SVM

CS 530
Chapman

Spring 2021

1

Take 🏠 message for rest of course: SVM

- What is the SVM method and how does it work?
- What are kernels (kernel methods & kernel “trick”) and how to they work?
- In what sense is SVM similar to and different from other classification methods?

2

TABLE OF CONTENTS

- Support Vector Machines (SVMs)
  - Separating Hyperplanes
  - Maximal Margin Classifier
  - Support Vector Classifier
  - Feature Expansion
  - Support Vector Machines and Kernels
    - Linear Kernel
    - Polynomial Kernel
    - Radial Basis Function (RBF, Gaussian) Kernel
- Comparison with Other Classification Methods

3

SVM

We will now cover Support Vector Machines (SVM), a classification method that separates data linearly (possibly after projecting it to high-dimensional space) striving for a maximal margin between classes.

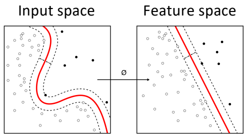
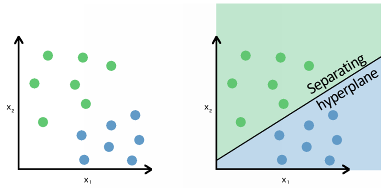


Image Source

4

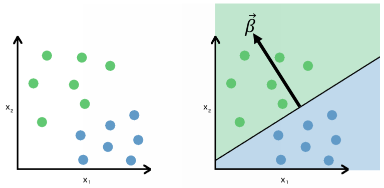
Separating Hyperplanes



If we have data where the categories can be linearly divided into two groups, then we can specify a separating hyperplane for the split.

5

Separating Hyperplanes



The separating hyperplane gives us a mathematical basis for defining regions and fitting a model. For any sample $(x_1, x_2, \cdots, x_n)$, or specifically $(x_1, x_2\ )$ in the example above:

$\beta_0\ +\ \beta_1 x_1 + \cdots\ + \beta_n x_n > 0$ ←green region (predict green)
$\beta_0\ +\ \beta_1 x_1 + \cdots\ + \beta_n x_n < 0$ ←blue region (predict blue)

6

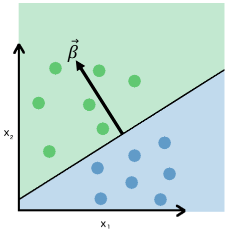
Separating Hyperplanes

We could also say that a separating hyperplane can classify data by setting

$y_i = 1 \leftarrow$ predict green
$y_i = -1 \leftarrow$ predict blue

so that the separating hyperplane always has the property

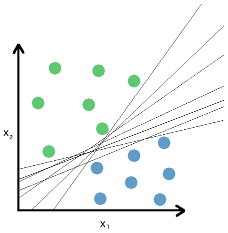
$$y_i(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_n x_{in}) > 0$$



7

Separating Hyperplanes

How do we know which hyperplane we should use to separate the data? It's possible there could be an infinite number of separating hyperplanes.

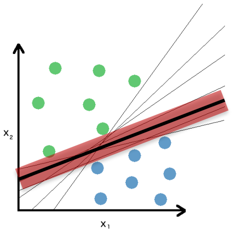


8

Separating Hyperplanes

How do we know which hyperplane we should use to separate the data? It's possible there could be an infinite number of separating hyperplanes.

We use the maximal-margin hyperplane.



9

Maximal Margin Classifier

The *maximal margin classifier* is the hyperplane with the widest gap (margin) between itself and surrounding training data.

It is found by solving an optimization problem:

Choose $\beta_0, \ldots, \beta_n$ that maximize M ("margin")

Subject to

$$\sum_{j=1}^{n} \beta_j^2 = 1$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_n x_{in}) \geq M \ \ \forall i$$

where $y_i \in \{-1,1\}$ is given by the class of the training data point (blue or purple), and M is proportional to the width of the margin.

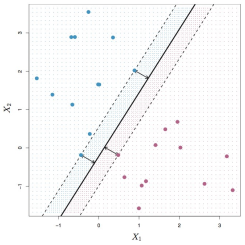


Image Source

10

Maximal Margin Classifier

In vector formulation, the optimization problem becomes:

$$\underset{\beta_0, \vec{\beta}}{maximize} \, M$$

Subject to

$$\|\vec{\beta}\| = 1$$

$$y_i(\beta_0 + \vec{\beta} \cdot \vec{x}_i) \geq M \, \forall i$$

where $y_i \in \{-1,1\}$ is given by the class of the training data point (blue or purple), and M is proportional to the width of the margin.

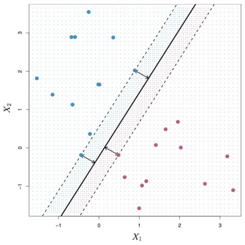


Image Source

11

Maximal Margin Classifier

1. $\underset{\beta_0, \vec{\beta}}{maximize} \, M$, because we want the maximal margin classifier.
2. The distance from the i'th sample to the hyperplane is given by $|(\beta_0 + \vec{\beta} \cdot \vec{x}_i)|$. So, $y_i(\beta_0 + \vec{\beta} \cdot \vec{x}_i) \geq M \, \forall i$ guarantees that each sample will be classified properly (remember $y_i \in \{-1,1\}$) and will be at least M away from the hyperplane.
3. $\|\vec{\beta}\| = 1$ guarantees that $\|\vec{\beta}\|$ will not just be as large as we wish to maximize M, because $\forall k \neq 0, k\vec{\beta}$, represents the same hyperplane.

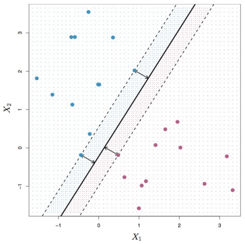
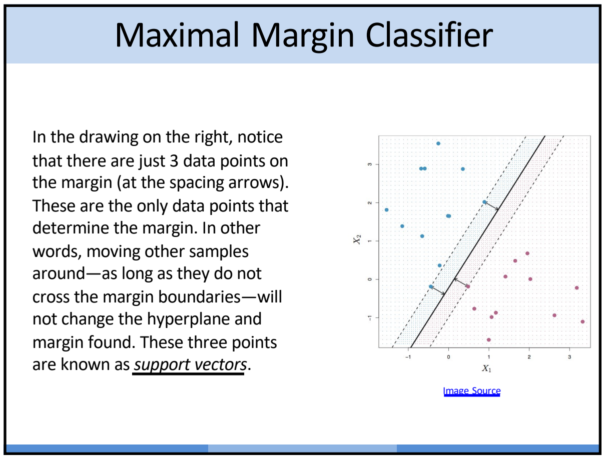


Image Source

12

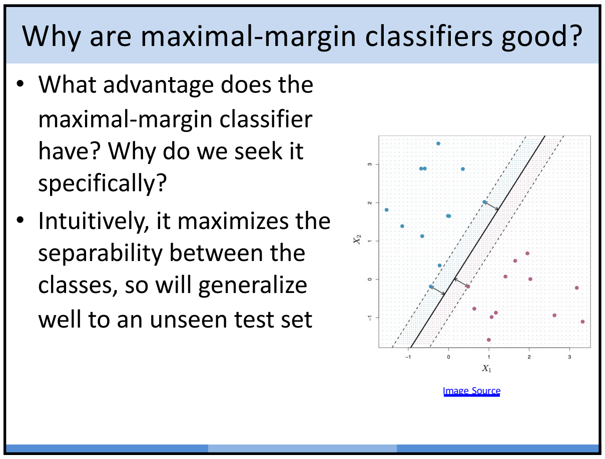


Maximal Margin Classifier

In the drawing on the right, notice that there are just 3 data points on the margin (at the spacing arrows). These are the only data points that determine the margin. In other words, moving other samples around—as long as they do not cross the margin boundaries—will not change the hyperplane and margin found. These three points are known as *support vectors*.

Image Source

13

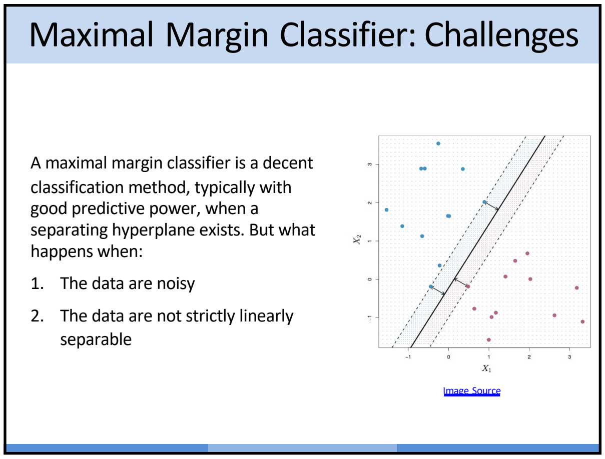


Why are maximal-margin classifiers good?

- What advantage does the maximal-margin classifier have? Why do we seek it specifically?
- Intuitively, it maximizes the separability between the classes, so will generalize well to an unseen test set

Image Source

14



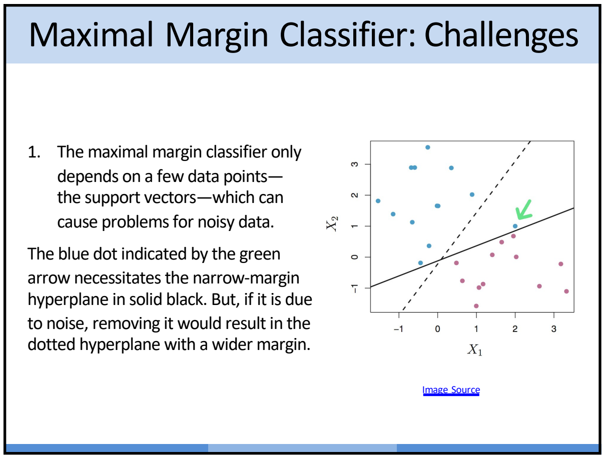
Maximal Margin Classifier: Challenges

A maximal margin classifier is a decent classification method, typically with good predictive power, when a separating hyperplane exists. But what happens when:

1. The data are noisy
2. The data are not strictly linearly separable

Image Source

15



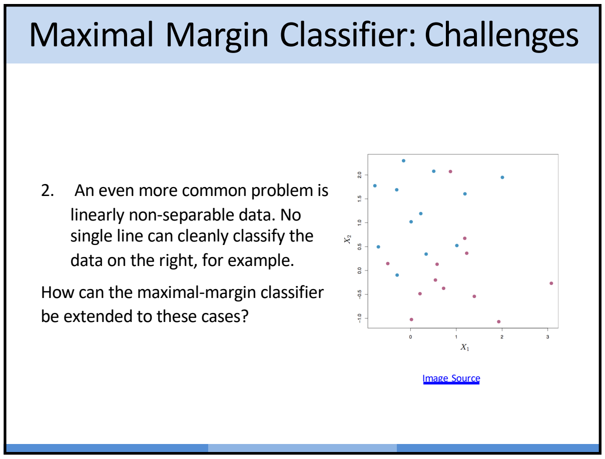
Maximal Margin Classifier: Challenges

1. The maximal margin classifier only depends on a few data points—the support vectors—which can cause problems for noisy data.

The blue dot indicated by the green arrow necessitates the narrow-margin hyperplane in solid black. But, if it is due to noise, removing it would result in the dotted hyperplane with a wider margin.

Image Source

16



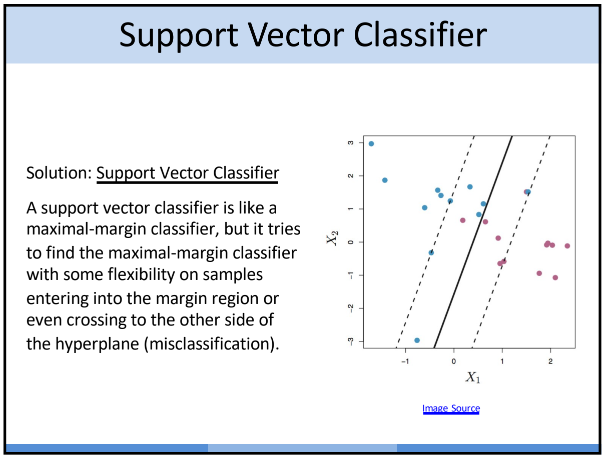
Maximal Margin Classifier: Challenges

2. An even more common problem is linearly non-separable data. No single line can cleanly classify the data on the right, for example.

How can the maximal-margin classifier be extended to these cases?

Image Source

17



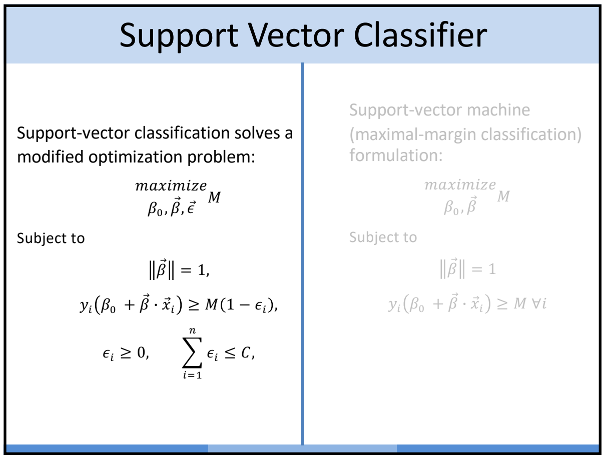
Support Vector Classifier

Solution: Support Vector Classifier

A support vector classifier is like a maximal-margin classifier, but it tries to find the maximal-margin classifier with some flexibility on samples entering into the margin region or even crossing to the other side of the hyperplane (misclassification).

Image Source

18



Support Vector Classifier

Support-vector classification solves a modified optimization problem:

$$\underset{\beta_0, \vec{\beta}, \vec{\epsilon}}{maximize} \; M$$

Subject to

$$\|\vec{\beta}\| = 1,$$

$$y_i(\beta_0 + \vec{\beta} \cdot \vec{x}_i) \geq M(1 - \epsilon_i),$$

$$\epsilon_i \geq 0, \qquad \sum_{i=1}^{n} \epsilon_i \leq C,$$

Support-vector machine (maximal-margin classification) formulation:

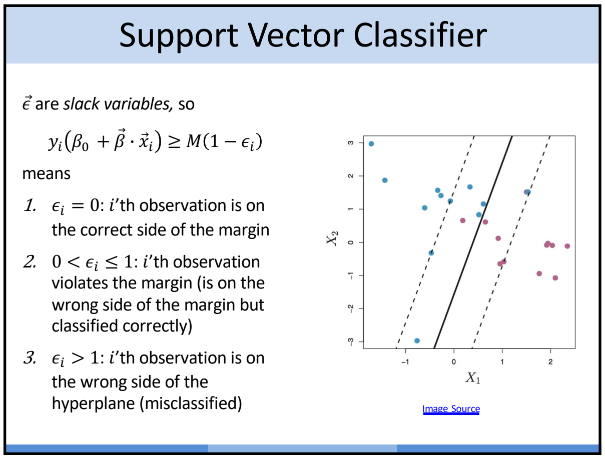
$$\underset{\beta_0, \vec{\beta}}{maximize} \; M$$

Subject to

$$\|\vec{\beta}\| = 1$$

$$y_i(\beta_0 + \vec{\beta} \cdot \vec{x}_i) \geq M \; \forall i$$

19



Support Vector Classifier

$\vec{\epsilon}$ are *slack variables*, so

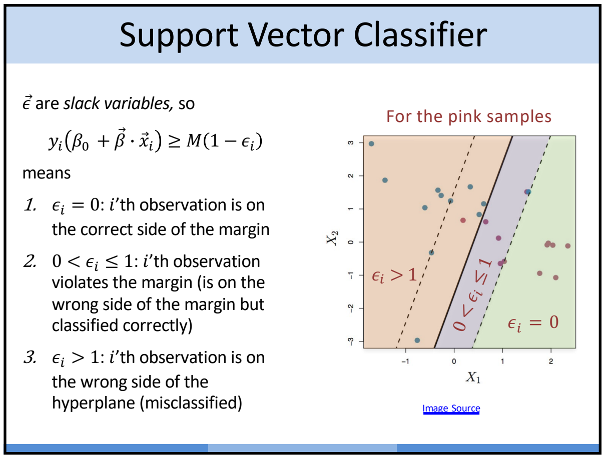
$$y_i(\beta_0 + \vec{\beta} \cdot \vec{x}_i) \geq M(1 - \epsilon_i)$$

means

1. $\epsilon_i = 0$: i'th observation is on the correct side of the margin
2. $0 < \epsilon_i \leq 1$: i'th observation violates the margin (is on the wrong side of the margin but classified correctly)
3. $\epsilon_i > 1$: i'th observation is on the wrong side of the hyperplane (misclassified)

Image Source

20



Support Vector Classifier

$\vec{\epsilon}$ are *slack variables*, so

$$y_i(\beta_0 + \vec{\beta} \cdot \vec{x}_i) \geq M(1 - \epsilon_i)$$

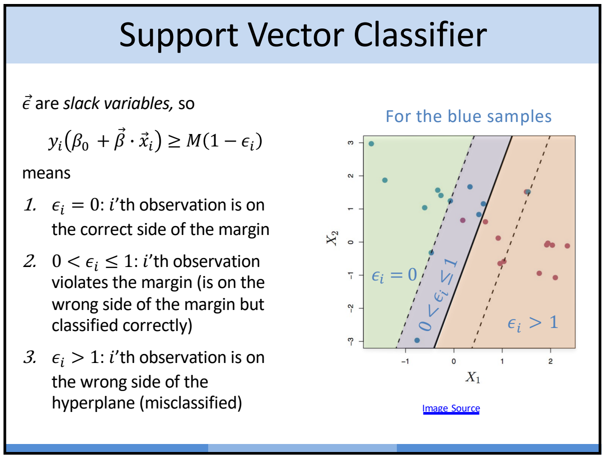
means

1. $\epsilon_i = 0$: i'th observation is on the correct side of the margin
2. $0 < \epsilon_i \leq 1$: i'th observation violates the margin (is on the wrong side of the margin but classified correctly)
3. $\epsilon_i > 1$: i'th observation is on the wrong side of the hyperplane (misclassified)

For the pink samples

Image Source

21



Support Vector Classifier

$\vec{\epsilon}$ are *slack variables*, so

$$y_i(\beta_0 + \vec{\beta} \cdot \vec{x}_i) \geq M(1 - \epsilon_i)$$

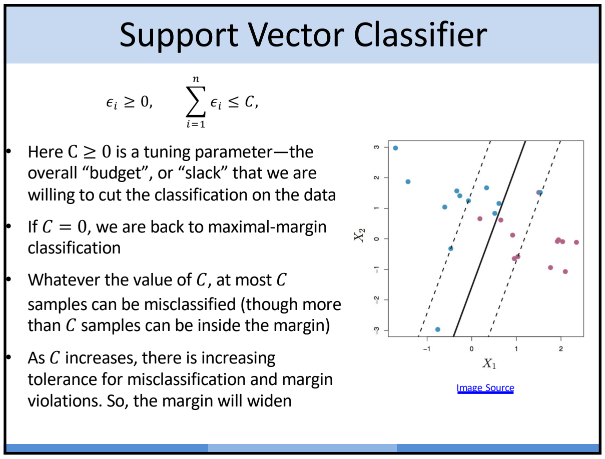
means

1. $\epsilon_i = 0$: i'th observation is on the correct side of the margin
2. $0 < \epsilon_i \leq 1$: i'th observation violates the margin (is on the wrong side of the margin but classified correctly)
3. $\epsilon_i > 1$: i'th observation is on the wrong side of the hyperplane (misclassified)

For the blue samples

Image Source

22



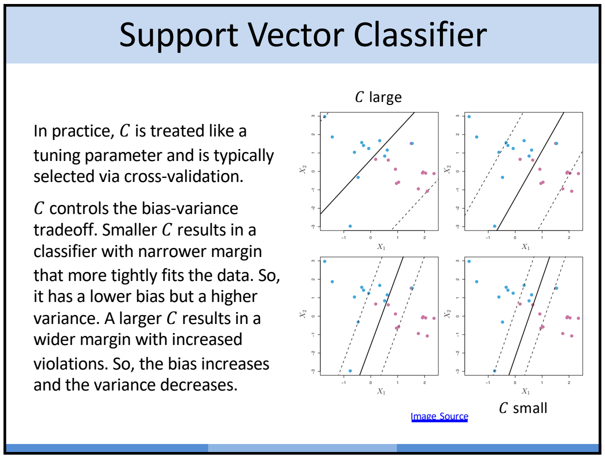
Support Vector Classifier

$$\epsilon_i \geq 0, \qquad \sum_{i=1}^{n} \epsilon_i \leq C,$$

- Here C ≥ 0 is a tuning parameter—the overall “budget”, or “slack” that we are willing to cut the classification on the data
- If $C = 0$, we are back to maximal-margin classification
- Whatever the value of C, at most C samples can be misclassified (though more than C samples can be inside the margin)
- As C increases, there is increasing tolerance for misclassification and margin violations. So, the margin will widen

Image Source

23



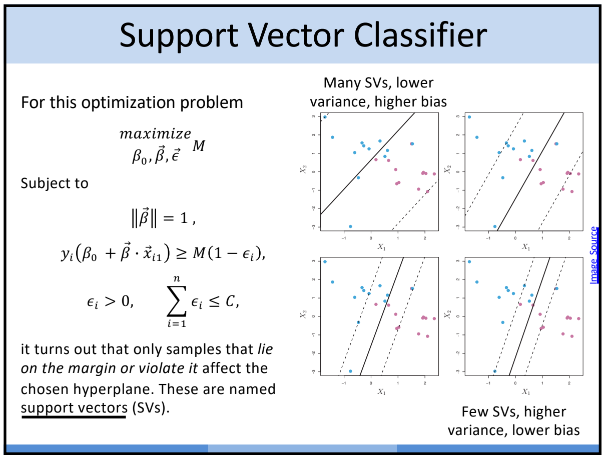
Support Vector Classifier

In practice, C is treated like a tuning parameter and is typically selected via cross-validation.

C controls the bias-variance tradeoff. Smaller C results in a classifier with narrower margin that more tightly fits the data. So, it has a lower bias but a higher variance. A larger C results in a wider margin with increased violations. So, the bias increases and the variance decreases.

Image Source

24



Support Vector Classifier

For this optimization problem

$$\underset{\beta_0, \vec{\beta}, \vec{\epsilon}}{maximize}\ M$$

Subject to

$$\|\vec{\beta}\| = 1\,,$$

$$y_i\left(\beta_0 + \vec{\beta} \cdot \vec{x}_{i1}\right) \geq M(1 - \epsilon_i),$$

$$\epsilon_i > 0, \qquad \sum_{i=1}^{n} \epsilon_i \leq C,$$

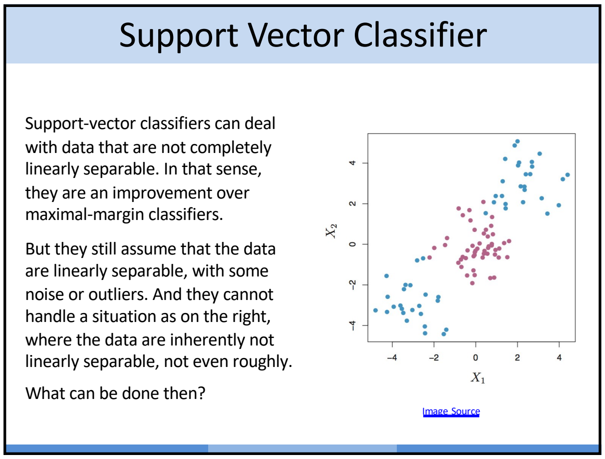
it turns out that only samples that *lie on the margin or violate it* affect the chosen hyperplane. These are named support vectors (SVs).

Many SVs, lower variance, higher bias

Few SVs, higher variance, lower bias

Image Source

25



Support Vector Classifier

Support-vector classifiers can deal with data that are not completely linearly separable. In that sense, they are an improvement over maximal-margin classifiers.

But they still assume that the data are linearly separable, with some noise or outliers. And they cannot handle a situation as on the right, where the data are inherently not linearly separable, not even roughly.

What can be done then?

Image Source

26

Feature Expansion

One solution is to enlarge the feature space by transforming the variables, for instance, to a cubed space:

$$(x_1, x_2)\ {}^{\varphi}\!\!\Rightarrow (x_1, x_2, {x_1}^2, {x_2}^2, {x_1}^3, {x_2}^3)$$

Or

$$\vec{x}\ {}^{\varphi}\!\!\Rightarrow (\,{}^{1}\vec{x},\ {}^{2}\vec{x},\ {}^{3}\vec{x}).$$

The optimization problem then becomes:

$$\underset{\beta_0,\ {}^{1}\vec{\beta},\ {}^{2}\vec{\beta},\ {}^{3}\vec{\beta}, \vec{\epsilon}}{maximize}\ M$$

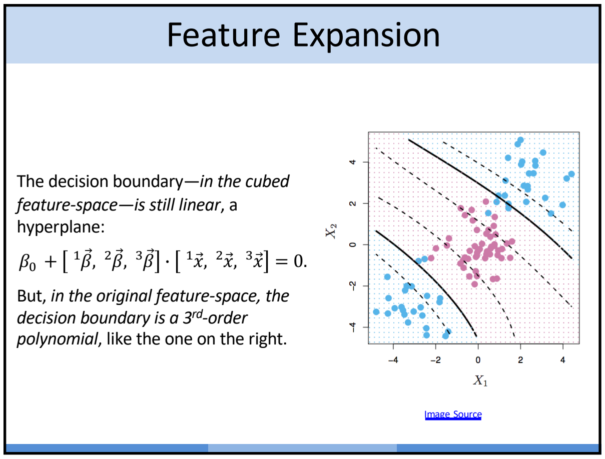
Subject to

$$\|[\,{}^{1}\vec{\beta},\ {}^{2}\vec{\beta},\ {}^{3}\vec{\beta}]\| = 1\,,$$

$$y_i\left(\beta_0 + [\,{}^{1}\vec{\beta},\ {}^{2}\vec{\beta},\ {}^{3}\vec{\beta}] \cdot [\,{}^{1}\vec{x},\ {}^{2}\vec{x},\ {}^{3}\vec{x}]\right) \geq M(1 - \epsilon_i),$$

$$\epsilon_i > 0, \qquad \sum_{i=1}^{n} \epsilon_i \leq C,$$

27



Feature Expansion

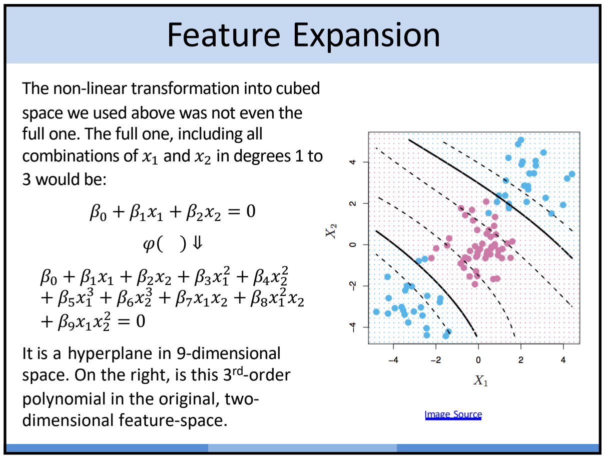
The decision boundary—*in the cubed feature-space—is still linear*, a hyperplane:

$$\beta_0 + [\,{}^{1}\vec{\beta},\ {}^{2}\vec{\beta},\ {}^{3}\vec{\beta}] \cdot [\,{}^{1}\vec{x},\ {}^{2}\vec{x},\ {}^{3}\vec{x}] = 0.$$

But, *in the original feature-space, the decision boundary is a 3rd-order polynomial*, like the one on the right.

Image Source

28



Feature Expansion

The non-linear transformation into cubed space we used above was not even the full one. The full one, including all combinations of x_1 and x_2 in degrees 1 to 3 would be:

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$$

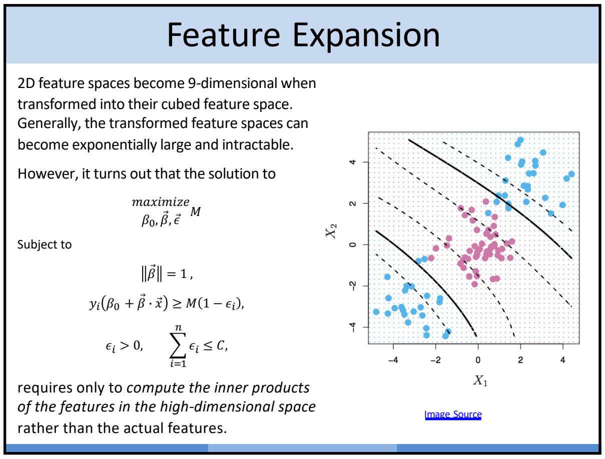
$$\varphi(\quad) \Downarrow$$

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1^2 + \beta_4 x_2^2 + \beta_5 x_1^3 + \beta_6 x_2^3 + \beta_7 x_1 x_2 + \beta_8 x_1^2 x_2 + \beta_9 x_1 x_2^2 = 0$$

It is a hyperplane in 9-dimensional space. On the right, is this 3rd-order polynomial in the original, two-dimensional feature-space.

Image Source

29



Feature Expansion

2D feature spaces become 9-dimensional when transformed into their cubed feature space. Generally, the transformed feature spaces can become exponentially large and intractable.

However, it turns out that the solution to

$$\underset{\beta_0, \vec{\beta}, \vec{\epsilon}}{maximize}\ M$$

Subject to

$$\|\vec{\beta}\| = 1\,,$$

$$y_i\left(\beta_0 + \vec{\beta} \cdot \vec{x}\right) \geq M(1 - \epsilon_i),$$

$$\epsilon_i > 0, \qquad \sum_{i=1}^{n} \epsilon_i \leq C,$$

requires only to *compute the inner products of the features in the high-dimensional space* rather than the actual features.

Image Source

30

Feature Expansion

The solution to

$$\underset{\beta_0, \vec{\beta}, \vec{\epsilon}}{maximize} M$$

Subject to

$$\|\vec{\beta}\| = 1\,,$$
$$y_i(\beta_0 + \vec{\beta} \cdot \vec{x}) \geq M(1 - \epsilon_i),$$
$$\epsilon_i > 0, \qquad \sum_{i=1}^{n} \epsilon_i \leq C,$$

requires only to *compute the inner products of the features in the high-dimensional space* rather than the actual features.

It further turns out that *these inner products involve only the support vectors.*

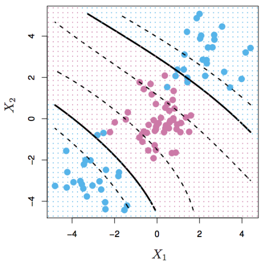


Image Source

31

SVMs and Kernels

Therefore, we could replace inner product computations with their generalization to higher-dimensional space

$${}_i\vec{x} \cdot {}_j\vec{x} \Longrightarrow \varphi({}_i\vec{x}) \cdot \varphi({}_j\vec{x}) = K({}_i\vec{x},\ {}_j\vec{x}).$$

To the right, we have

$$K({}_i\vec{x},\ {}_j\vec{x}) = \left(1 + {}_i\vec{x} \cdot {}_j\vec{x}\right)^d,$$

with $d = 3$. These generalizations are called kernels.

$K({}_i\vec{x},\ {}_j\vec{x}) = \left(1 + {}_i\vec{x} \cdot {}_j\vec{x}\right)^3$

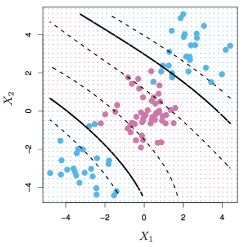


Image Source

32

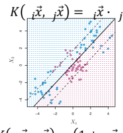
SVMs and Kernels

A few common kernels are

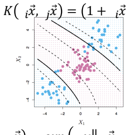
- Linear kernel (support-vector classifiers):

$$K({}_i\vec{x},\ {}_j\vec{x}) = {}_i\vec{x} \cdot {}_j\vec{x}$$

$K({}_i\vec{x},\ {}_j\vec{x}) = {}_i\vec{x} \cdot {}_j\vec{x}$



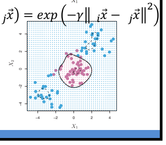
$K({}_i\vec{x},\ {}_j\vec{x}) = \left(1 + {}_i\vec{x} \cdot {}_j\vec{x}\right)^d$



- Polynomial kernel (of degree d):

$$K({}_i\vec{x},\ {}_j\vec{x}) = \left(1 + {}_i\vec{x} \cdot {}_j\vec{x}\right)^d$$

$K({}_i\vec{x},\ {}_j\vec{x}) = exp\left(-\gamma\|{}_i\vec{x} - {}_j\vec{x}\|^2\right)$



- RBF kernel:

$$K({}_i\vec{x},\ {}_j\vec{x}) = exp\left(-\gamma\|{}_i\vec{x} - {}_j\vec{x}\|^2\right)$$

33

SVMs and Kernels: RBF

The most common kernel is the *radial-basis function (RBF)* or *Gaussian* kernel:

$$K({}_i\vec{x},\ {}_j\vec{x}) = exp\left(-\gamma\|{}_i\vec{x} - {}_j\vec{x}\|^2\right).$$

This function decays exponentially with the distance between ${}_i\vec{x}$ and ${}_j\vec{x}$. So only very local information matters to it for class-label prediction. It is thus reminiscent of *k*-NN.

$K({}_i\vec{x},\ {}_j\vec{x}) = exp\left(-\gamma\|{}_i\vec{x} - {}_j\vec{x}\|^2\right)$

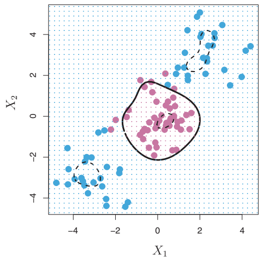


Image Source

34

SVMs and Kernels: RBF

We saw how polynomial kernels are expansions into higher-dimensional feature-spaces. What is the dimension of the RBF-kernel expansion?

The full expansion associated with RBF is into infinite-dimensional space. This is because the polynomial expansion (Taylor series) of exponentials is infinite:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$$

$K({}_i\vec{x},\ {}_j\vec{x}) = exp\left(-\gamma\|{}_i\vec{x} - {}_j\vec{x}\|^2\right)$

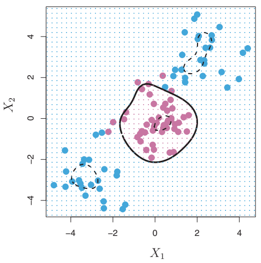


Image Source

35

Alternative SVM Formulation

It has been proven that an alternative formulation to

$$\underset{\beta_0, \vec{\beta}, \vec{\epsilon}}{maximize} M$$

Subject to

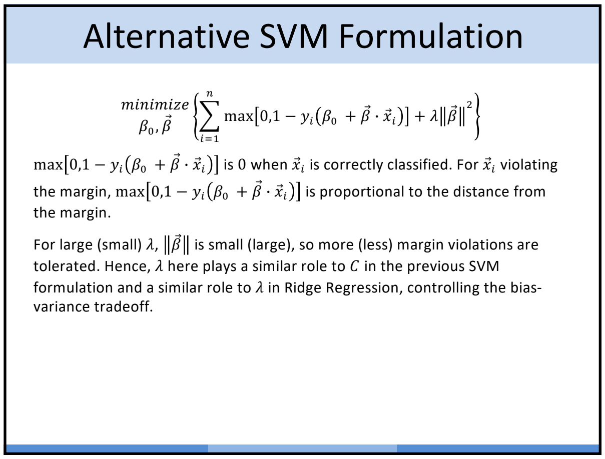
$$\|\vec{\beta}\| = 1\,,$$
$$y_i(\beta_0 + \vec{\beta} \cdot \vec{x}) \geq M(1 - \epsilon_i),$$
$$\epsilon_i > 0, \qquad \sum_{i=1}^{n} \epsilon_i \leq C,$$

is

$$\underset{\beta_0, \vec{\beta}}{minimize} \left\{\sum_{i=1}^{n} \max\left[0{,}1 - y_i\left(\beta_0 + \vec{\beta} \cdot \vec{x}_i\right)\right] + \lambda\|\vec{\beta}\|^2\right\}.$$

Here $\lambda > 0$ is a tuning parameter and $\max\left[0{,}1 - y_i\left(\beta_0 + \vec{\beta} \cdot \vec{x}_{i1}\right)\right]$ is “hinge loss”.

36



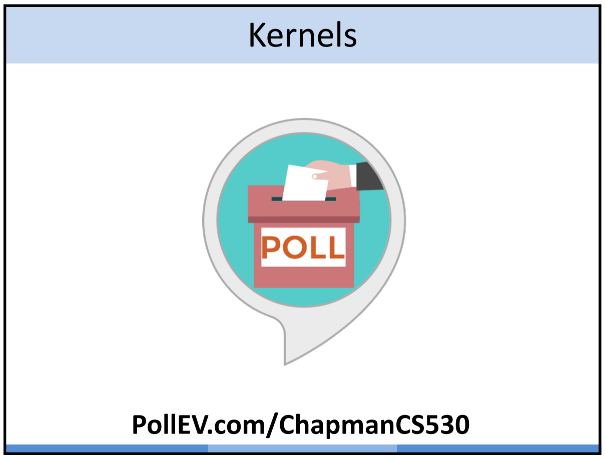
Alternative SVM Formulation

$$\underset{\beta_0,\vec{\beta}}{minimize}\left\{\sum_{i=1}^{n}\max[0,1-y_i(\beta_0+\vec{\beta}\cdot\vec{x}_i)]+\lambda\|\vec{\beta}\|^2\right\}$$

$\max[0,1-y_i(\beta_0+\vec{\beta}\cdot\vec{x}_i)]$ is 0 when $\vec{x}_i$ is correctly classified. For $\vec{x}_i$ violating the margin, $\max[0,1-y_i(\beta_0+\vec{\beta}\cdot\vec{x}_i)]$ is proportional to the distance from the margin.

For large (small) λ, $\|\vec{\beta}\|$ is small (large), so more (less) margin violations are tolerated. Hence, λ here plays a similar role to C in the previous SVM formulation and a similar role to λ in Ridge Regression, controlling the bias-variance tradeoff.

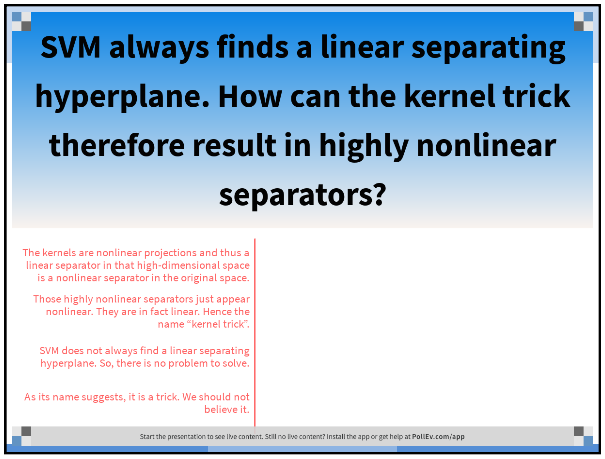
37



Kernels

PollEV.com/ChapmanCS530

38



SVM always finds a linear separating hyperplane. How can the kernel trick therefore result in highly nonlinear separators?

The kernels are nonlinear projections and thus a linear separator in that high-dimensional space is a nonlinear separator in the original space.

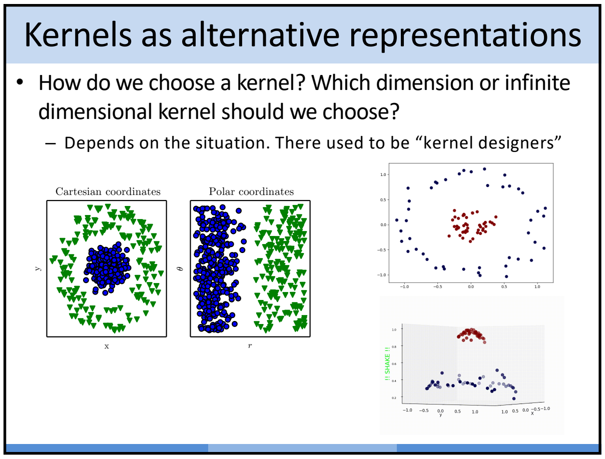
Those highly nonlinear separators just appear nonlinear. They are in fact linear. Hence the name “kernel trick”.

SVM does not always find a linear separating hyperplane. So, there is no problem to solve.

As its name suggests, it is a trick. We should not believe it.

Start the presentation to see live content. Still no live content? Install the app or get help at **PollEv.com/app**

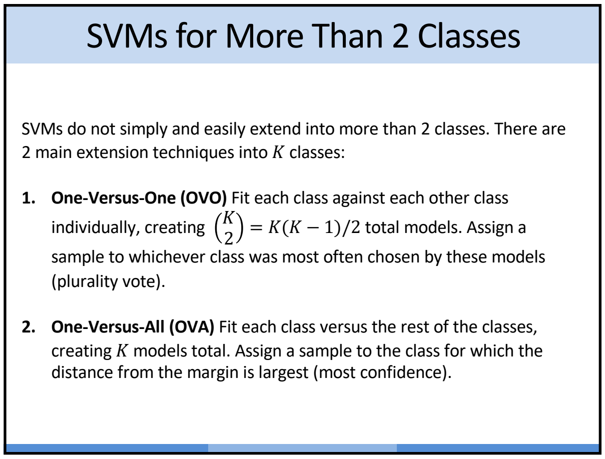
39



Kernels as alternative representations

- How do we choose a kernel? Which dimension or infinite dimensional kernel should we choose?
  - Depends on the situation. There used to be “kernel designers”

40

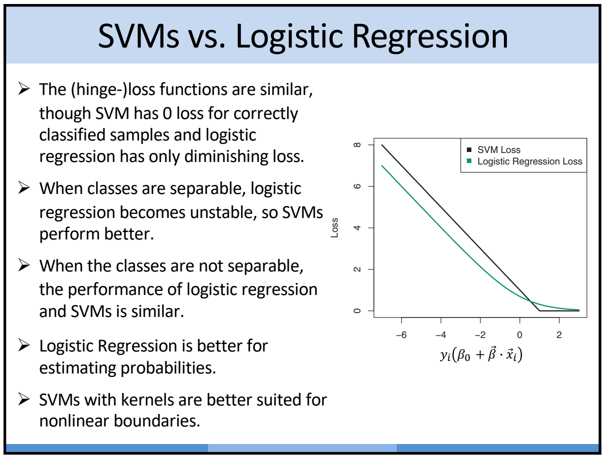


SVMs for More Than 2 Classes

SVMs do not simply and easily extend into more than 2 classes. There are 2 main extension techniques into K classes:

1. **One-Versus-One (OVO)** Fit each class against each other class individually, creating $\binom{K}{2}=K(K-1)/2$ total models. Assign a sample to whichever class was most often chosen by these models (plurality vote).
2. **One-Versus-All (OVA)** Fit each class versus the rest of the classes, creating K models total. Assign a sample to the class for which the distance from the margin is largest (most confidence).

41



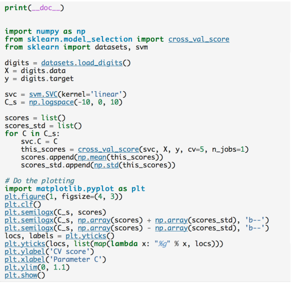
SVMs vs. Logistic Regression

- The (hinge-)loss functions are similar, though SVM has 0 loss for correctly classified samples and logistic regression has only diminishing loss.
- When classes are separable, logistic regression becomes unstable, so SVMs perform better.
- When the classes are not separable, the performance of logistic regression and SVMs is similar.
- Logistic Regression is better for estimating probabilities.
- SVMs with kernels are better suited for nonlinear boundaries.

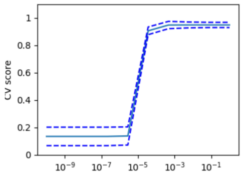
■ SVM Loss
■ Logistic Regression Loss
Loss
0
2
4
6
8
−6
−4
−2
0
2
$y_i(\beta_0+\vec{\beta}\cdot\vec{x}_i)$

42

SVM Scikit-example



```
print(__doc__)


import numpy as np
from sklearn.model_selection import cross_val_score
from sklearn import datasets, svm

digits = datasets.load_digits()
X = digits.data
y = digits.target

svc = svm.SVC(kernel='linear')
C_s = np.logspace(-10, 0, 10)

scores = list()
scores_std = list()
for C in C_s:
    svc.C = C
    this_scores = cross_val_score(svc, X, y, cv=5, n_jobs=1)
    scores.append(np.mean(this_scores))
    scores_std.append(np.std(this_scores))

# Do the plotting
import matplotlib.pyplot as plt
plt.figure(1, figsize=(4, 3))
plt.clf()
plt.semilogx(C_s, scores)
plt.semilogx(C_s, np.array(scores) + np.array(scores_std), 'b--')
plt.semilogx(C_s, np.array(scores) - np.array(scores_std), 'b--')
locs, labels = plt.yticks()
plt.yticks(locs, list(map(lambda x: "%g" % x, locs)))
plt.ylabel('CV score')
plt.xlabel('Parameter C')
plt.ylim(0, 1.1)
plt.show()
```



43

Summary

- Support Vector Machines (SVMs)
  - The hyperplane with the largest margin is a good separator between linearly separable classes due to high generalizability to unseen samples.
  - When the classes are linearly separable with noise, we can budget the amount of margin violations and misclassification we allow
  - When the data are highly non-linearly separable, we can use the kernel trick to find separators in higher- (or even infinite-) dimensional space that then correspond to non-linear curves in the original space.
- SVM has connections with other classification functions
  - RBF-kernel results are similar to *k*-NN
  - SVM in general is similar to logistic regression

44

Take 🏠 message for rest of course: SVM

- SVM is a method for finding maximum-margin linear classifiers in feature space
- Kernels enable SVM (and other methods) to work on (highly) nonlinearly separable data
- SVM is not the unique, distinct method it was thought to be at first. It has deep connections with other methods like logistic regression and *k*-NN

45