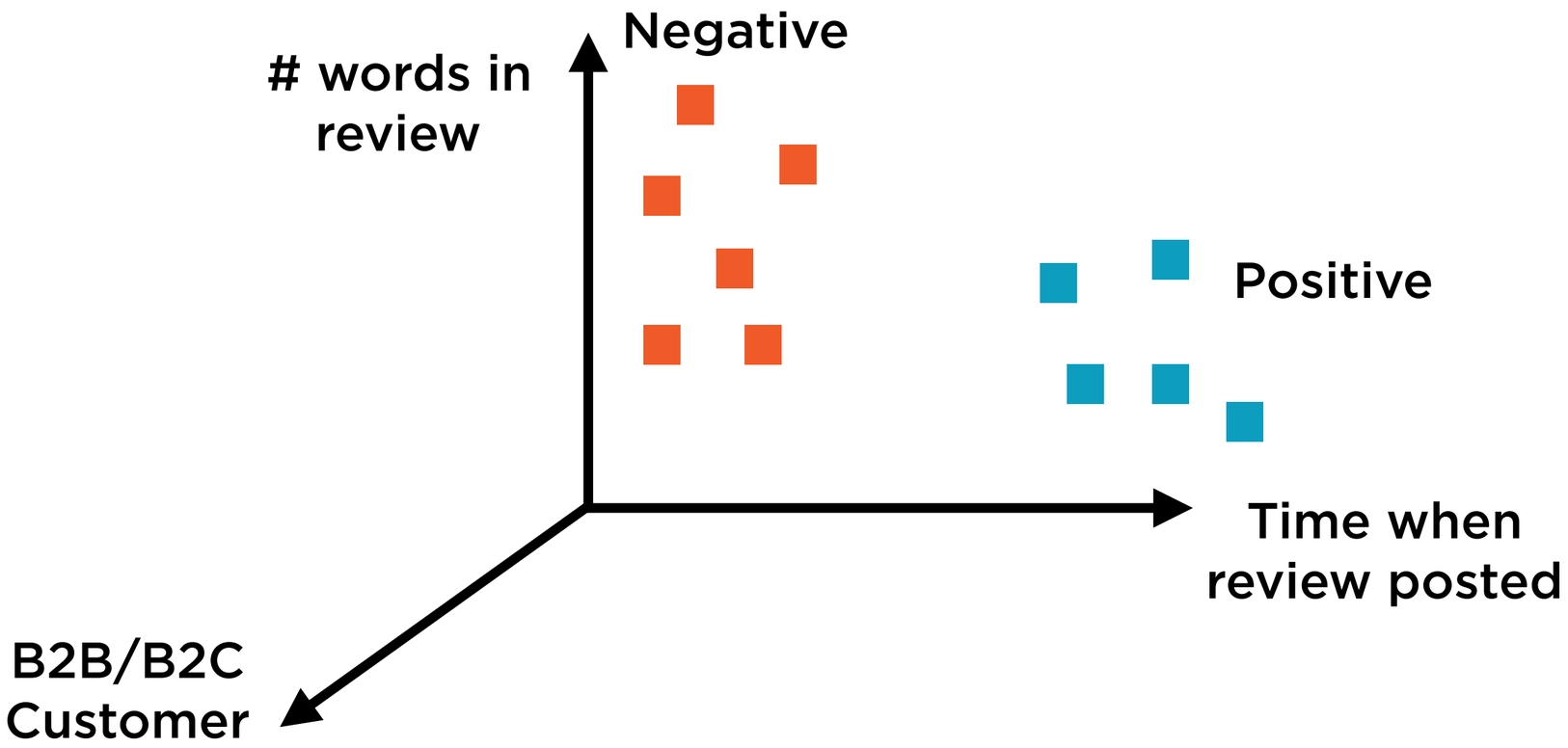
Bidimensional data points can be represented using a plane, and classified using a line

Data in N Dimensions



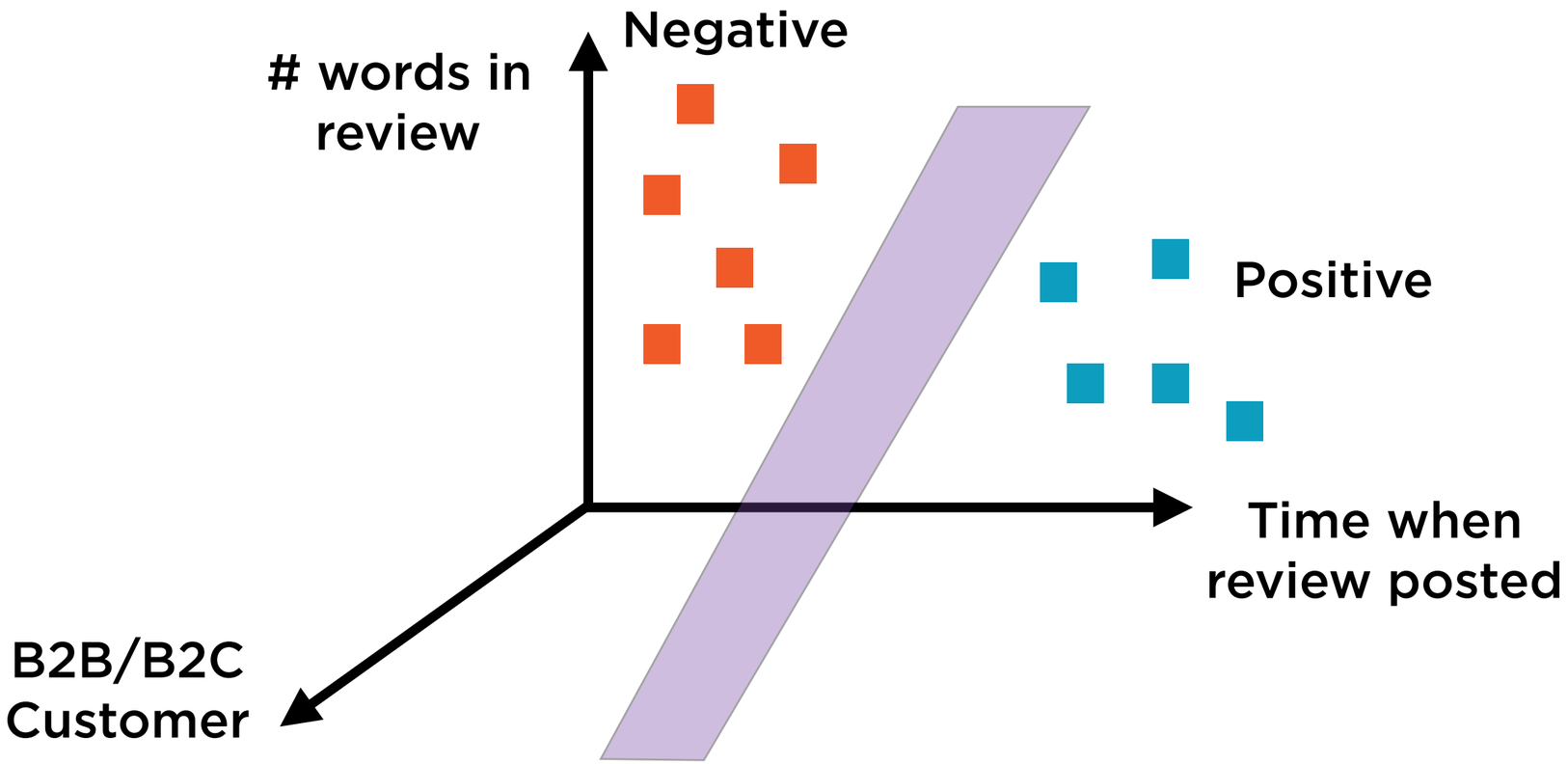
N-dimensional data can be represented in a hypercube, and classified using a hyperplane

Data in N Dimensions



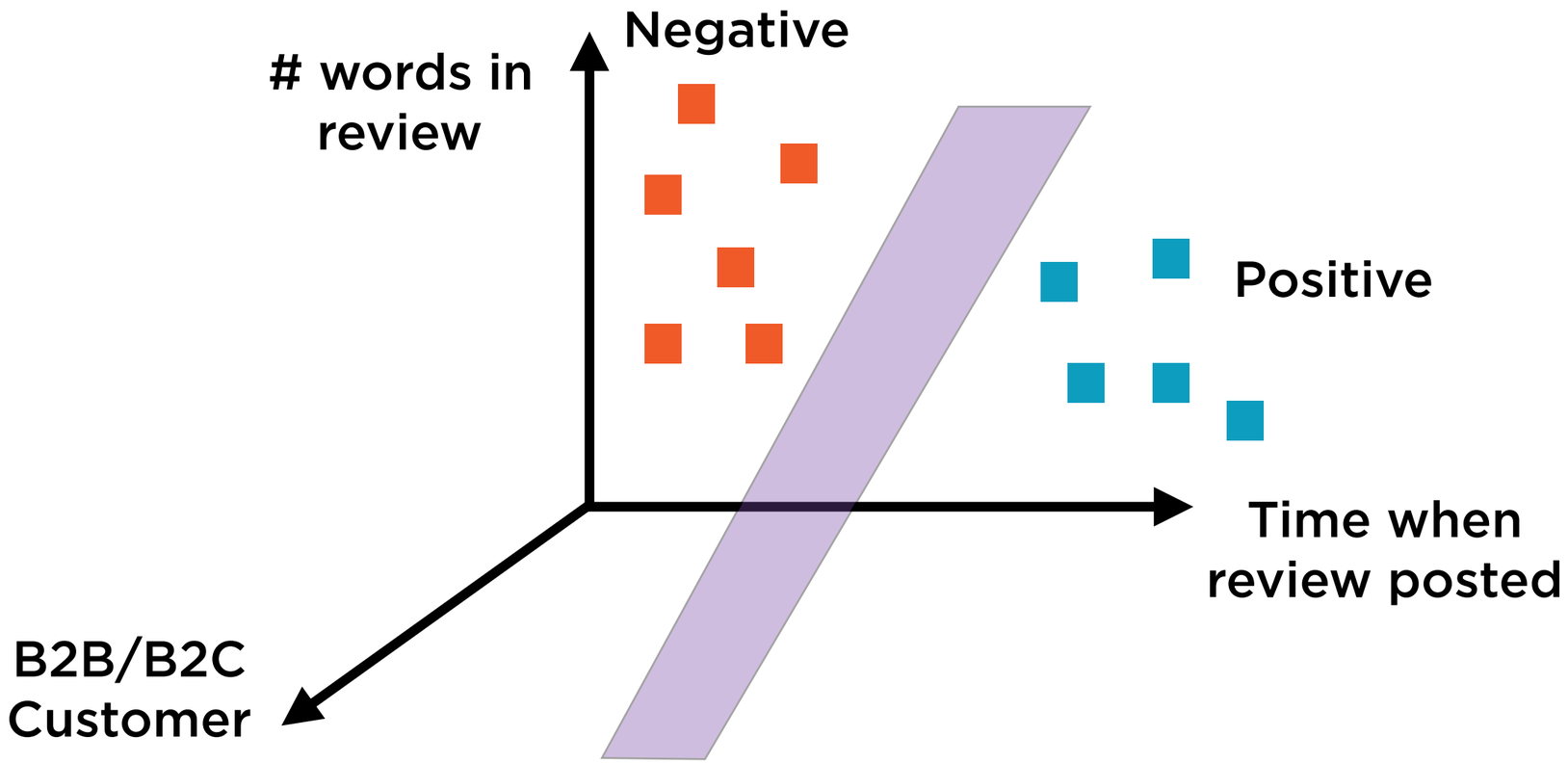
N-dimensional data can be represented in a hypercube, and classified using a hyperplane

Data in N Dimensions



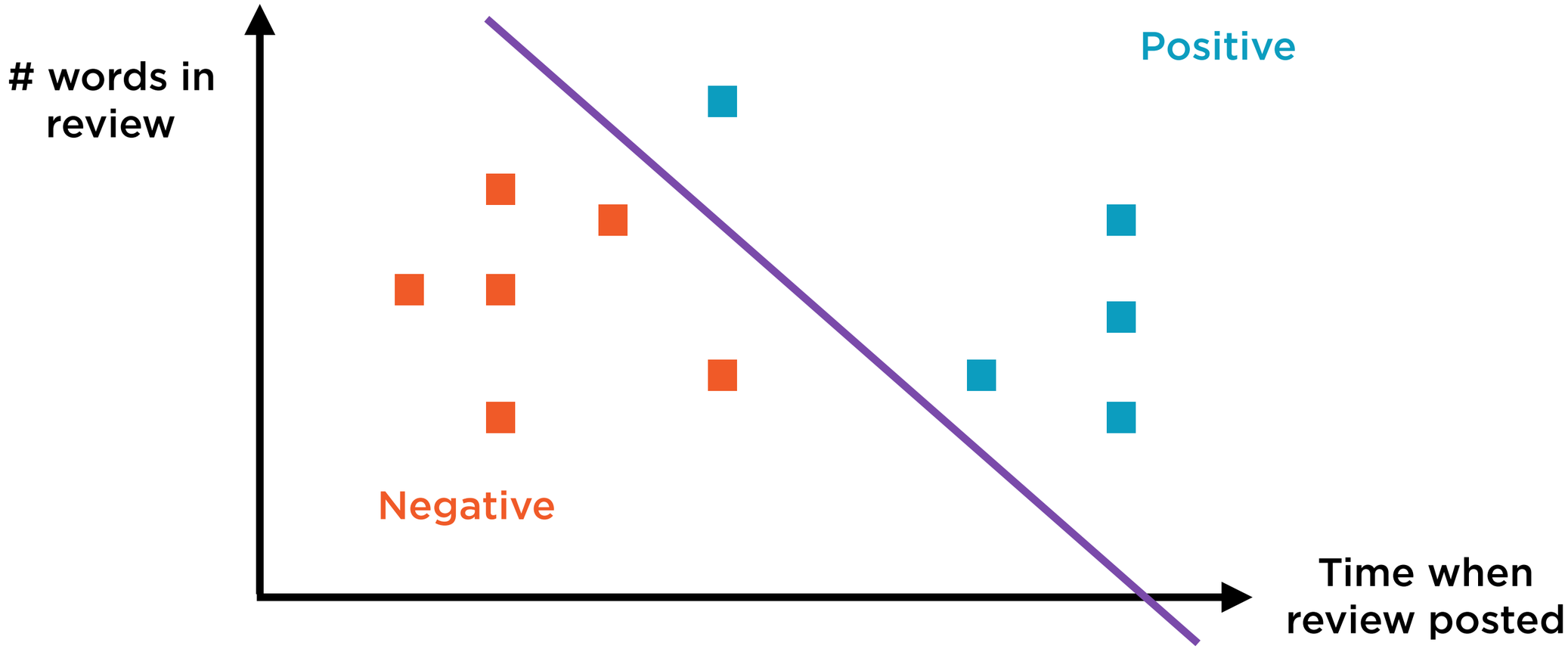
N-dimensional data can be represented in a hypercube, and classified using a hyperplane

Support Vector Machines



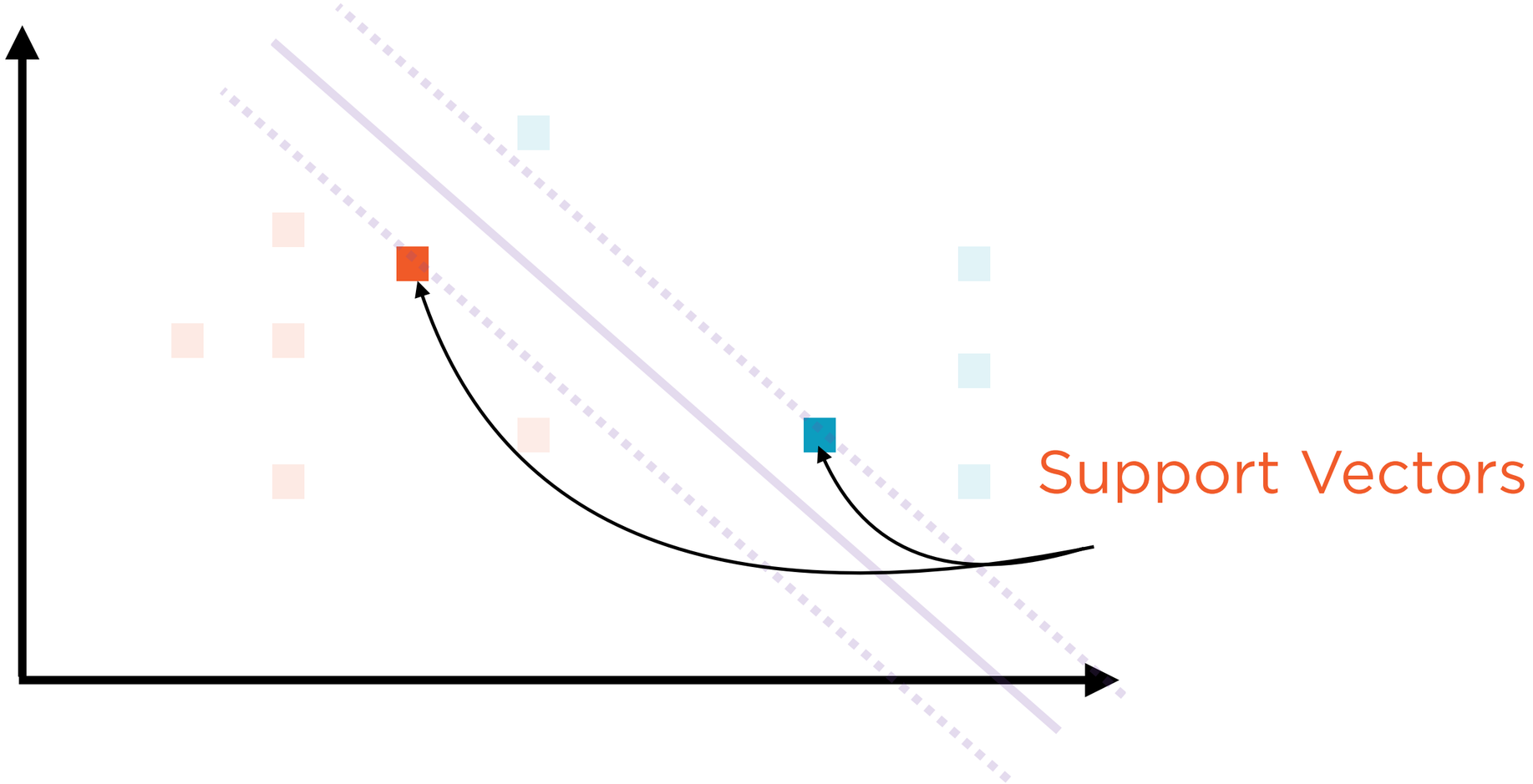
SVM classifiers find the hyperplane that best separates points in a hypercube

Hard Margin Classification



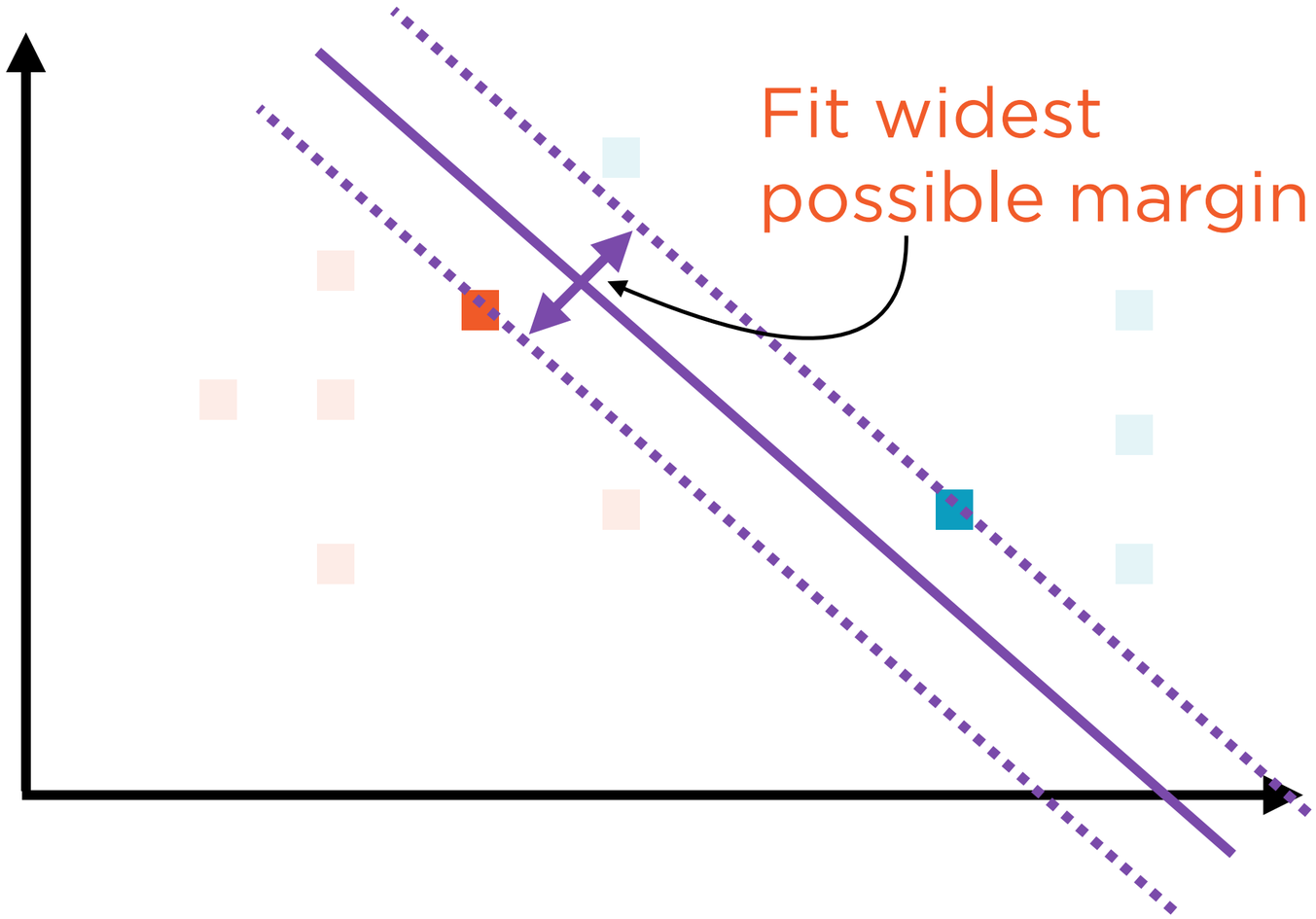
Ideally, data is linearly separable - hard decision boundary

Hard Margin Classification



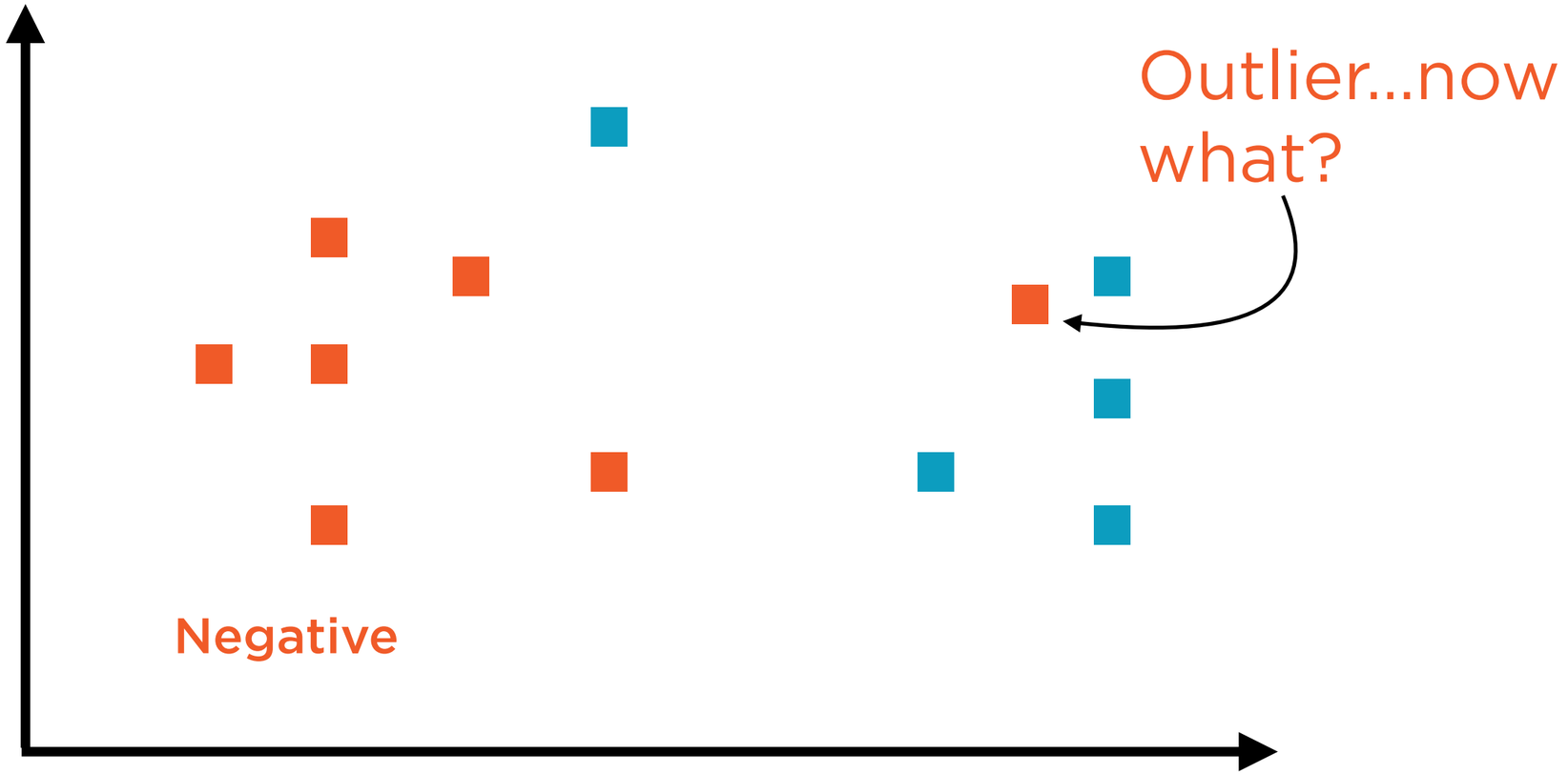
The nearest instances on either side of the boundary are called the support vectors

Hard Margin Classification



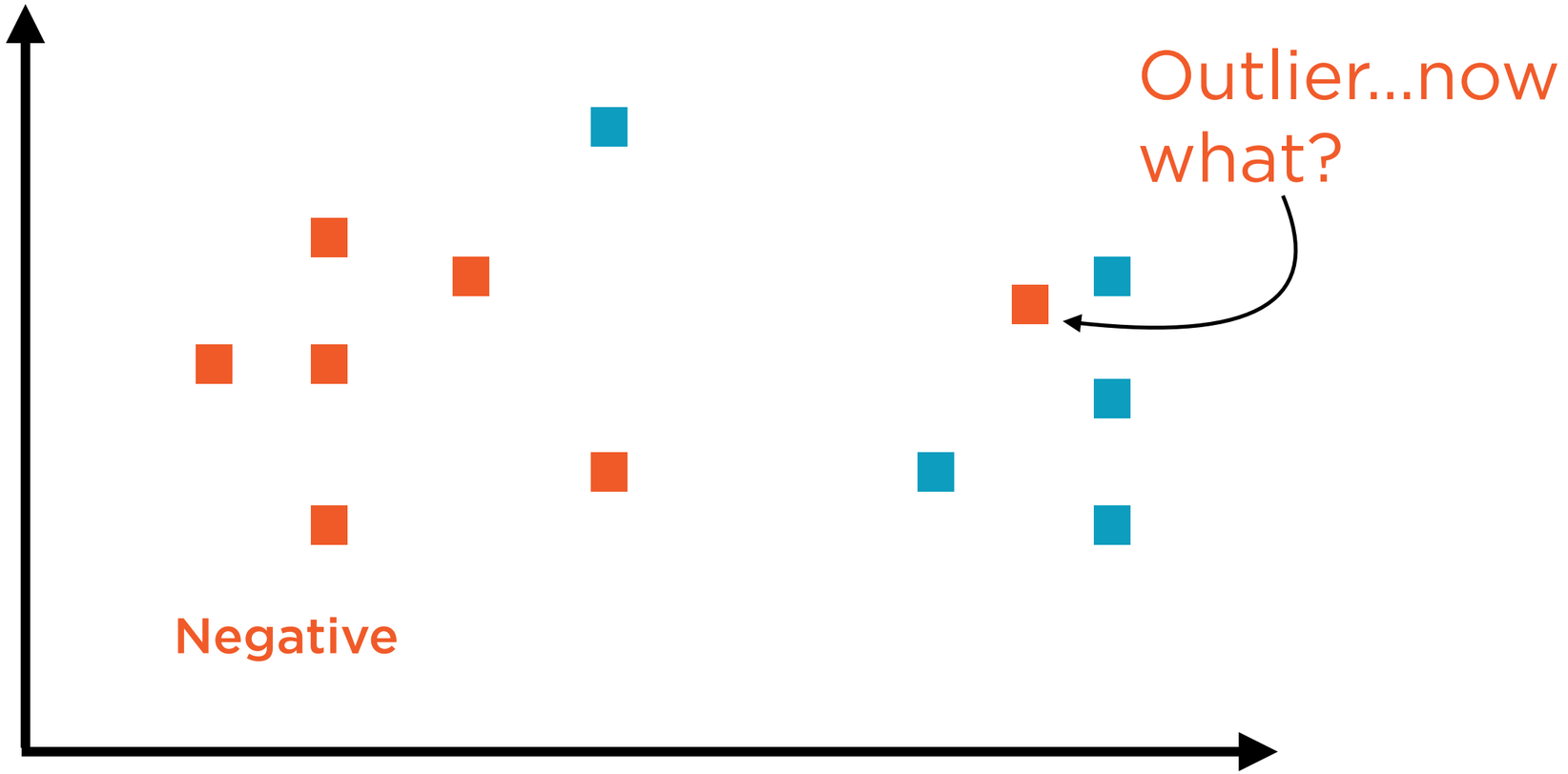
SVM finds the widest street between the nearest points on either side

Soft Margin Classification



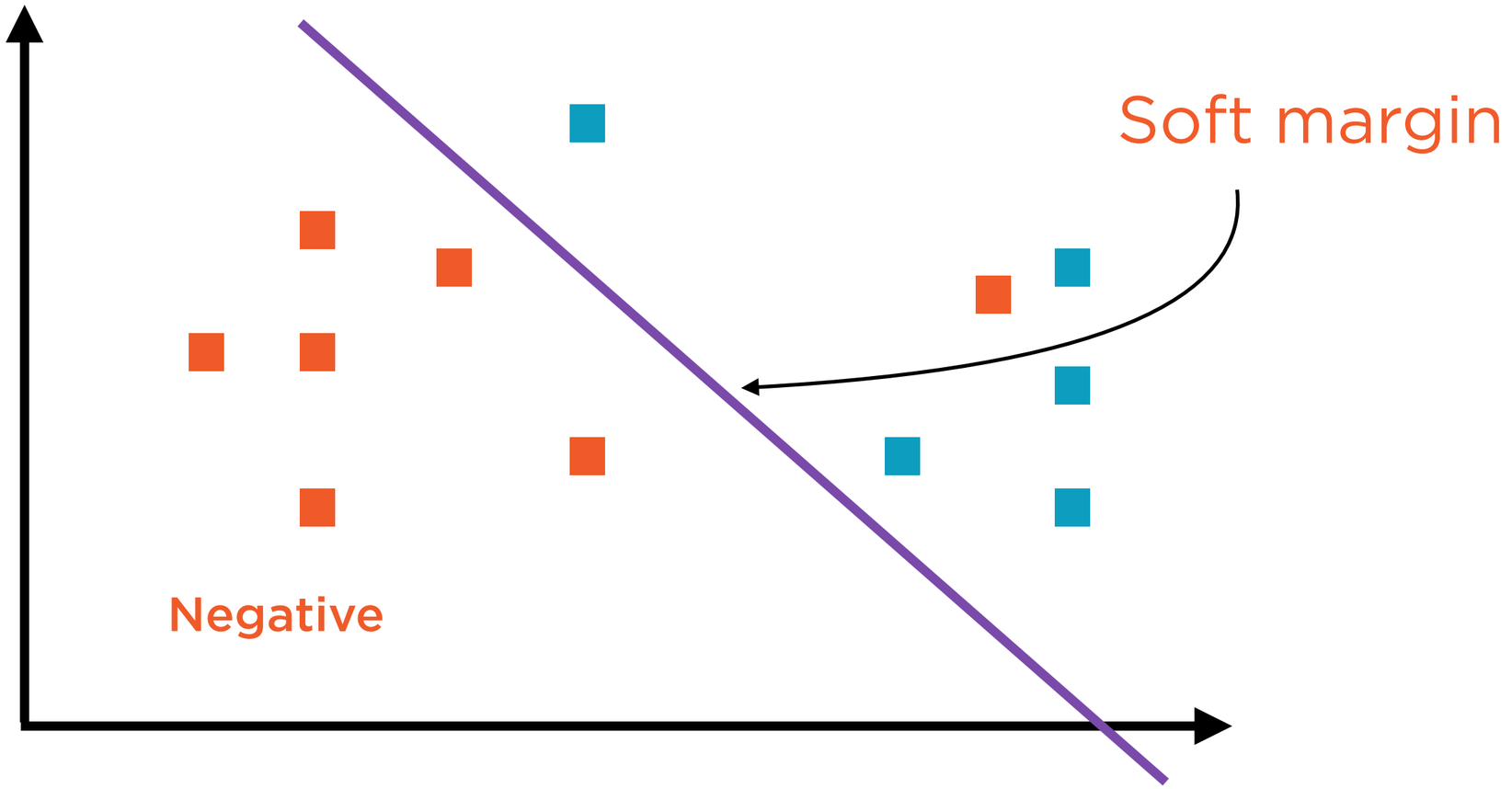
Hard margin classifiers are sensitive to outliers...

Soft Margin Classification



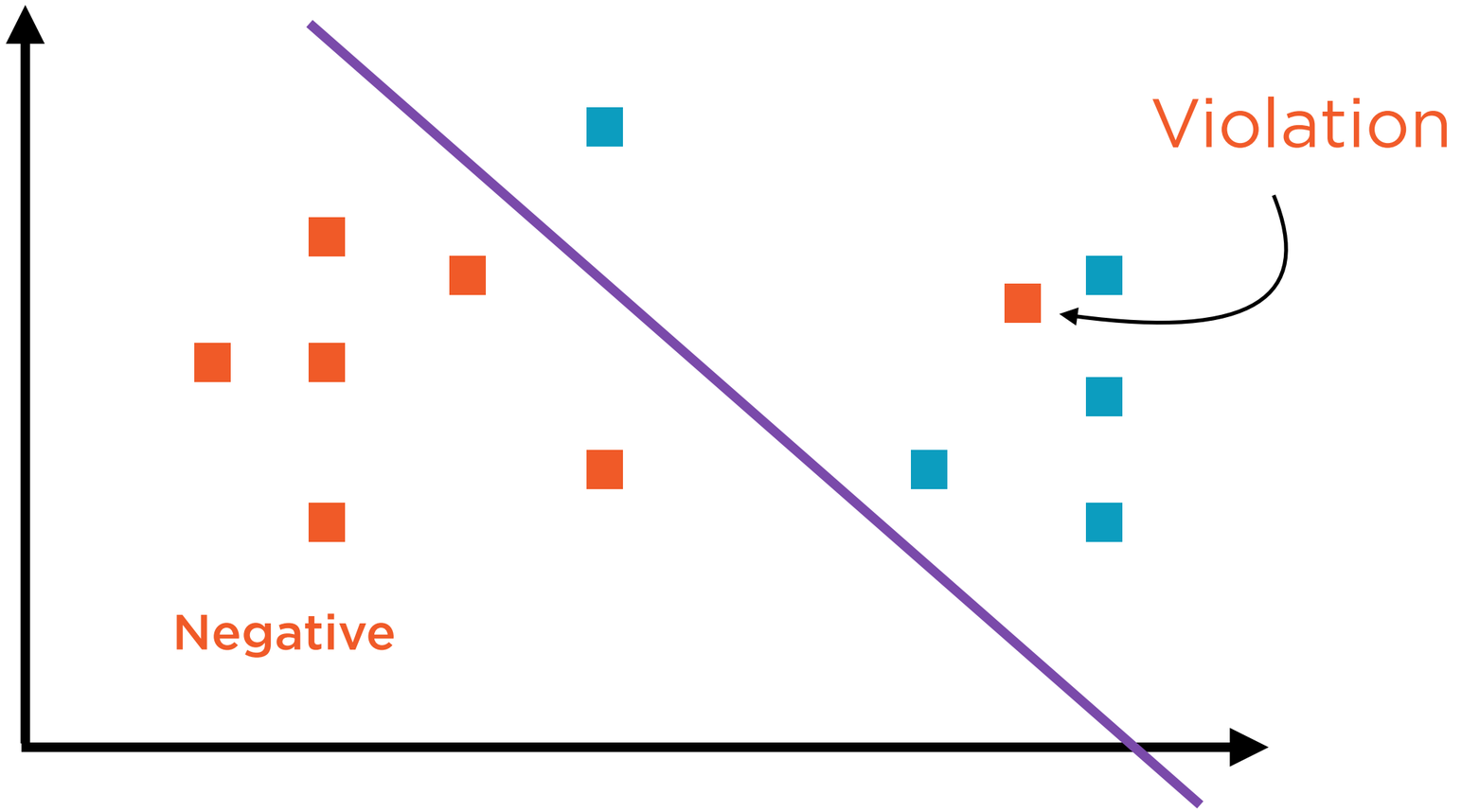
...and require perfectly linear separability in data

Soft Margin Classification



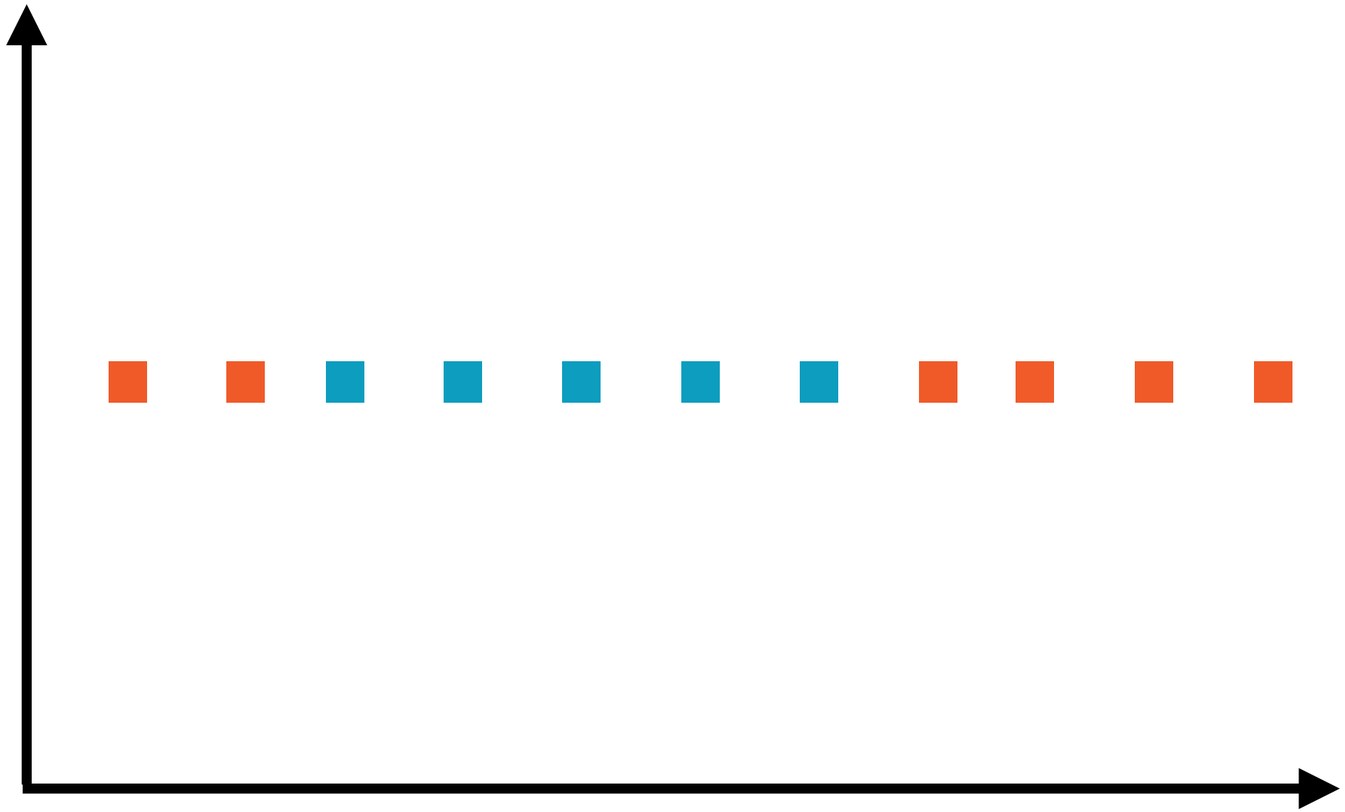
Soft margin classifiers allow some violations of the decision boundary

Soft Margin Classification



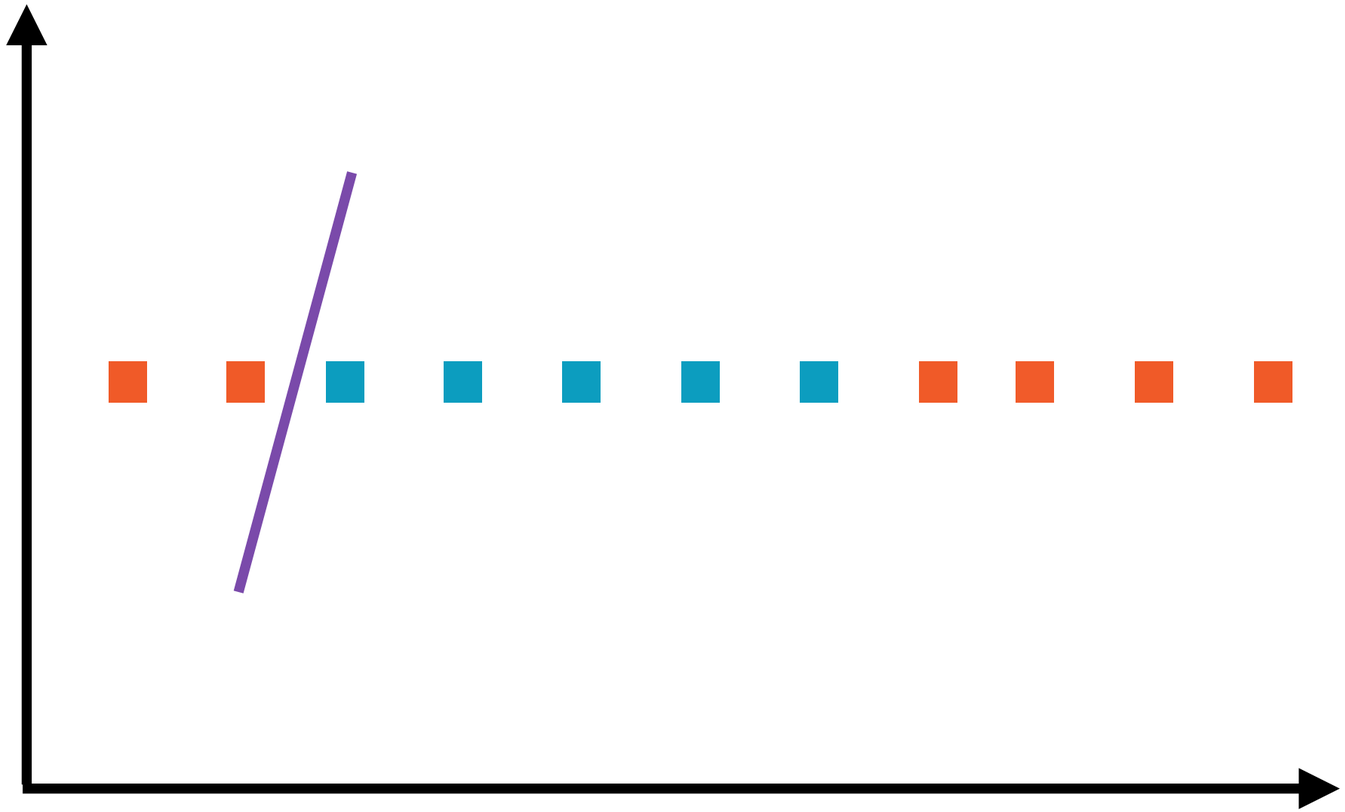
Soft margin classifiers allow some violations of the decision boundary

Non-separable Data



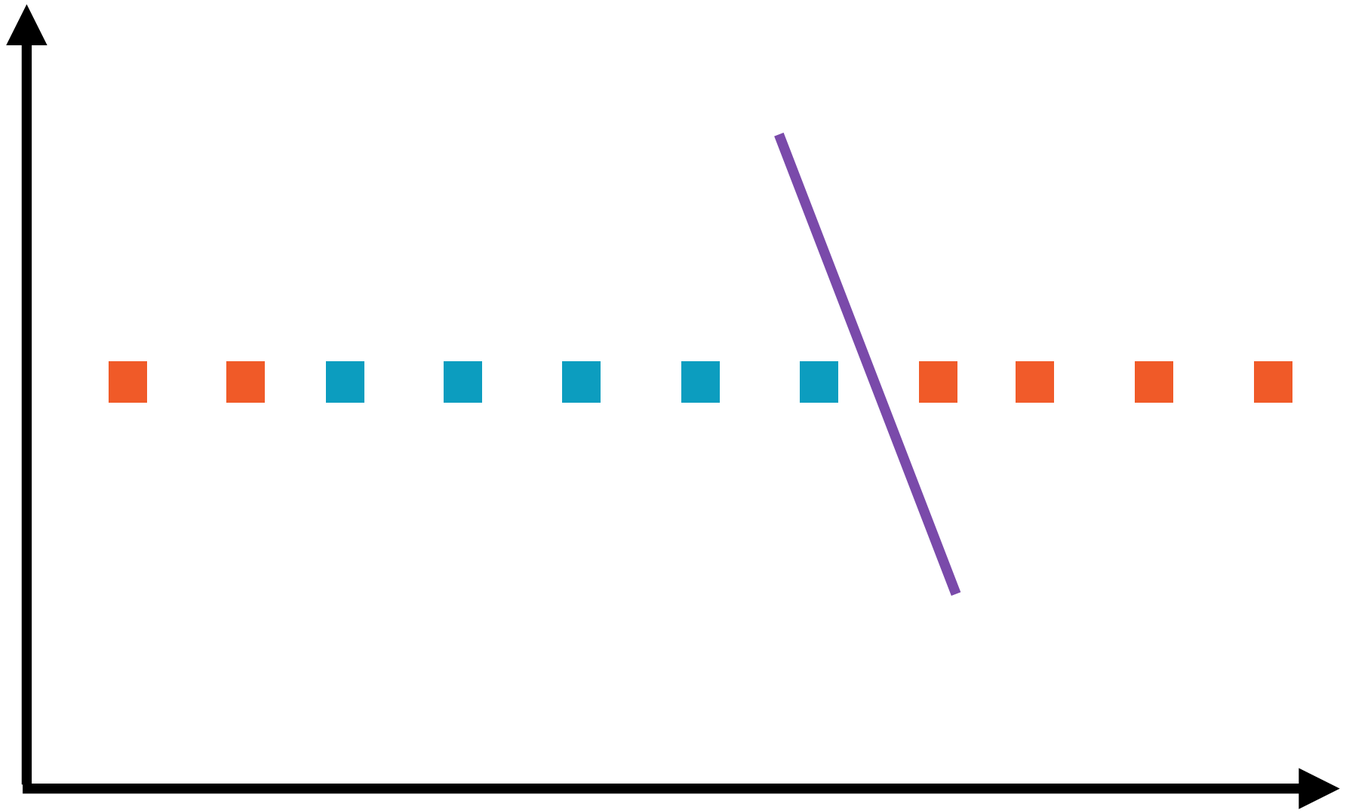
Smart transformations resolve surprisingly many such cases

Non-separable Data



Smart transformations resolve surprisingly many such cases

Non-separable Data

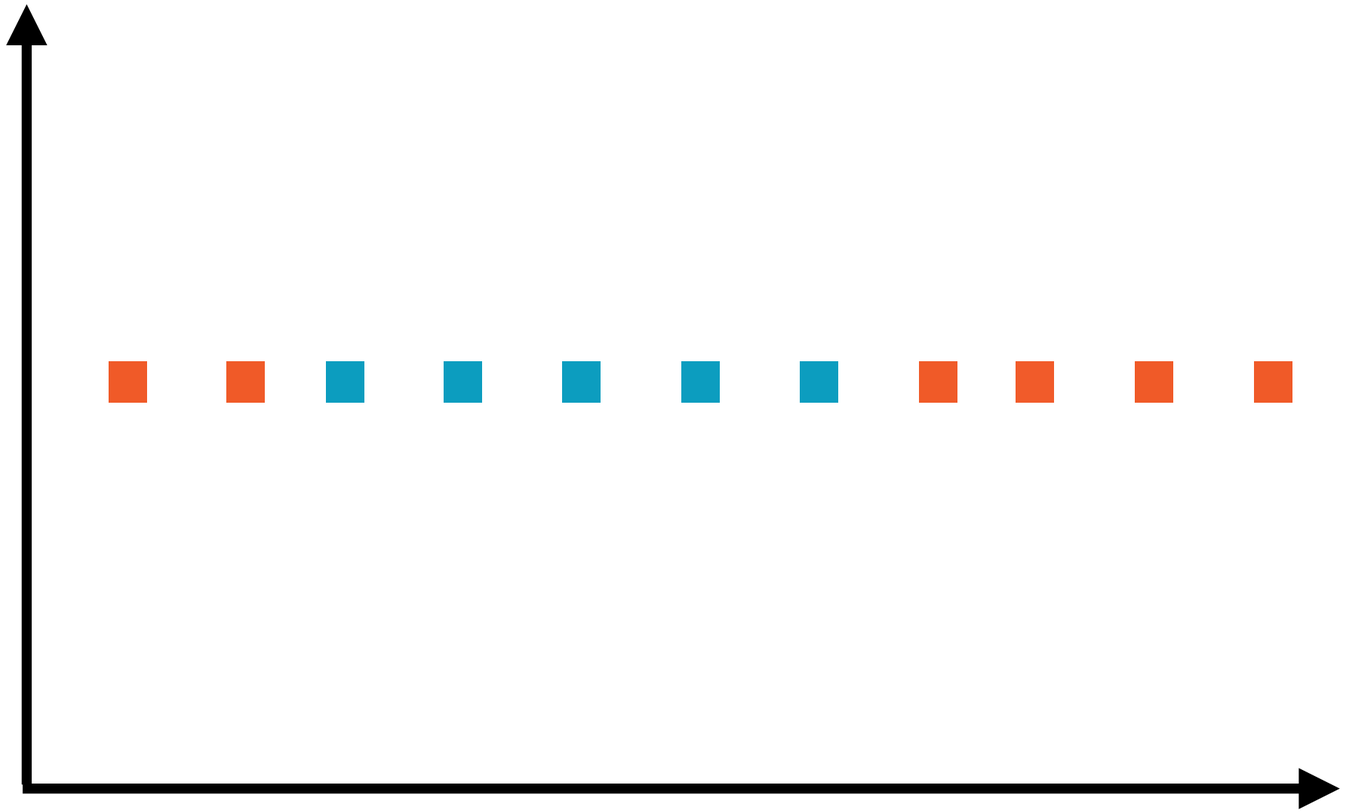


Smart transformations resolve surprisingly many such cases

SVM classification can be extended to almost any data using something called the **kernel trick**

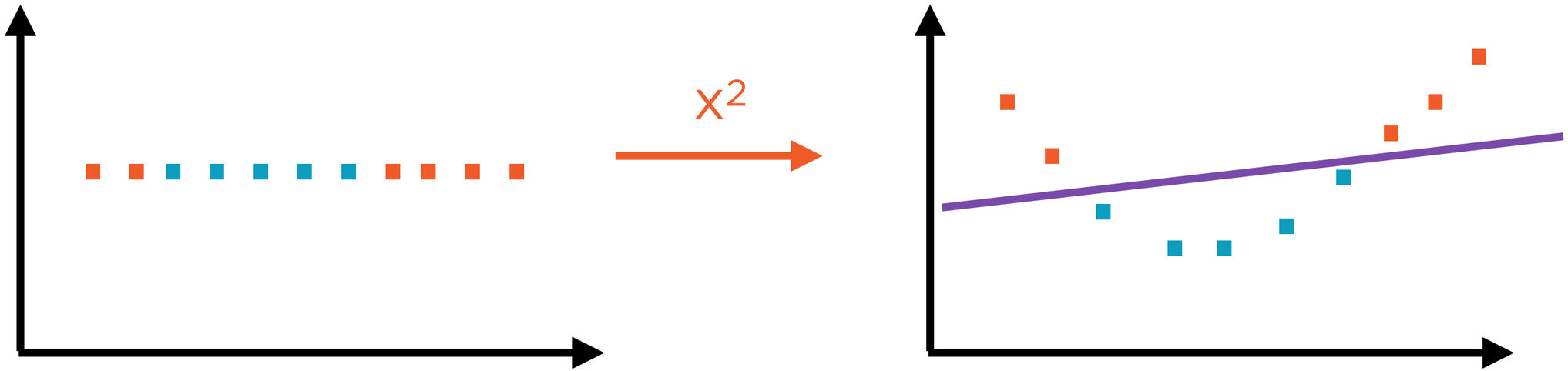
Kernel Trick and Kernel Approximations

Non-separable Data



Smart transformations resolve surprisingly many such cases

Nonlinear SVM



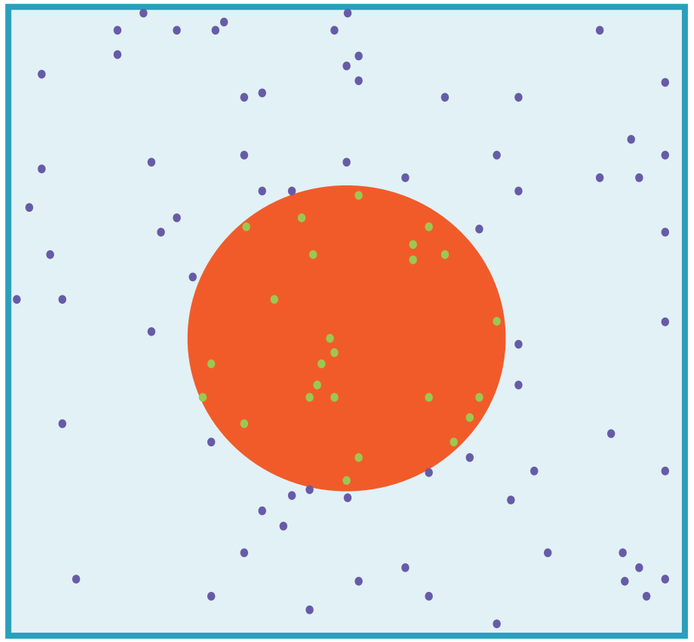
Original Data

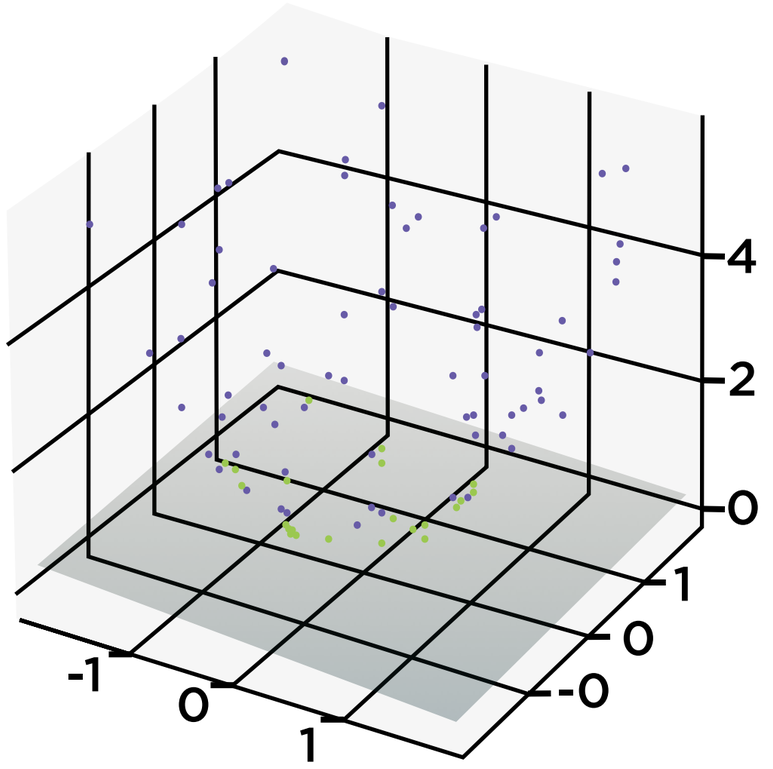
Not linearly separable

Square of original data

Now linearly separable!

Kernel Trick: From 2-D to 3-D





$$\phi((a,\ b)) = (a,\ b,\ a^2 + b^2)$$

Points are mapped to a 3-dimensional space where a separating hyperplane can be easily found

Kernel Trick

Apply a function (called the kernel function) to transform data so that it becomes much easier to model and process.

Kernel Trick

Very widely used in ML

Output of kernel function is called feature map

Mapping of the features from original dimension-space to kernel-space

This mapping is implicit

Generated feature maps are implicit feature maps

Implicit Feature Maps

Feature Maps require applying kernel function to each point

Common kernel functions usually

- Non-linear
- Quite complex

Finding feature maps is computationally intensive

Does not scale well to large datasets

Kernel computation
does not scale well;
kernel approximations
ride to the rescue

Kernel Approximations

Much simpler, approximations of kernel functions

Scale well to large datasets

Used to pre-process data before feeding to an ML model

Kernel Approximations

Pre-processing data allows us to fit simpler, more efficient, ML model

With some ML techniques, yield results comparable to non-linear (full) kernel

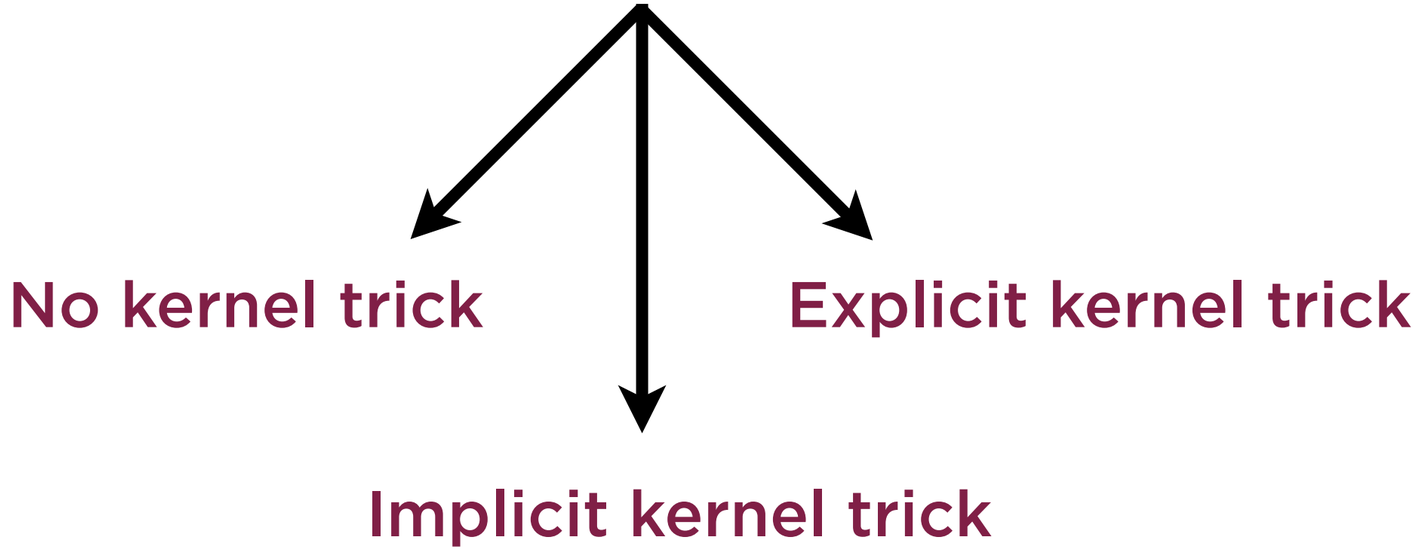
Explicit Feature Maps

Kernel approximations generate explicit feature maps

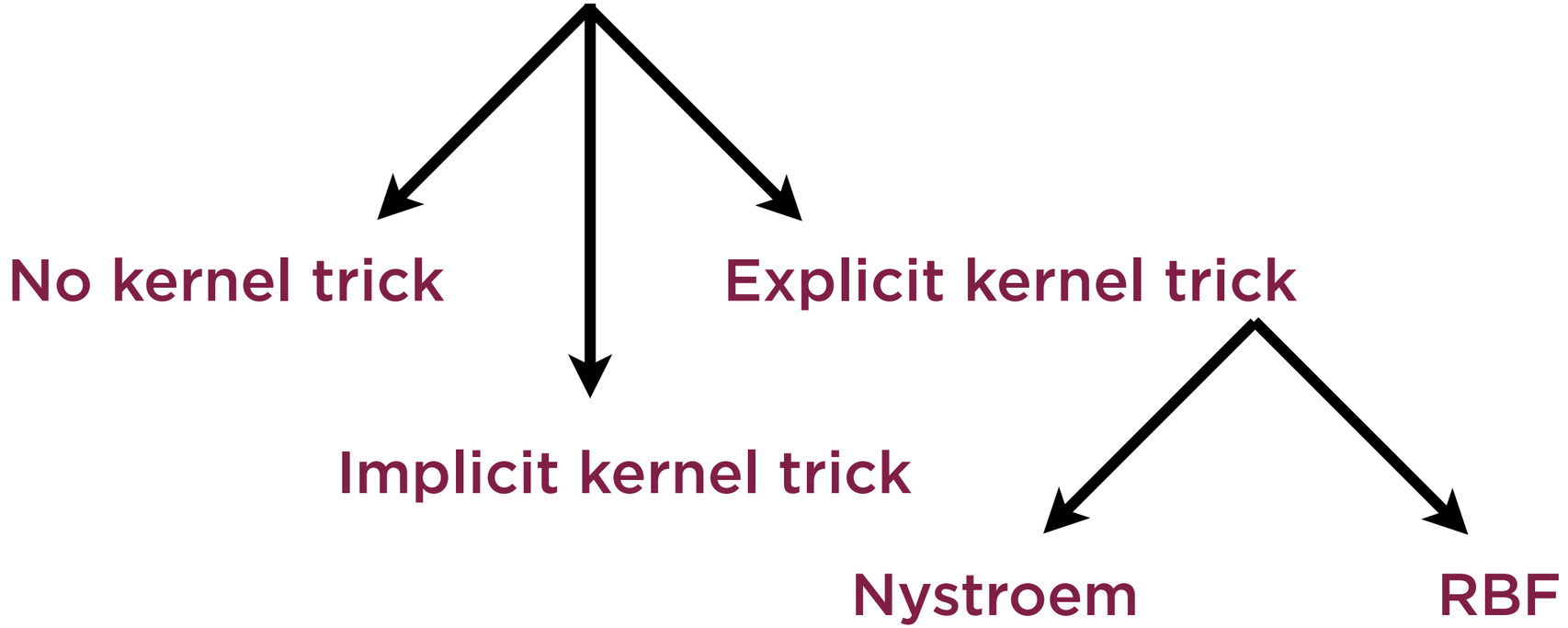
Nystroem kernels use a low-rank approximation of kernels

RBF or Radial Basis Function Kernel relies on a Monte Carlo approximation to kernel values

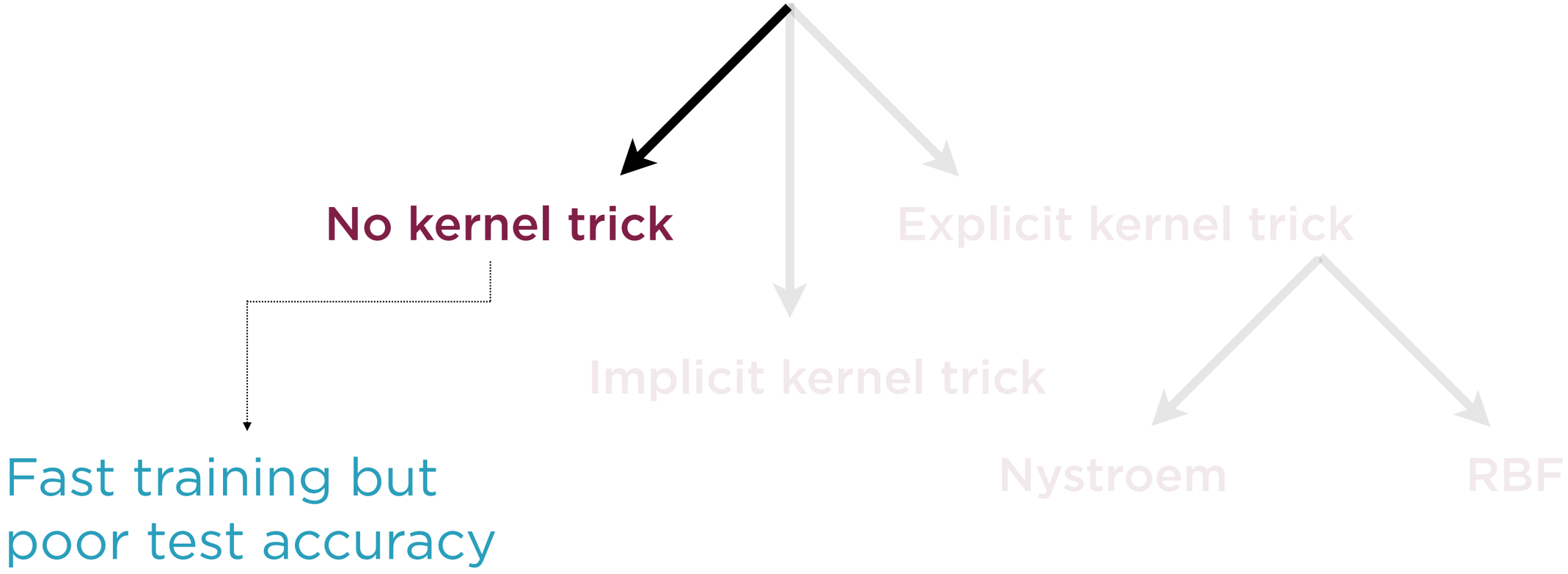
Using Kernels with Support Vector Classifiers



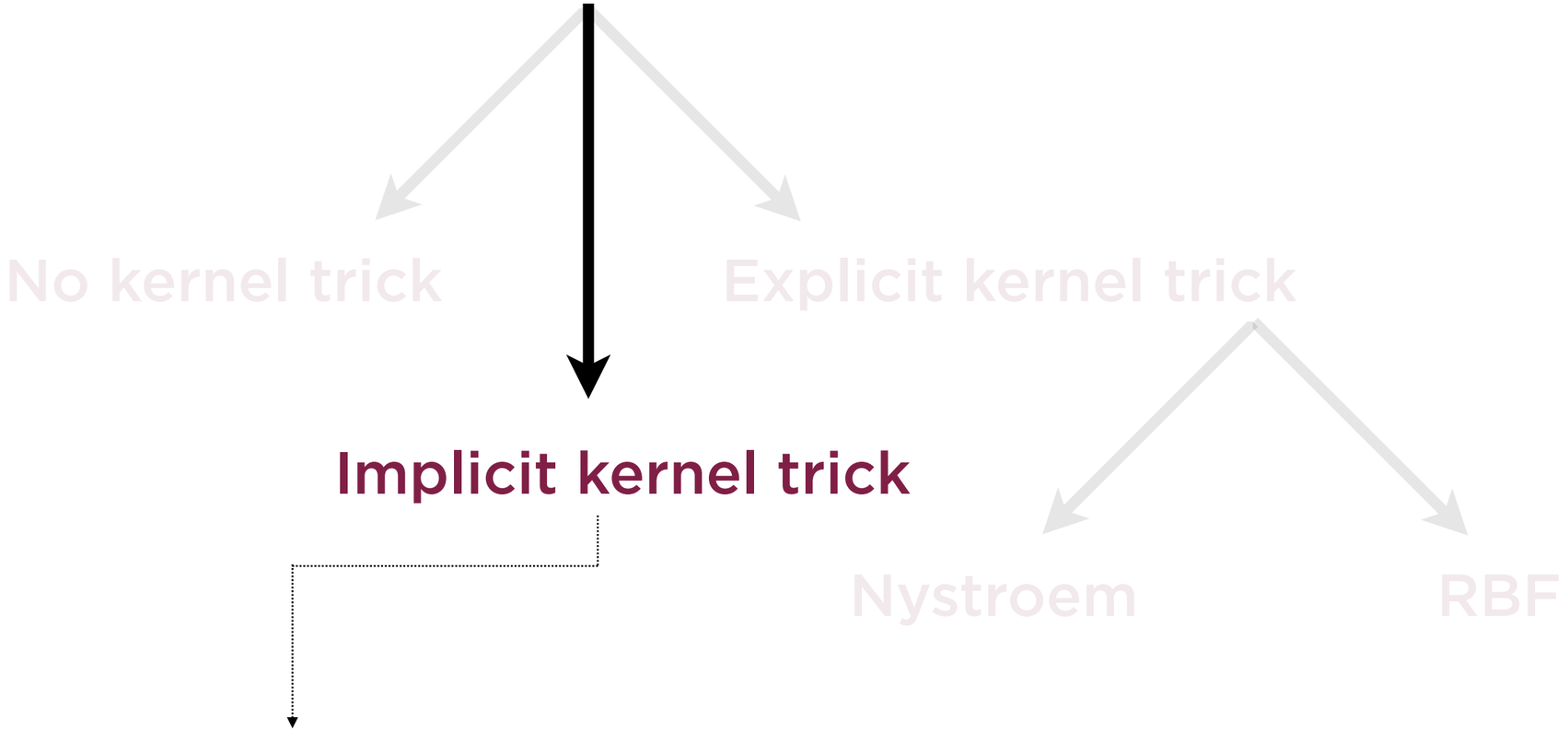
Using Kernels with Support Vector Classifiers



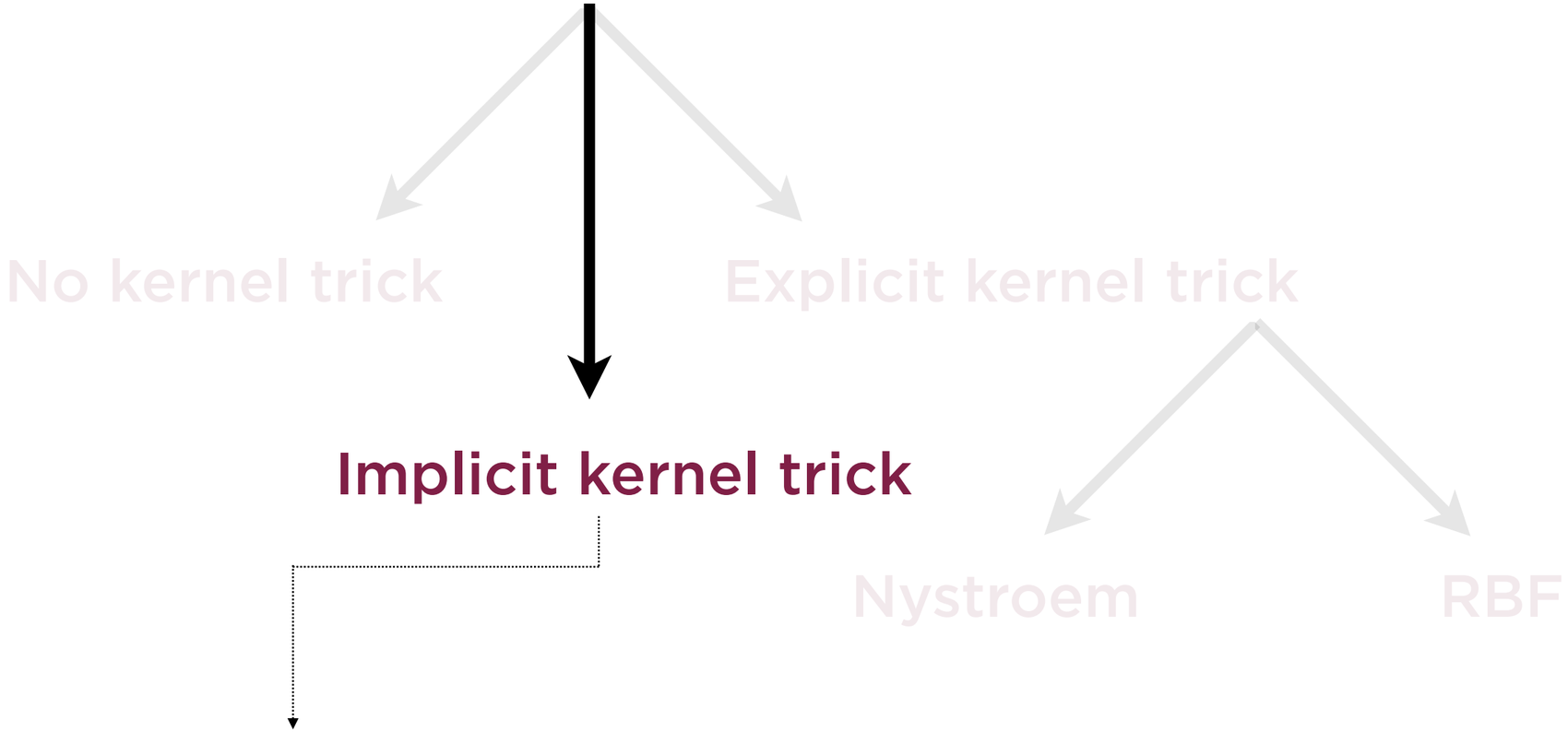
Using Kernels with Support Vector Classifiers



Using Kernels with Support Vector Classifiers

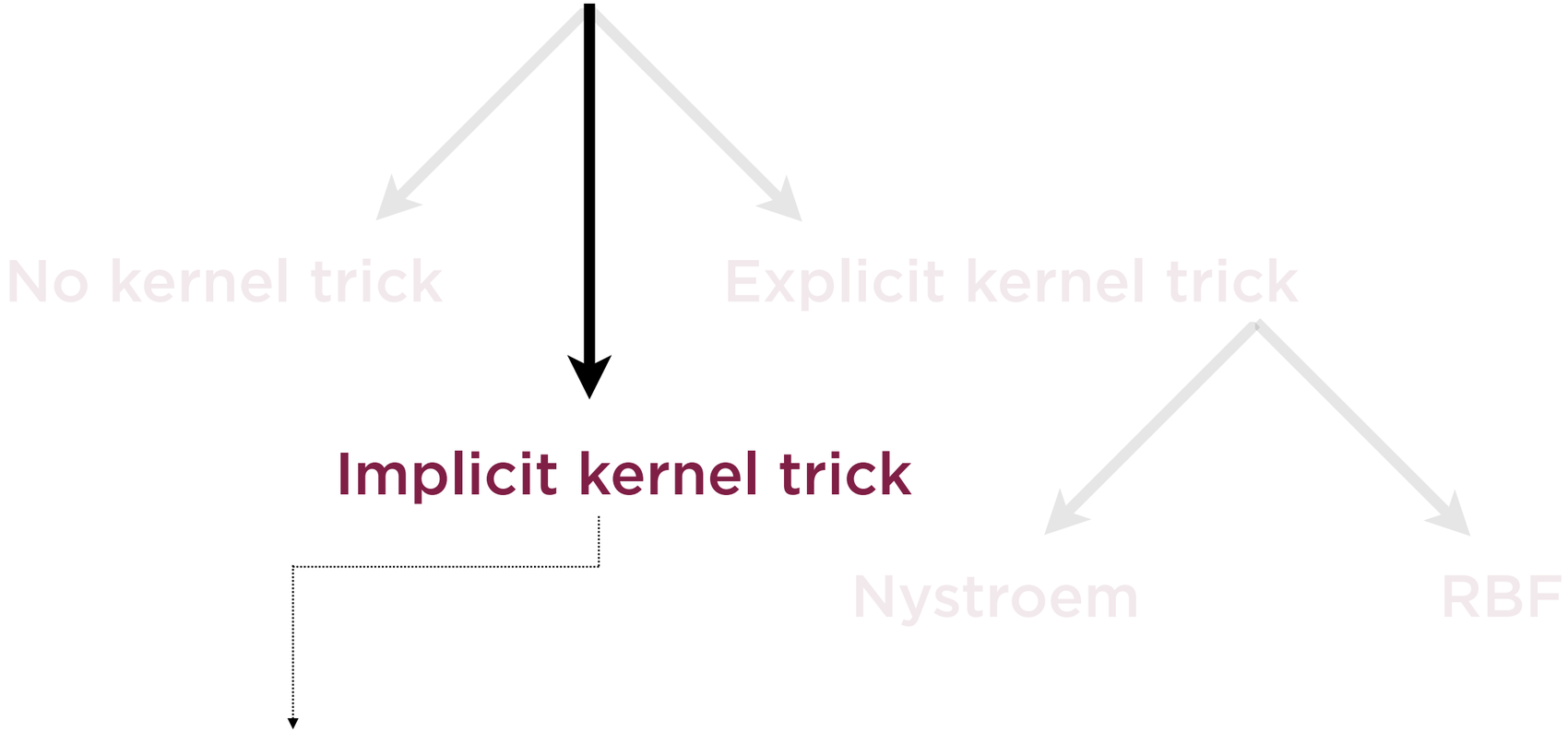


Using Kernels with Support Vector Classifiers

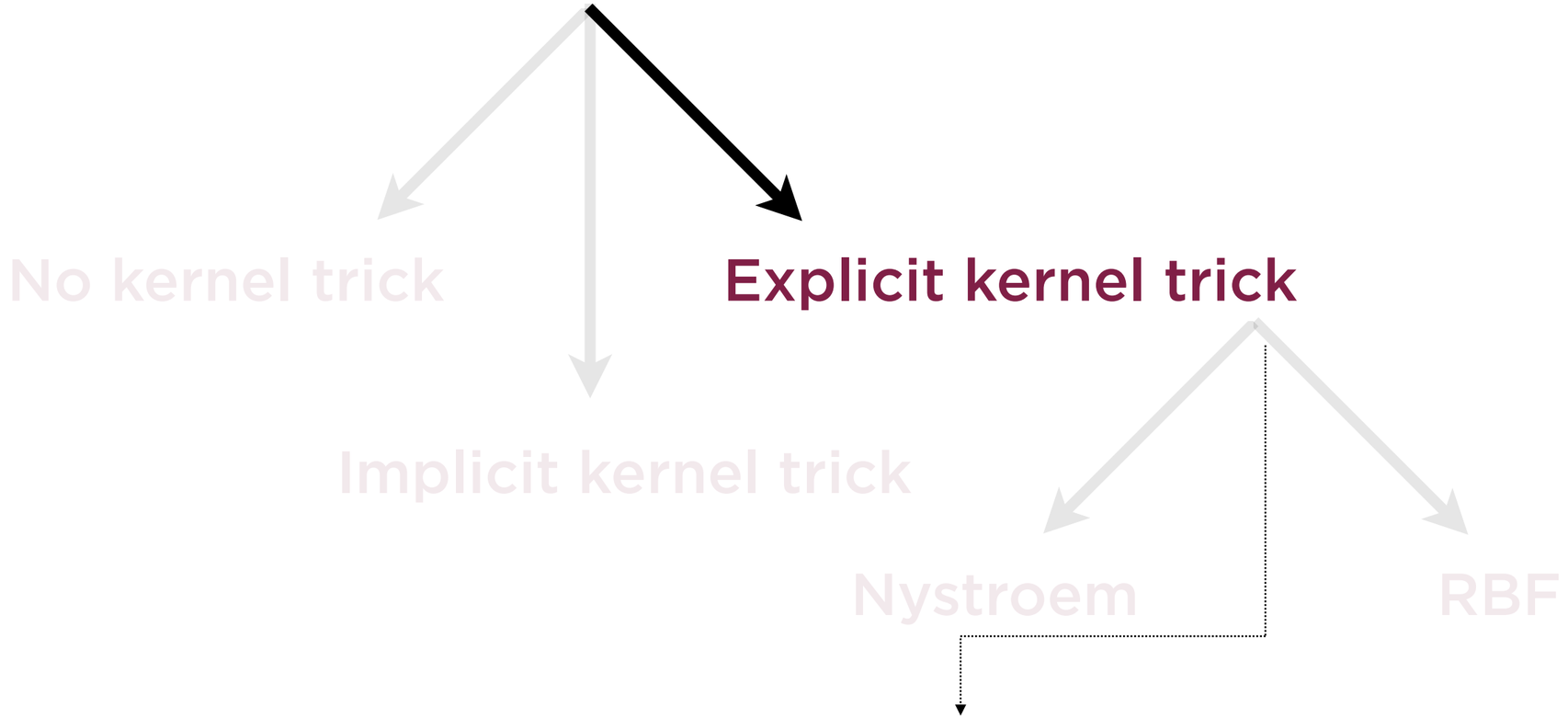


"Implicit" because classifier implicitly applies kernel inside estimator object

Using Kernels with Support Vector Classifiers

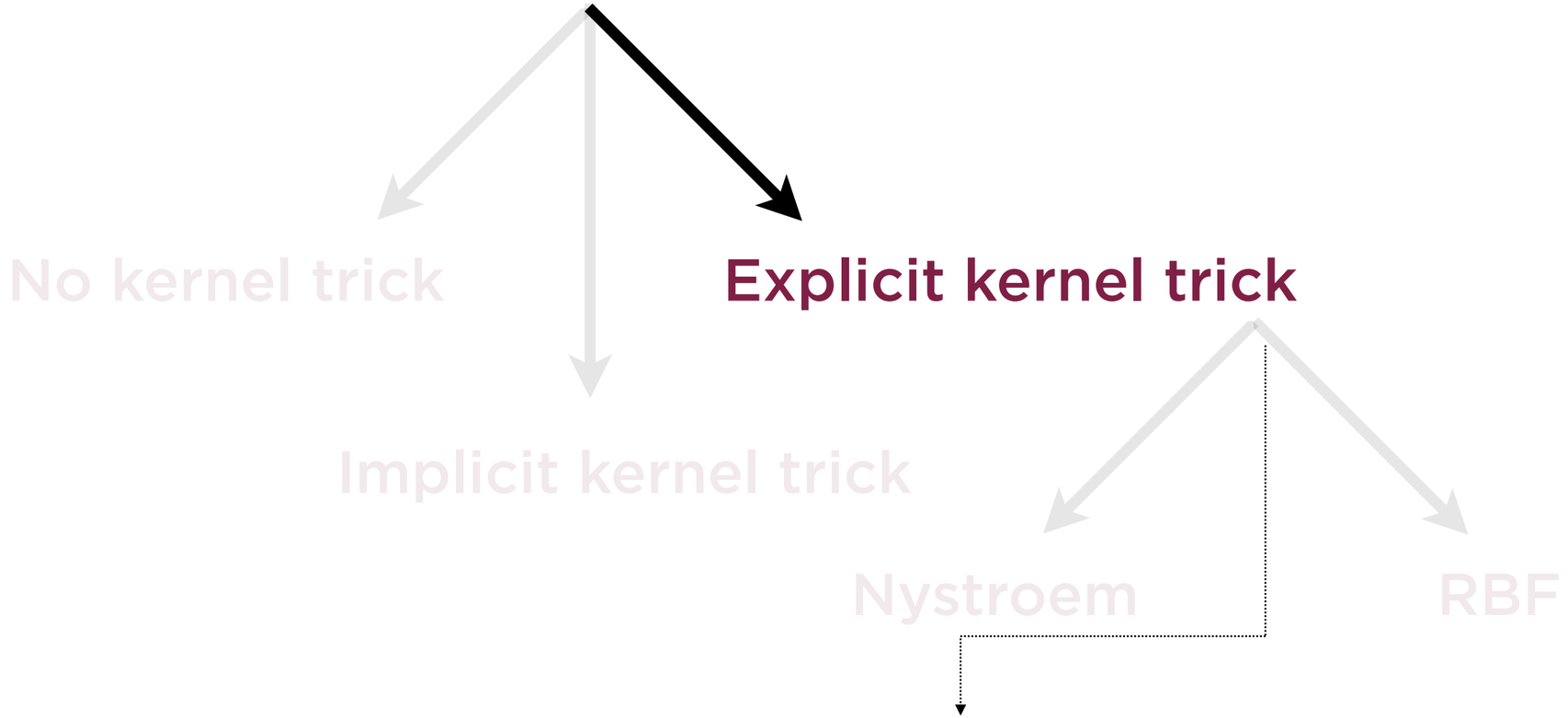


Using Kernels with Support Vector Classifiers



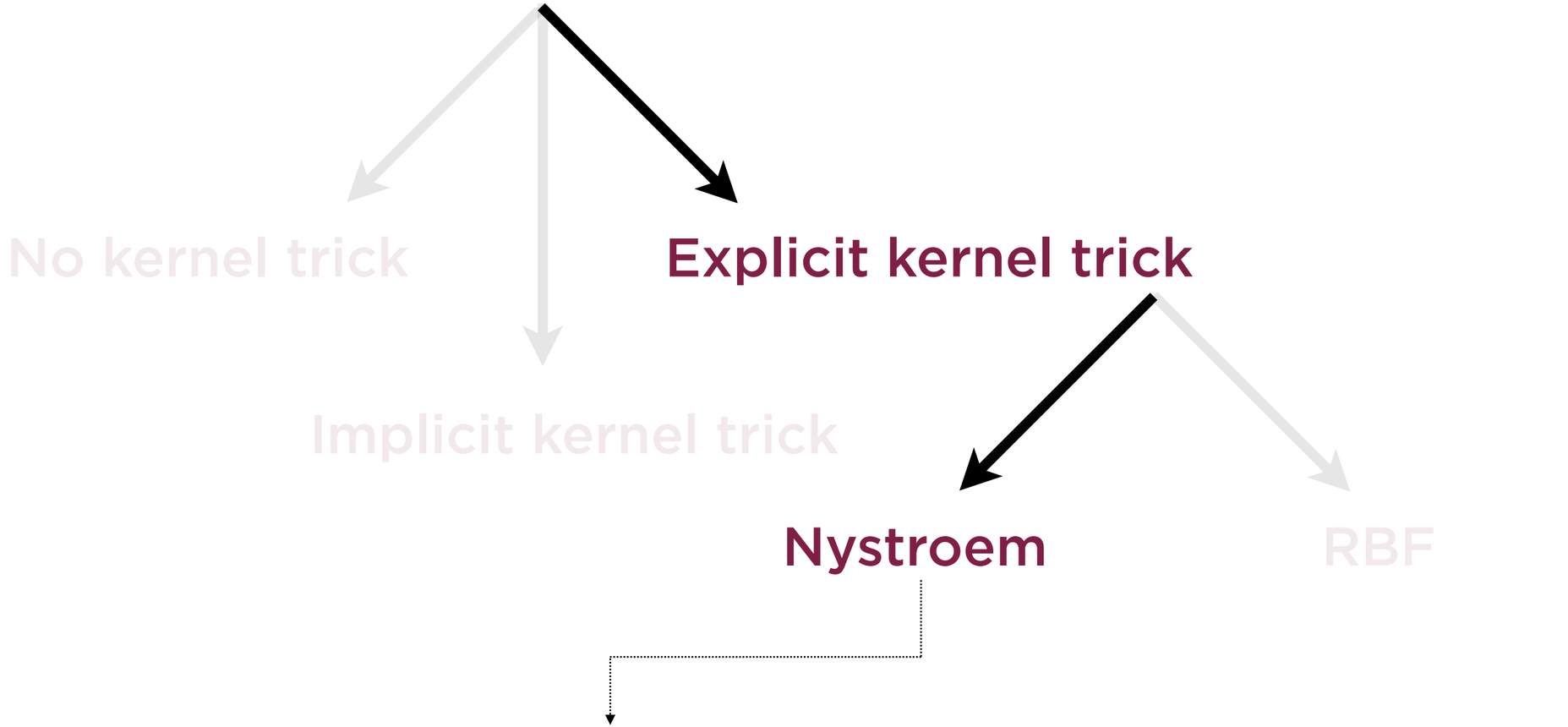
"Explicit" because kernel trick
applied during pre-processing

Using Kernels with Support Vector Classifiers

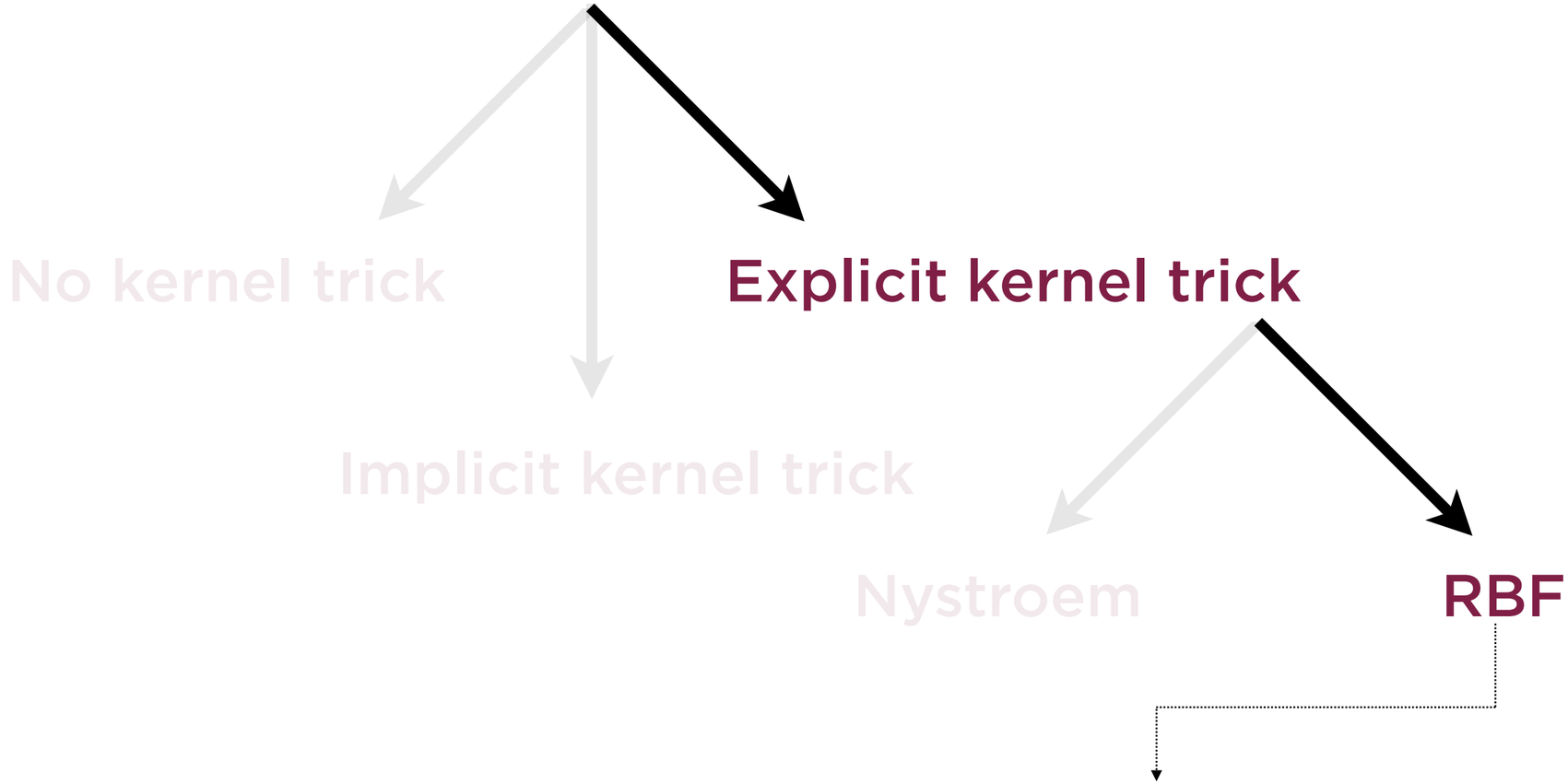


Use scikit-learn pipeline object to
pipe transformed data into estimator

Using Kernels with Support Vector Classifiers



Using Kernels with Support Vector Classifiers



Demo

Comparing classification models built using implicit and explicit feature maps

Summary

Understanding the need for kernel approximations

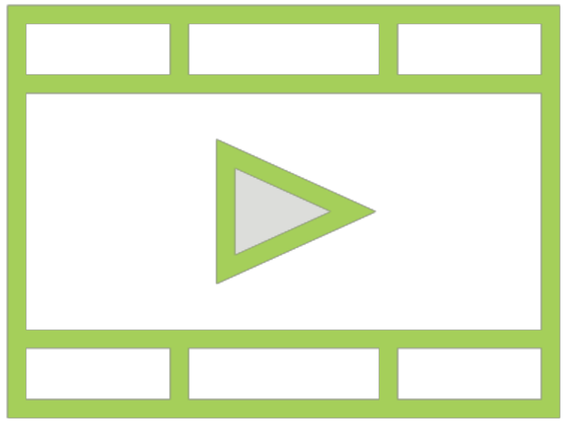
Feature mappings to approximate specific kernels

Nystroem method

Radial Basis Function (RBF) method

Contrasting SVMs with full kernel to simpler kernels with approximation

Related Courses

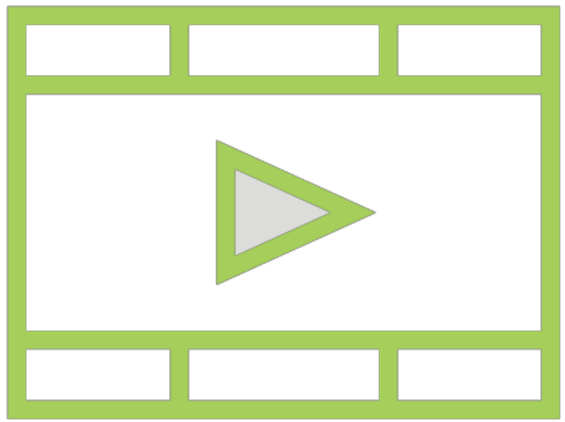


Building Features from Numeric Data

Building Features from Text Data

Building Features from Image Data

Related Courses



Building Clustering Models with scikit-learn

Building Regression Models with scikit-learn

Foundations of PyTorch