6.867 Fall 2017
Introduction to Classification

Kernels

Derived in part from Alex Smola's lectures for CMU 10-701

Lecture 8: 3rd Oct, 2017

Admin

HW2 out

HW1 Reviews due

Exercises 4 out

Exam: Material up to 10/5 is fair game

Outline

- ★ Entering nonlinear lifestyle
- ★ Nonlinear features
- ★ Kernel functions
  - ★ Some theory
- ★ Kernelizing algorithms

```
Delivered-To: suvrit@gmail.com
Received: by 10.157.56.186 with SMTP id p55csp703174otc;
        Sat, 30 Sep 2017 07:34:01 -0700 (PDT)
X-Google-Smtp-Source: AOwi7QCK9y8KOdBTvMLfrbOcOxH51i9LN6lkw+lhc/IkwZLmb8UrH7HW7JKfqLt7xj1TwDsf3dX1
X-Received: by 10.80.193.26 with SMTP id l26mr13268612edf.97.1506782040979;
        Sat, 30 Sep 2017 07:34:00 -0700 (PDT)
ARC-Seal: i=1; a=rsa-sha256; t=1506782040; cv=none;
        d=google.com; s=arc-20160816;
        b=BxVulFx86G2rfNXnEsv2kNx5pFgiU0jkZKmJ6TwkNBg/dr5MrDXJsGEihegHN7u0IU
         6J1J5kn6q8sMk84OaGfeeDAOatE37P2Strqc4c4BivUY2YKD3wxp31wstrE5hLYDw1od
         lmzTjTzfWZKv1AB1fRuKDCgn+bnpIvWyL03aEo5dMiuztpvA0HId90leNsxJo442Cn3J
         0s3kVvw4nJZVoDUO5zsLwD8XrHFP8M6k739NXdTLfVyci78BjQseVXrlf8a/ya/fHfvL
         xYl07foHl7nXUvVprfx/D9EooB+8v0e/IjiBHFf1lnPNAi6fcx1p53S70dNQ0VJss/GM
         rFdA==
ARC-Message-Signature: i=1; a=rsa-sha256; c=relaxed/relaxed; d=google.com; s=arc-20160816;
        h=feedback-id:date:to:reply-to:from:subject:message-id:mime-version
         :dkim-signature:dkim-signature:arc-authentication-results;
        bh=R4godP4US7Q4tcEB7kBYBKAfq6im2AktH8nd+AKvSNY=;
        b=oJa78ArtTLtDeN2+Byqt5hil7chs4JrD2djnrXA5t6u7Jab5SfRXYWY9y9YNOl6Cqu
         3a58btCCKgs1e84Rf2GNlAv7+zdV2rpGmI79cQ6dGKPQPItshGRzGMCl39tprnAUOGvd
         jLYV8/o/RCejaHDelJ/UKPRaLjrAtTYjaDOL5wJSTkjkjBgiR8dP2lea7xmLnPZ+cB2H
         FkZIGrEOpG4in01bfx1yvapCv9vrwtR4+DFAOGbZxE4NRdq7CUu9ewTTpjkFmaFZMXB8
         SwrDrkh8P86DUjcCsOlIYUp7EVQdw7Lc0p7AMP1ZaFkWoIuTcwShzDUe6Q1MAZudto0B
         xGsQ==
ARC-Authentication-Results: i=1; mx.google.com;
       dkim=pass header.i=@axolbio.com header.s=yeey7jbyihrzecpea7e3xbglnfzh65yg header.b=N75Hyb7v;
       dkim=pass header.i=@amazonses.com header.s=shh3fegwg5fppqsuzphvschd53n6ihuv header.b=WkaTV2p1;
       spf=pass (google.com: domain of
0102015ed3354feb-392b28ee-3ca2-485c-8b4d-87aff57c8b67-000000@amazonses.axolbio.com designates 54.240.4.3
as permitted sender)
smtp.mailfrom=0102015ed3354feb-392b28ee-3ca2-485c-8b4d-87aff57c8b67-000000@amazonses.axolbio.com;
       dmarc=pass (p=NONE sp=NONE dis=NONE) header.from=axolbio.com
Return-Path: <0102015ed3354feb-392b28ee-3ca2-485c-8b4d-87aff57c8b67-000000@amazonses.axolbio.com>
Received: from a4-3.smtp-out.eu-west-1.amazonses.com (a4-3.smtp-out.eu-west-1.amazonses.com.
[54.240.4.3])
        by mx.google.com with ESMTPS id x13si6722343edx.227.2017.09.30.07.34.00
        for <suvrit@gmail.com>
        (version=TLS1 cipher=ECDHE-RSA-AES128-SHA bits=128/128);
        Sat, 30 Sep 2017 07:34:00 -0700 (PDT)
Received-SPF: pass (google.com: domain of
0102015ed3354feb-392b28ee-3ca2-485c-8b4d-87aff57c8b67-000000@amazonses.axolbio.com designates 54.240.4.3
as permitted sender) client-ip=54.240.4.3;
Authentication-Results: mx.google.com;
       dkim=pass header.i=@axolbio.com header.s=yeey7jbyihrzecpea7e3xbglnfzh65yg header.b=N75Hyb7v;
       dkim=pass header.i=@amazonses.com header.s=shh3fegwg5fppqsuzphvschd53n6ihuv header.b=WkaTV2p1;
       spf=pass (google.com: domain of
0102015ed3354feb-392b28ee-3ca2-485c-8b4d-87aff57c8b67-000000@amazonses.axolbio.com designates 54.240.4.3
as permitted sender)
smtp.mailfrom=0102015ed3354feb-392b28ee-3ca2-485c-8b4d-87aff57c8b67-000000@amazonses.axolbio.com;
       dmarc=pass (p=NONE sp=NONE dis=NONE) header.from=axolbio.com
DKIM-Signature: v=1; a=rsa-sha256; q=dns/txt; c=relaxed/simple; s=yeey7jbyihrzecpea7e3xbglnfzh65yg;
d=axolbio.com; t=1506782040; h=Content-Type:MIME-Version:Message-Id:Subject:From:Reply-To:To:Date;
bh=R4godP4US7Q4tcEB7kBYBKAfq6im2AktH8nd+AKvSNY=; b=N75Hyb7vZ3s13rX
+JqDoN5U5Yoo9cwUTbNRLI9KUrG07oGpokoLxYWPhL8WcTw21 IkTsnw2S1M3jPmX5gVrcqehxtE7CkjnkaX0z4P
+q31nSYOoeOvmxN35SjB6tNBPGMvh lczINQOPSKtSkd2y812VB4hrBg6ToPL2dP3xYtcg=
DKIM-Signature: v=1; a=rsa-sha256; q=dns/txt; c=relaxed/simple; s=shh3fegwg5fppqsuzphvschd53n6ihuv;
d=amazonses.com; t=1506782040; h=Content-Type:MIME-Version:Message-Id:Subject:From:Reply-
To:To:Date:Feedback-ID; bh=R4godP4US7Q4tcEB7kBYBKAfq6im2AktH8nd+AKvSNY=; b=WkaTV2p1v987maaZaWS/
6JbrUq13AC7JNw12Jq0jP/pDyF20WALENqNXB1Er+4Wj AXU3ARVMePb+hHSAyqKghaWIo6iCH1u7XWV9kbHz1k69LK+/
cg2Dci8zIyhWZ8PsywW +x4qLfirwM2uQTqYnArzLe1sq/lcl6Wtfn3AulVg=
Content-Type: multipart/mixed; boundary="===============5772367175723715400=="
MIME-Version: 1.0
Message-ID: <0102015ed3354feb-392b28ee-3ca2-485c-8b4d-87aff57c8b67-000000@eu-west-1.amazonses.com>
Subject: Come meet us at Drug Discovery 2017 in Liverpool!
From: Axol Bioscience <query@axolbio.com>
Reply-To: Axol Bioscience <query@axolbio.com>
To: suvrit@gmail.com
Date: Sat, 30 Sep 2017 14:34:00 +0000
X-SES-Outgoing: 2017.09.30-54.240.4.3
Feedback-ID: 1.eu-west-1.dsZhP2MFwA9Gu8ylynbiR1E1xEwjZOlznrjLL2AvoJg=:AmazonSES
```

Feature Engineering for Spam Filtering

- bag of words
- pairs of words
- date & time
- recipient path
- IP number
- sender
- encoding
- links
- ... secret sauce ...

Constructing Features: naive OCR system

	1	2	3	4	5	6	7	8	9	0
Loops	0	0	0	1	0	1	0	2	1	1
3 Joints	0	0	0	0	0	1	0	0	1	0
4 Joints	0	0	0	1	0	0	0	1	0	0
Angles	0	1	1	1	1	0	1	0	0	0
Ink	1	2	2	2	2	2	1	3	2	2

Feature engineering

- Handwritten Japanese Character Recognition
  - Break down the images into strokes and recognize it
  - Lookup based on stroke order
- Medical Diagnosis
  - Physician's comments
  - Blood status / ECG / height / weight / temperature ...
  - Medical knowledge
- Preprocessing
  - Zero mean, unit variance to fix scaling (e.g. weight vs. income)
  - Probability integral transform (inverse CDF) as alternative
- Click through rate (CTR)
  - One-hot, or one-of-K encoding
  - But first need to decide "what raw features" to collect!

Difficult, expensive, create feature, hope it has discriminatory power

https://imgs.xkcd.com/comics/machine_learning.png

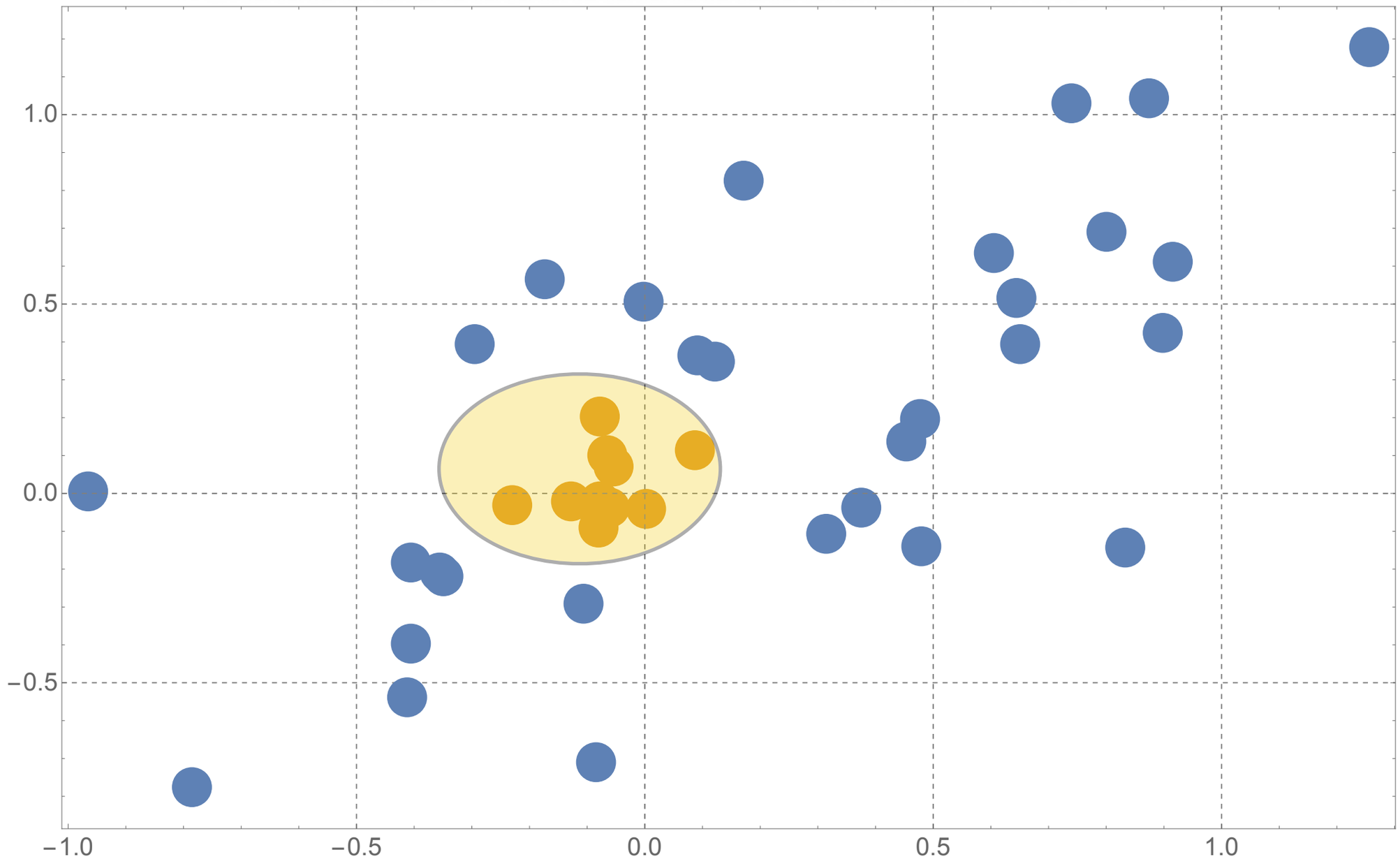
Nonlinear classification

Use a nonlinear classifier with given features

Use nonlinear features with linear classifier

Actually these ideas are not too different!

Nonlinear classifier



Predict +1 if $(x_1 - c_1)^2 + b(x_2 - c_2)^2 \leq 1$, otherwise predict -1

Nonlinear classifier

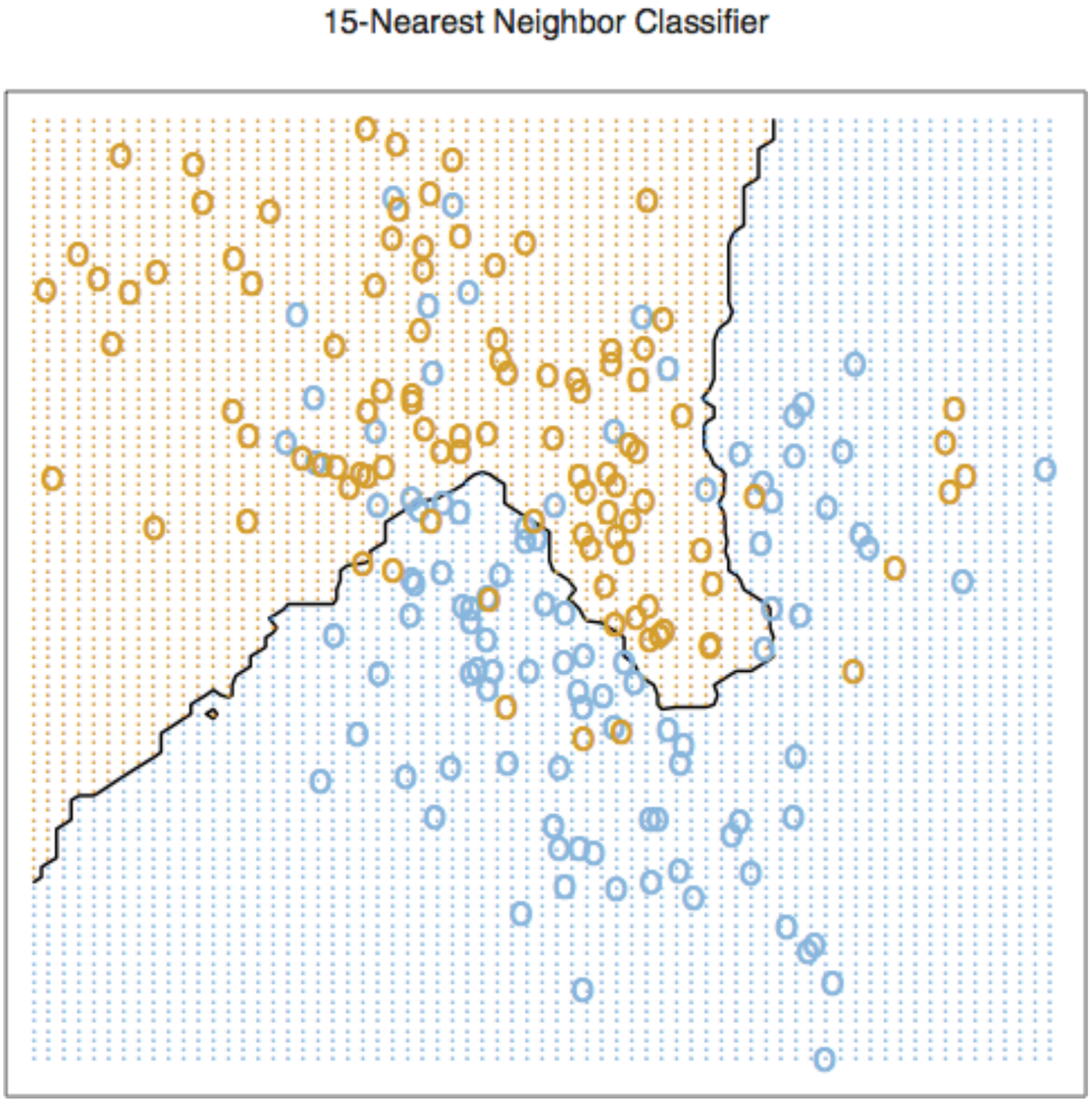
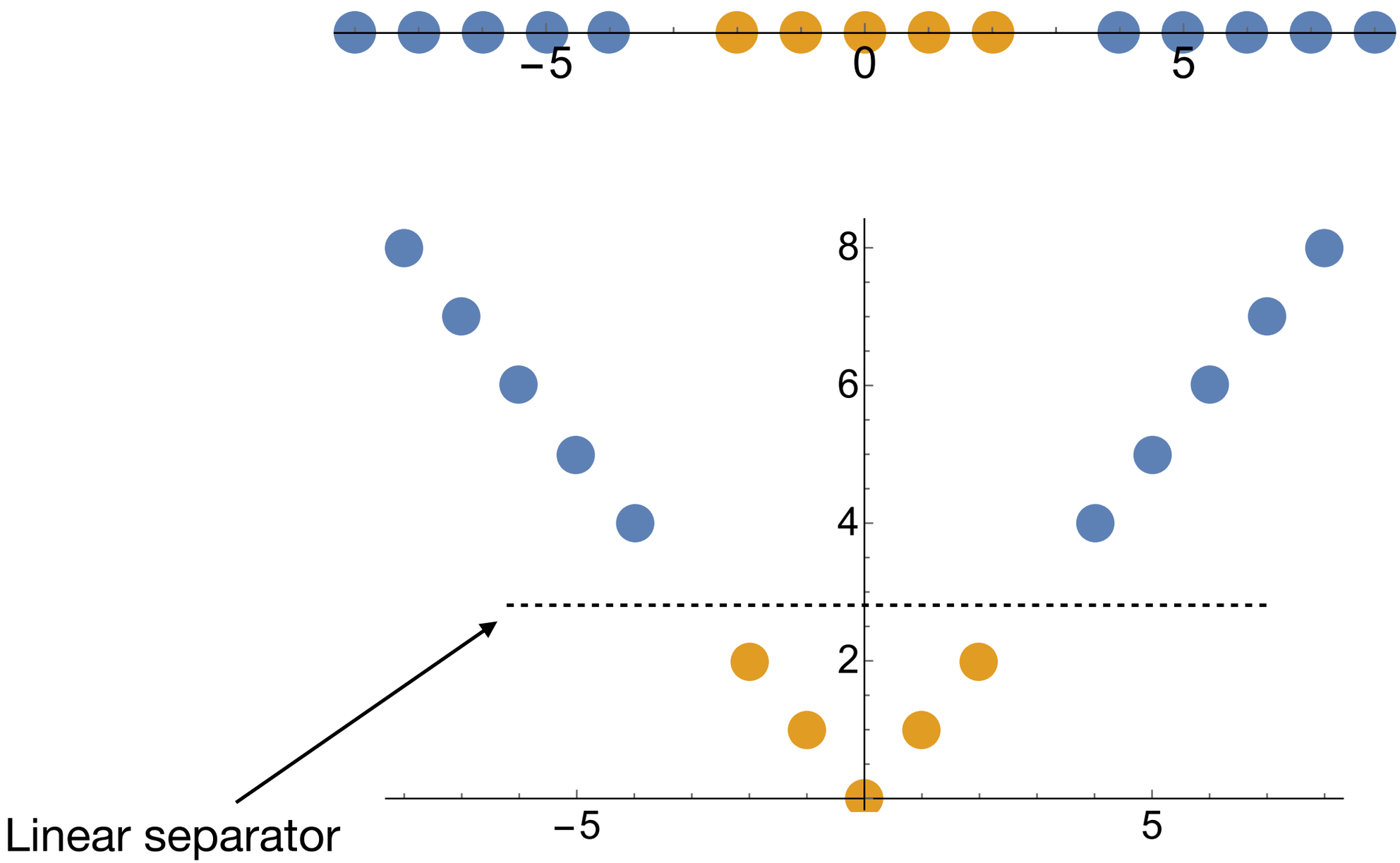


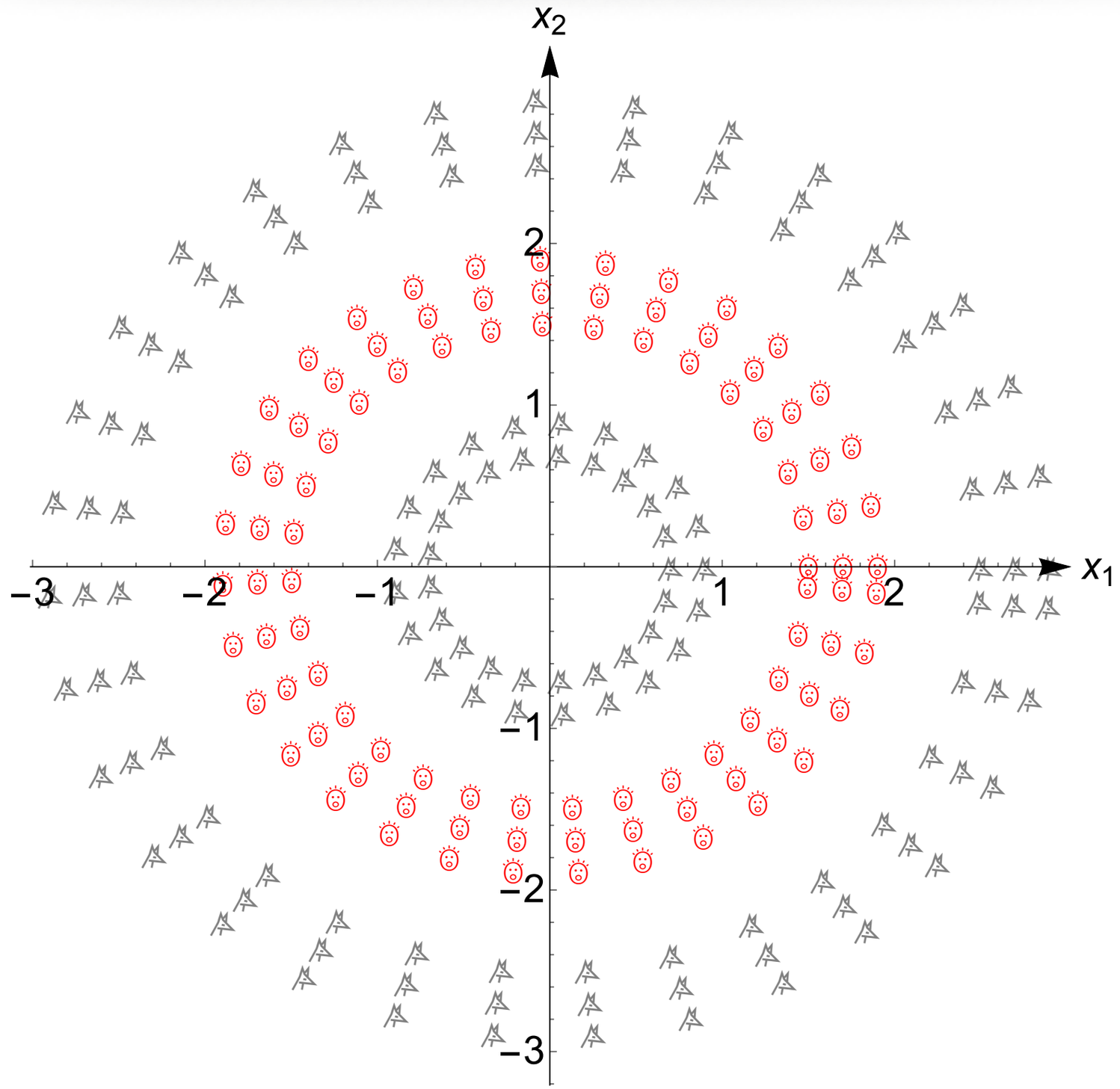
Image from: *Elements of Statistical Learning Theory*

Nonlinear features

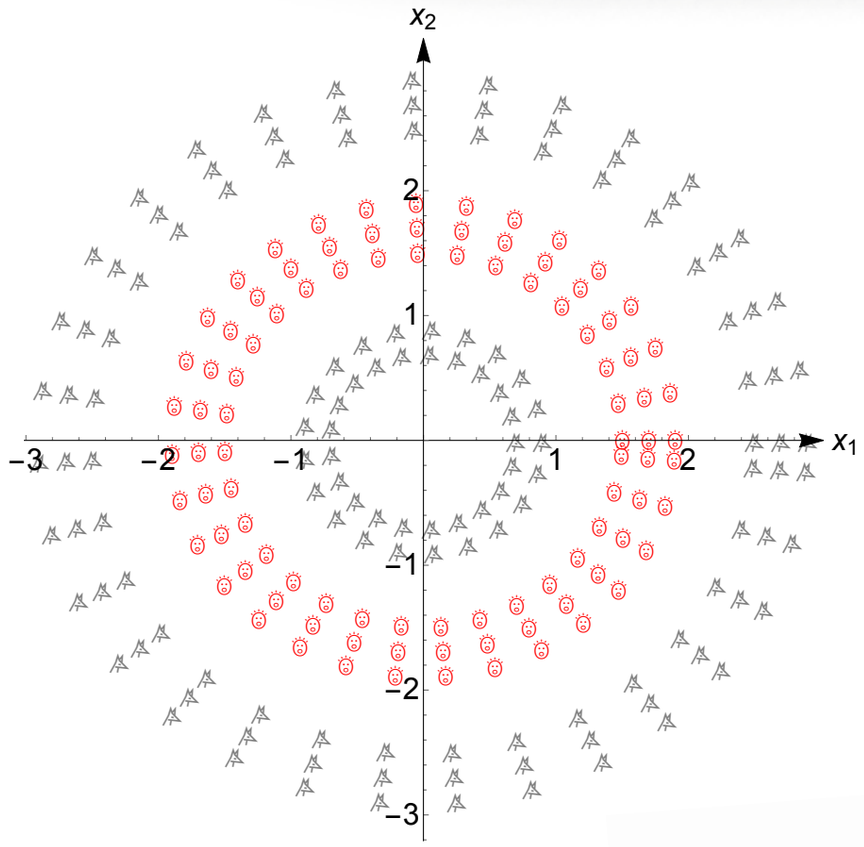


$$\phi(x) : x \mapsto (x, |x|)$$

Nonlinear features

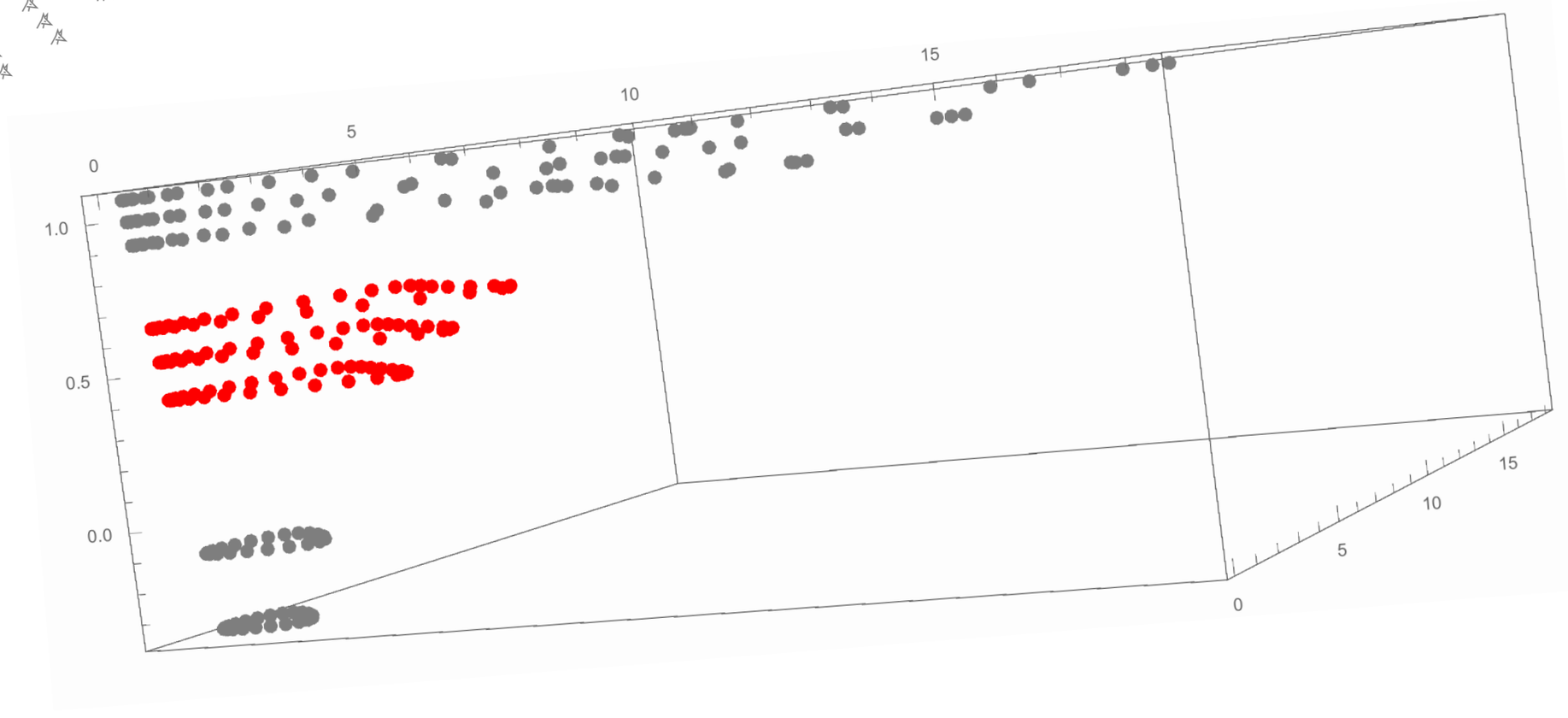


Nonlinear features

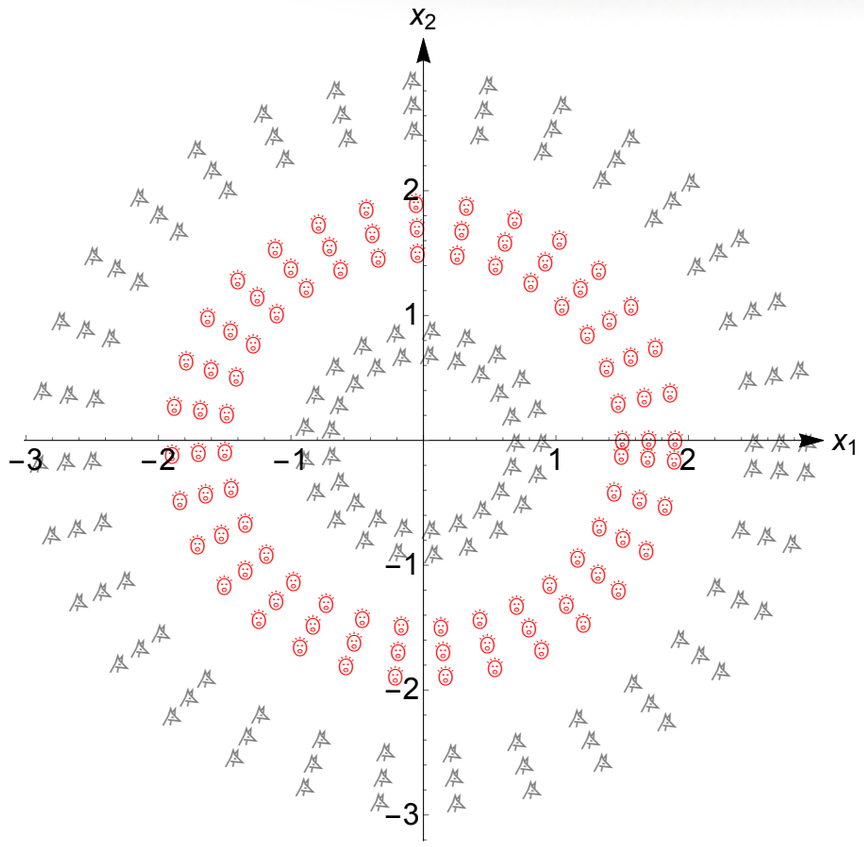


(create some random nonlinear features)

$$(e^{r\cos t}, e^{r\sin t}, \log r)$$

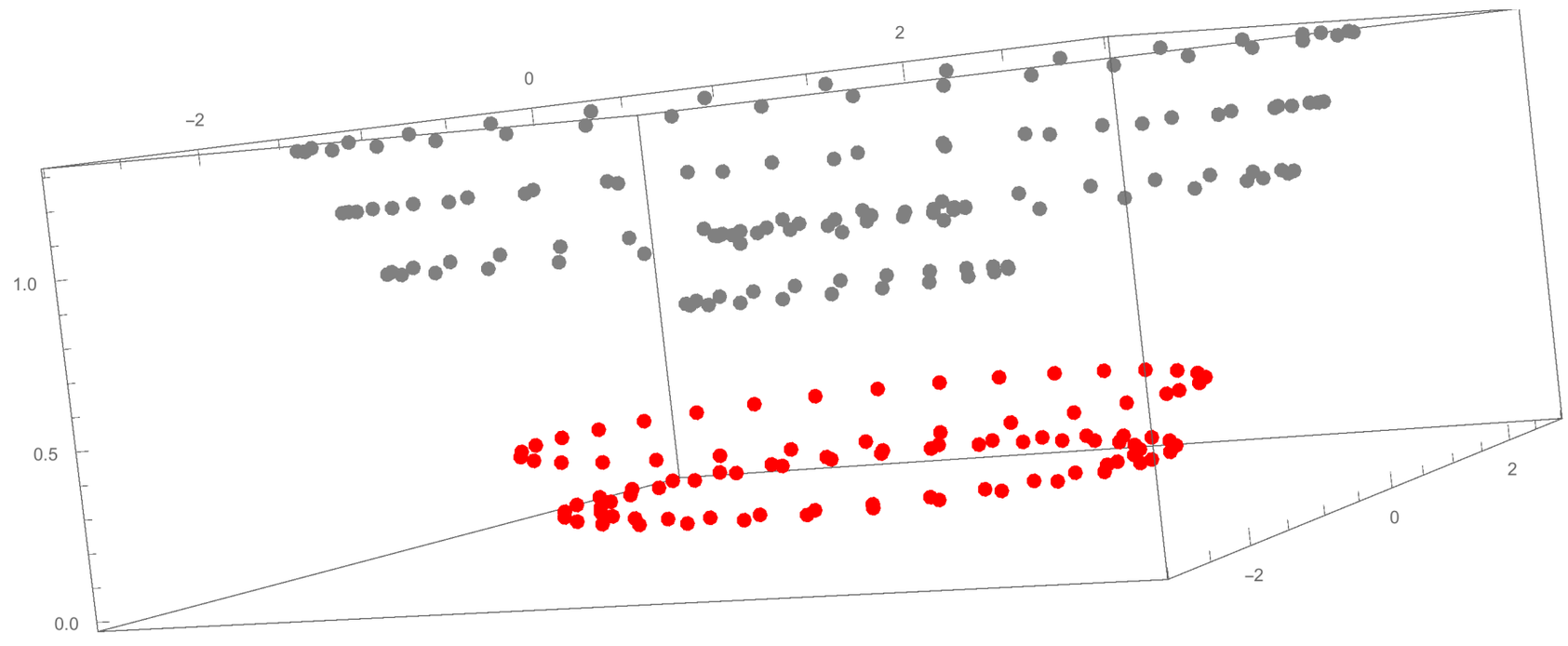


Nonlinear features



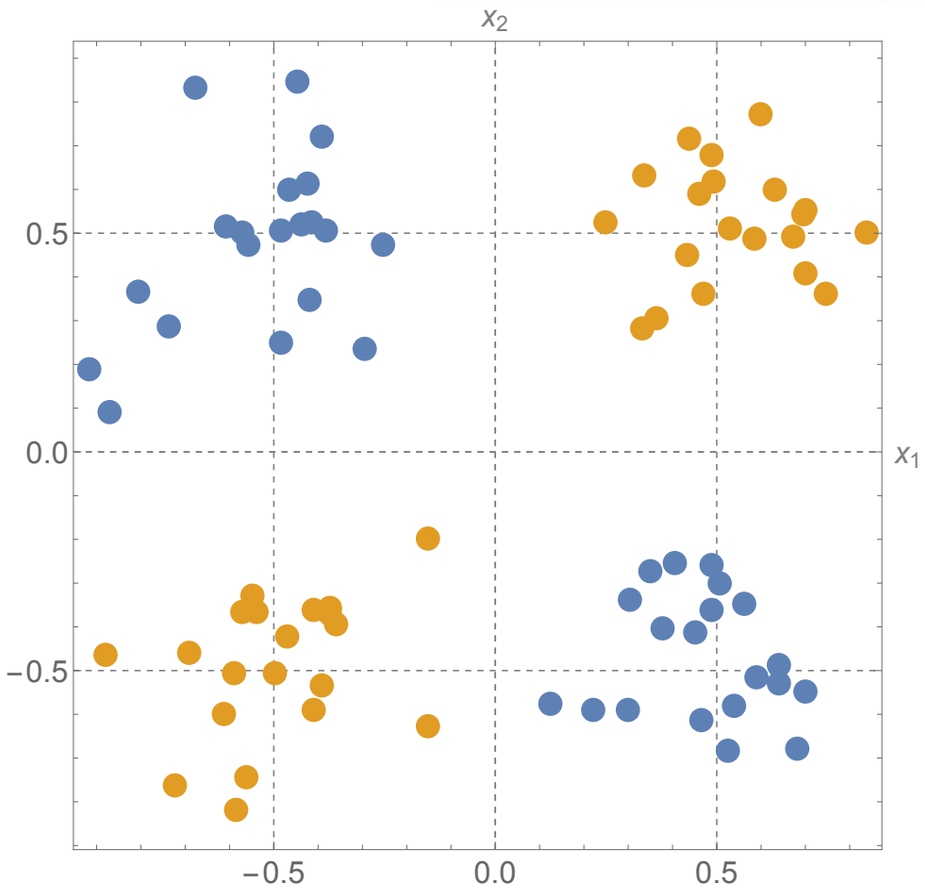
(slightly less random nonlinear features)

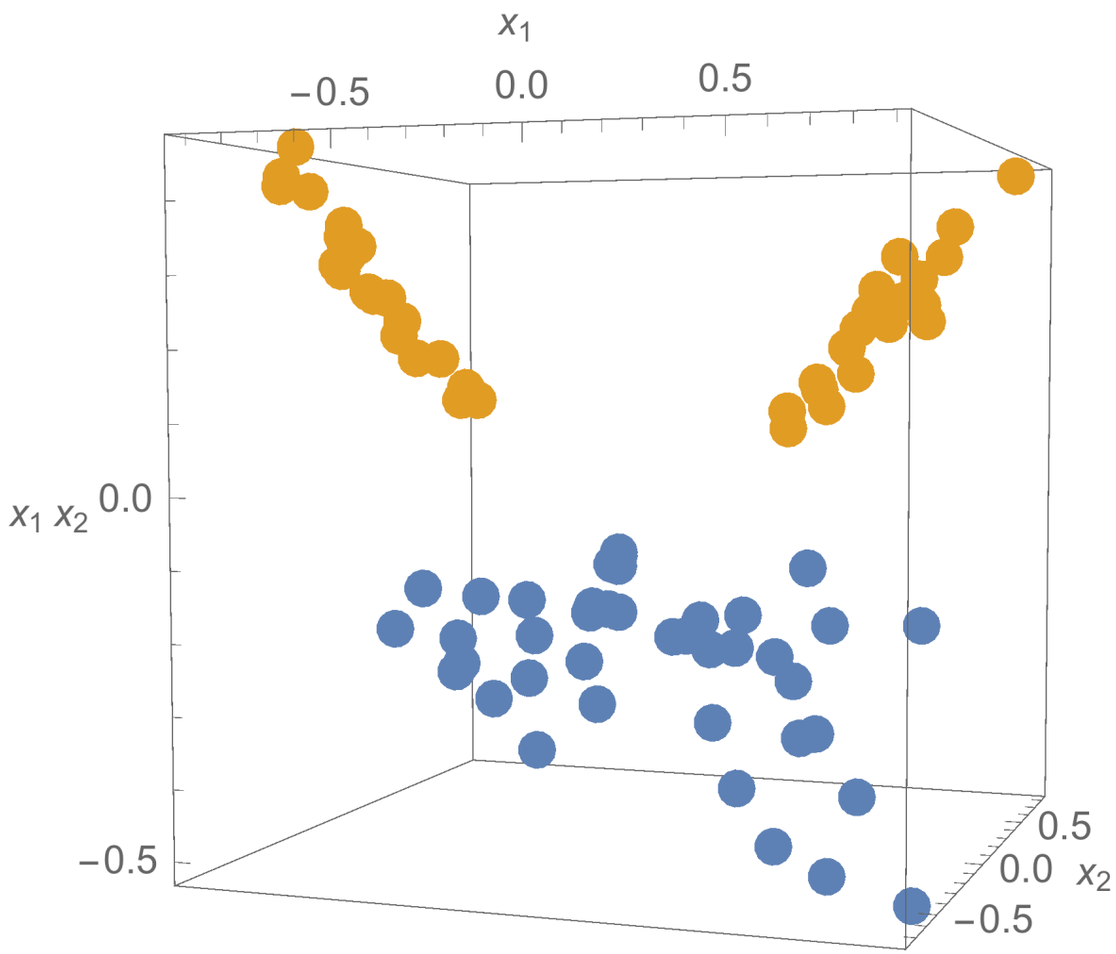
$$(r\cos t, r\sin t, |r - 3/2|)$$



Exercise: Create 1D nonlinear features that allow linear separation.

Quadratic features





$$\phi(x) : (x_1, x_2) \mapsto (x_1, x_1 x_2, x_2)$$

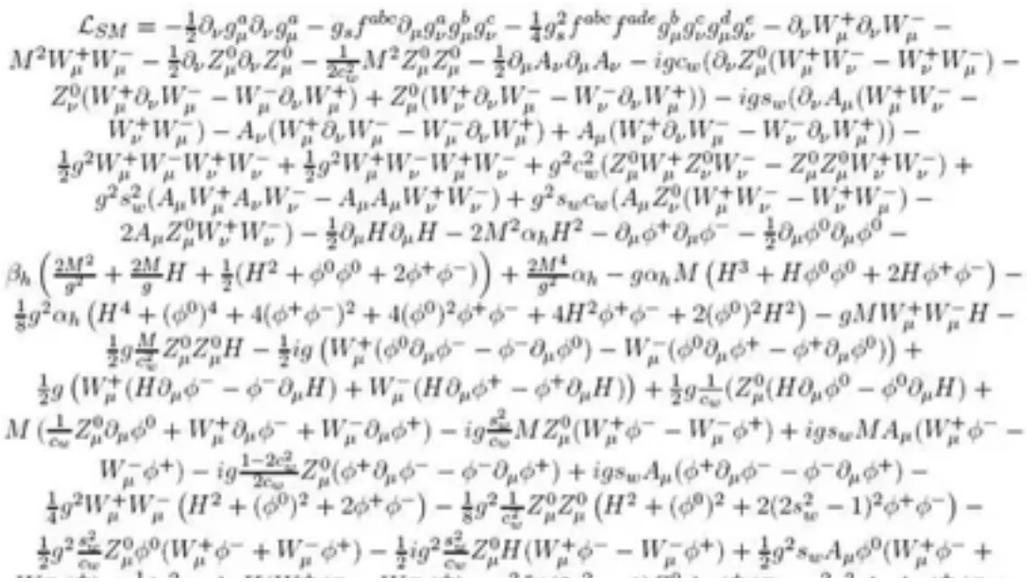
Nonlinear features

Linear $(x_1, \ldots, x_p)$

Quadratic $(x_1^2, x_1 x_2, \ldots, x_p^2)$

Polynomial $(x_1, x_1^2, \ldots, x_1^d, x_1 x_2, \ldots, x_1 x_2 x_3, \ldots)$

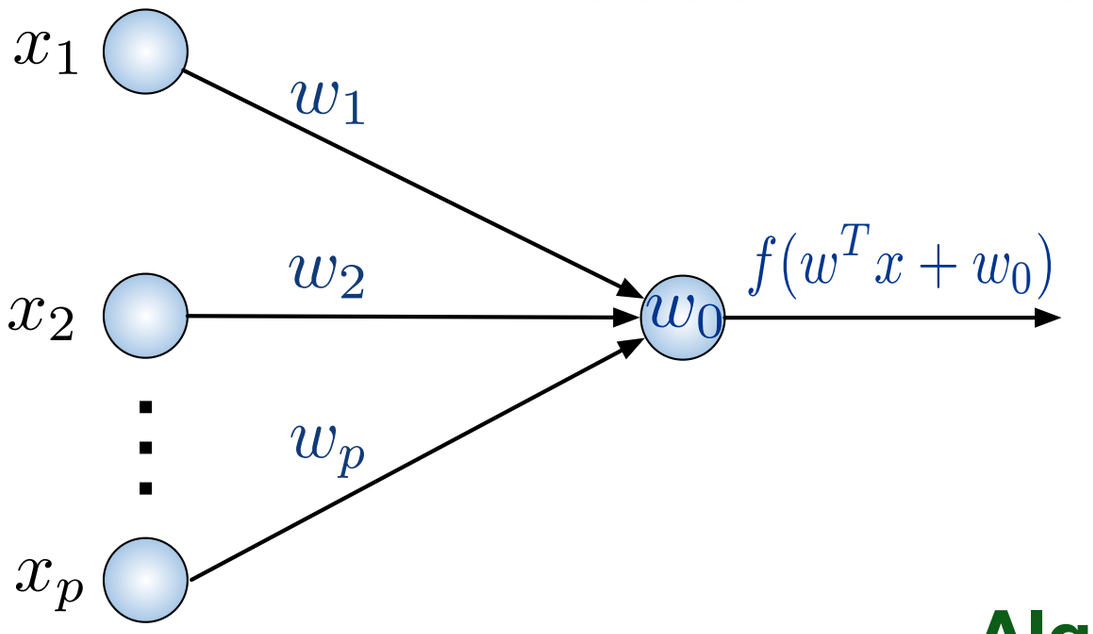
MCFs



$$\mathcal{L}_{SM} = -\tfrac{1}{2}\partial_\nu g^a_\mu \partial_\nu g^a_\mu - g_s f^{abc}\partial_\mu g^a_\nu g^b_\mu g^c_\nu - \tfrac{1}{4} g_s^2 f^{abc} f^{ade} g^b_\mu g^c_\nu g^d_\mu g^e_\nu - \partial_\nu W^+_\mu \partial_\nu W^-_\mu - M^2 W^+_\mu W^-_\mu - \tfrac{1}{2}\partial_\nu Z^0_\mu \partial_\nu Z^0_\mu - \tfrac{1}{2c_w^2} M^2 Z^0_\mu Z^0_\mu - \tfrac{1}{2}\partial_\mu A_\nu \partial_\mu A_\nu - igc_w(\partial_\nu Z^0_\mu (W^+_\mu W^-_\nu - W^+_\nu W^-_\mu) - Z^0_\nu (W^+_\mu \partial_\nu W^-_\mu - W^-_\mu \partial_\nu W^+_\mu) + Z^0_\mu (W^+_\nu \partial_\nu W^-_\mu - W^-_\nu \partial_\nu W^+_\mu)) - igs_w(\partial_\nu A_\mu (W^+_\mu W^-_\nu - W^+_\nu W^-_\mu) - A_\nu (W^+_\mu \partial_\nu W^-_\mu - W^-_\mu \partial_\nu W^+_\mu) + A_\mu (W^+_\nu \partial_\nu W^-_\mu - W^-_\nu \partial_\nu W^+_\mu)) - \tfrac{1}{2} g^2 W^+_\mu W^-_\mu W^+_\nu W^-_\nu + \tfrac{1}{2} g^2 W^+_\mu W^-_\nu W^+_\mu W^-_\nu + g^2 c_w^2 (Z^0_\mu W^+_\mu Z^0_\nu W^-_\nu - Z^0_\mu Z^0_\mu W^+_\nu W^-_\nu) + g^2 s_w^2 (A_\mu W^+_\mu A_\nu W^-_\nu - A_\mu A_\mu W^+_\nu W^-_\nu) + g^2 s_w c_w (A_\mu Z^0_\nu (W^+_\mu W^-_\nu - W^+_\nu W^-_\mu) - 2A_\mu Z^0_\nu W^+_\nu W^-_\mu) - \tfrac{1}{2}\partial_\mu H \partial_\mu H - 2M^2 \alpha_h H^2 - \partial_\mu \phi^+ \partial_\mu \phi^- - \tfrac{1}{2}\partial_\mu \phi^0 \partial_\mu \phi^0 - \beta_h \left(\tfrac{2M^2}{g^2} + \tfrac{2M}{g} H + \tfrac{1}{2}(H^2 + \phi^0\phi^0 + 2\phi^+\phi^-)\right) + \tfrac{2M^4}{g^2}\alpha_h - g\alpha_h M (H^3 + H\phi^0\phi^0 + 2H\phi^+\phi^-) - \tfrac{1}{8} g^2 \alpha_h (H^4 + (\phi^0)^4 + 4(\phi^+\phi^-)^2 + 4(\phi^0)^2\phi^+\phi^- + 4H^2\phi^+\phi^- + 2(\phi^0)^2 H^2) - gMW^+_\mu W^-_\mu H - \tfrac{1}{2} g \tfrac{M}{c_w^2} Z^0_\mu Z^0_\mu H - \tfrac{1}{2} ig (W^+_\mu (\phi^0 \partial_\mu \phi^- - \phi^- \partial_\mu \phi^0) - W^-_\mu (\phi^0 \partial_\mu \phi^+ - \phi^+ \partial_\mu \phi^0)) + \tfrac{1}{2} g (W^+_\mu (H \partial_\mu \phi^- - \phi^- \partial_\mu H) + W^-_\mu (H \partial_\mu \phi^+ - \phi^+ \partial_\mu H)) + \tfrac{1}{2} g \tfrac{1}{c_w} (Z^0_\mu (H \partial_\mu \phi^0 - \phi^0 \partial_\mu H) + M (\tfrac{1}{c_w} Z^0_\mu \partial_\mu \phi^0 + W^+_\mu \partial_\mu \phi^- + W^-_\mu \partial_\mu \phi^+) - ig \tfrac{s_w^2}{c_w} M Z^0_\mu (W^+_\mu \phi^- - W^-_\mu \phi^+) + igs_w M A_\mu (W^+_\mu \phi^- - W^-_\mu \phi^+) - ig \tfrac{1-2c_w^2}{2c_w} Z^0_\mu (\phi^+ \partial_\mu \phi^- - \phi^- \partial_\mu \phi^+) + igs_w A_\mu (\phi^+ \partial_\mu \phi^- - \phi^- \partial_\mu \phi^+) - \tfrac{1}{4} g^2 W^+_\mu W^-_\mu (H^2 + (\phi^0)^2 + 2\phi^+\phi^-) - \tfrac{1}{8} g^2 \tfrac{1}{c_w^2} Z^0_\mu Z^0_\mu (H^2 + (\phi^0)^2 + 2(2s_w^2 - 1)^2 \phi^+\phi^-) - \tfrac{1}{2} g^2 \tfrac{s_w^2}{c_w} Z^0_\mu \phi^0 (W^+_\mu \phi^- + W^-_\mu \phi^+) - \tfrac{1}{2} ig^2 \tfrac{s_w^2}{c_w} Z^0_\mu H (W^+_\mu \phi^- - W^-_\mu \phi^+) + \tfrac{1}{2} g^2 s_w A_\mu \phi^0 (W^+_\mu \phi^- +$$

Exercise: How long is the feature vector in each case?

Perceptron with nonlinear features



Nonlinear feature map

$$x_i \in \mathbb{R}^p \mapsto \phi(x_i) \in \mathbb{R}^P$$

Algorithm:

1. Initialize parameters; set iteration counter t = 1.
2. Cycle through training data $(x_1, y_1), \ldots, (x_N, y_N)$ and update

$$\left.\begin{array}{l} \text{if } y_i \neq h(x_i; w^t, w_0^t), \text{ then} \\ w^{t+1} = w^t + y_i \phi(x_i) \end{array}\right\} \; y_i(\phi(x_i)^T w^t + w_0^t) \leq 0$$

$$w_0^{t+1} = w_0^t + y_i$$

The computational burden

If ‘x’ has 1000 dimensions, quadratic features will have ~ 500K features!

Exercise: How many dimensions will polynomial features of degree 20 have?

Answer: Too many!

Corollary: By the theory of lectures, there must be a way around this!

Perceptron with nonlinear features

Perceptron loop

$$\text{if } y_i(\langle w^t, \phi(x_i)\rangle + w_0^t) \leq 0, \text{ then}$$
$$w^{t+1} = w^t + y_i \phi(x_i)$$
$$w_0^{t+1} = w_0^t + y_i$$

$$\langle x, z\rangle = x \cdot z = x^T z = \sum_j x_j z_j$$

Notation

Final weight vector (linear combination of training data)

$$w = \sum_{i \in I} y_i \phi(x_i)$$

$i \in I$ ← index (multi)set of points for which updates where made

Decision function

$$h(x) = \text{sgn}\left(\langle w, \phi(x)\rangle + w_0\right)$$

where $$\langle w, \phi(x)\rangle = \sum_{i \in I} y_i \langle \phi(x_i), \phi(x)\rangle$$

SVMs with nonlinear features

C-SVM (soft-margin SVM with slacks)

$$\min_{w,w_0,\xi} \quad \tfrac{1}{2}\|w\|^2 + C\sum\nolimits_i \xi_i$$

$$y_i(\langle w, \phi(x_i)\rangle + w_0) \geqslant 1 - \xi_i, \quad 1 \leqslant i \leqslant N$$

$$\xi_i \geq 0, \quad 1 \leqslant i \leqslant N$$

Dual of C-SVM

$$\max_{\alpha} \quad -\tfrac{1}{2}\left\|\sum\nolimits_i \alpha_i y_i \phi(x_i)\right\|^2 + \sum\nolimits_i \alpha_i$$

$$\sum\nolimits_i y_i \alpha_i = 0$$

$$0 \leq \alpha \leq C.$$

$w = \sum_i \alpha_i y_i \phi(x_i)$ Recall, this is what an SVM solution looks like

Decision function

$$h(x) = \operatorname{sgn}\left(\langle w, \phi(x)\rangle + w_0\right) \qquad \langle w, \phi(x)\rangle = \sum\nolimits_i \alpha_i y_i \langle \phi(x_i), \phi(x)\rangle$$

The key idea

The key idea

Decision depends on just inner-products

This is where the idea of kernel functions comes in!

Suppose we had access to a function k(x, x') that could compute

$$k(x, x') = \langle \phi(x), \phi(x') \rangle$$

without ever actually constructing the nonlinear features $\phi(x)$, $\phi(x')$!

If k(x,x') is much cheaper to compute than ϕ, then we're in business

A **kernel function** $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a symmetric function that can be written as

$$k(x, x') = \langle \phi(x), \phi(x') \rangle$$

for some *feature map* ϕ

Kernel perceptron, kernel SVM

Decision function

$$h(x) = \text{sgn}\left(\langle w, \phi(x)\rangle + w_0\right)$$

Kernel perceptron

$$\langle w, \phi(x)\rangle = \sum_{i \in I} y_i k(x_i, x)$$

Kernel SVM

$$\langle w, \phi(x)\rangle = \sum\nolimits_i \alpha_i y_i k(x_i, x)$$

Crucial practical point

For neither the perceptron nor the SVM do we need to construct explicit features at any point, yet we can compute w. How's that?

This idea applies to **any method** that has $w = \sum\nolimits_i a_i \phi(x_i)$

Exercise: Show how to obtain kernelized ridge regression

Constructing kernels

- Linear kernel $k(x, x') = \langle x, x' \rangle$
- Scaling by a positive number $k(x, x') \to ck(x, x'), \ c \geq 0$
- Sum of two kernel functions $k_1(x, x') + k_2(x, x')$
- Product of two kernel functions $k_1(x, x')k_2(x, x')$

Let's call this the algebra of kernels

Example: the simple polynomial kernel $\langle x, x' \rangle^d$ (product rule)

Exercise: If k(x,x') is a kernel then $k(x, x') \mapsto e^{k(x,x')}$ is also a kernel

Exercise: Prove that $k(x, x') = (c + \langle x, x' \rangle)^d, \ c \geq 0, \ d \in \mathbb{N}$ is a kernel

Computational appreciation

Imagine the simple quadratic kernel $\langle x, x' \rangle^2$

[blackboard]

Exercise: How much is the computational savings for x, x' being p-dimensional?

SVM dual

$$\max_{\alpha} \quad -\tfrac{1}{2} \left\| \sum_i \alpha_i y_i \phi(x_i) \right\|^2 + \sum_i \alpha_i$$

$$\sum_i y_i \alpha_i = 0$$

$$0 \le \alpha \le C.$$

$$\Rightarrow \quad \sum_i \sum_j \alpha_i \alpha_j y_i y_j \langle \phi(x_i), \phi(x_j) \rangle$$

$$\Downarrow$$

$$\sum_i \sum_j \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

Kernel matrix: $K = [k(x_i, x_j)]$ (this is of size N x N, for $1 \le i, j \le N$)

Kernel Conditions

Computability

We have to be able to compute $k(x,x')$ efficiently, much cheaper than the explicit dot products

"Nice and useful" features and functions

The features must be useful for the learning problem. Often this means smooth kernel functions, i.e., $k(x, \bullet)$ is a smooth function.

Symmetry

Clearly $k(x,x') = k(x',x)$ because dot products are symmetric:

$$\langle \phi(x), \phi(x') \rangle = \langle \phi(x'), \phi(x) \rangle$$

Positive definiteness

Aka existence of dot products in feature space, i.e., can does there exist a map f such that $k(x,x') = \langle \phi(x), \phi(x') \rangle$

Mercer's Theorem

The Theorem

For any symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ which is square integrable in $\mathcal{X} \times \mathcal{X}$ and which satisfies

$$\int_{\mathcal{X}\times\mathcal{X}} k(x, x')f(x)f(x')dxdx' \geq 0 \text{ for all } f \in L_2(\mathcal{X})$$

there exist $\phi_i : \mathcal{X} \to \mathbb{R}$ and numbers $\lambda_i \geq 0$ where

$$k(x, x') = \sum_i \lambda_i \phi_i(x)\phi_i(x') \text{ for all } x, x' \in \mathcal{X}.$$

Interpretation

Double integral is the continuous version of a vector-matrix-vector multiplication. For positive semidefinite matrices we have

$$\sum\sum k(x_i, x_j)\alpha_i\alpha_j \geq 0$$

Kernel matrix (i,j)-element

Kernel properties

Distance in feature space

$$\|\phi(x) - \phi(x')\|^2 = \|\phi(x)\|^2 + \|\phi(x')\|^2 - 2\langle\phi(x), \phi(x')\rangle$$
$$= k(x,x) + k(x',x') - 2k(x,x')$$

Kernel matrix

To study observations, we make kernel matrix *K* with entries given by

$$K_{ij} = \langle\phi(x_i), \phi(x_j)\rangle = k(x_i, x_j)$$

Similarity measure

The kernel function k describes similarity. We can view K_{ij} capturing how similar point x_i is to x_j

General applicability

The input data x, x' can come from any space; we can use kernel methods on them as soon as we manage to define a valid kernel function on this space

Feature map

Is not unique; several different ϕ can generate same k(x,x')

Examples

Examples of kernels $k(x, x')$

Linear	$\langle x, x' \rangle$
Laplacian RBF	$\exp\left(-\lambda \|x - x'\|\right)$
Gaussian RBF	$\exp\left(-\lambda \|x - x'\|^2\right)$
Polynomial	$\left(\langle x, x' \rangle + c\rangle\right)^d, c \geq 0,\ d \in \mathbb{N}$
B-Spline	$B_{2n+1}(x - x')$
Cond. Expectation	$\mathbf{E}_c[p(x\|c)p(x'\|c)]$

Simple trick for checking Mercer's condition

Compute the Fourier transform of the kernel and check that it is nonnegative.

More fancy examples

Probability kernel

Let A, B be events drawn from a sample space Ω with distribution P. Then,

$$k(A, B) = P(A \cap B) - P(A)P(B)$$

Exercise: Show that this is a kernel

String kernel

Let x and x' be two strings of length d and d' on alphabet A. Then

$$k(x, x') = \sum_{S \in A^*} w_S [\![S \in x]\!] [\![S \in x']\!]$$

Notice this is an explicit feature map

$\phi_S(x)$ a 1-hot encoding over A*

A*: set of all finite strings over alphabet A

Graph kernels, image kernels, molecule kernels, kernels over kernels, you name it!

A Counterexample

A Candidate for a Kernel

$$k(x, x') = \begin{cases} 1 & \text{if } \|x - x'\| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

This is symmetric and gives us some information about the proximity of points, yet it is not a proper kernel . . .

Kernel Matrix

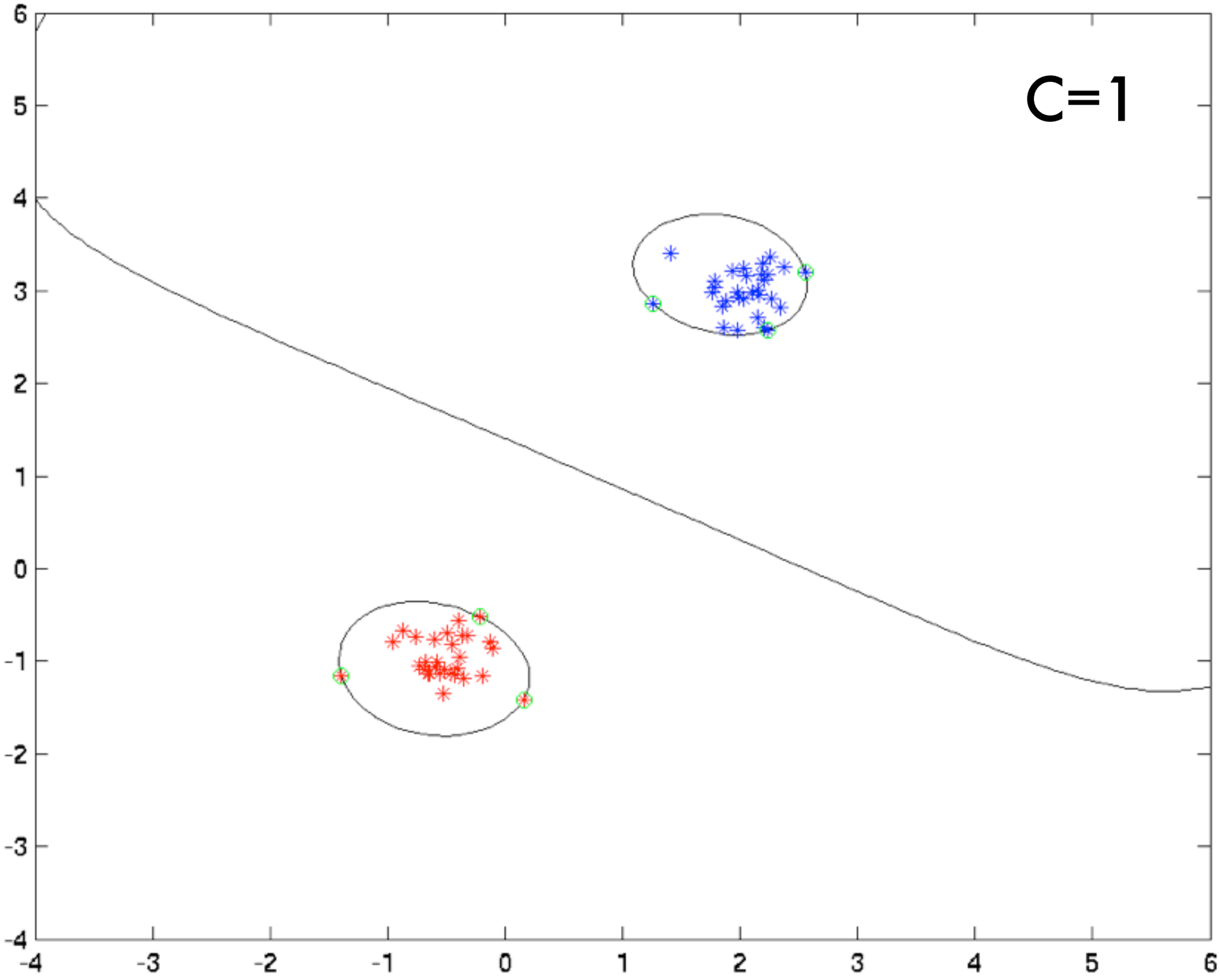
We use three points, $x_1 = 1, x_2 = 2, x_3 = 3$ and compute the resulting "kernelmatrix" K. This yields

$K = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$ and eigenvalues $(\sqrt{2}-1)^{-1}, 1$ and $(1-\sqrt{2})$.

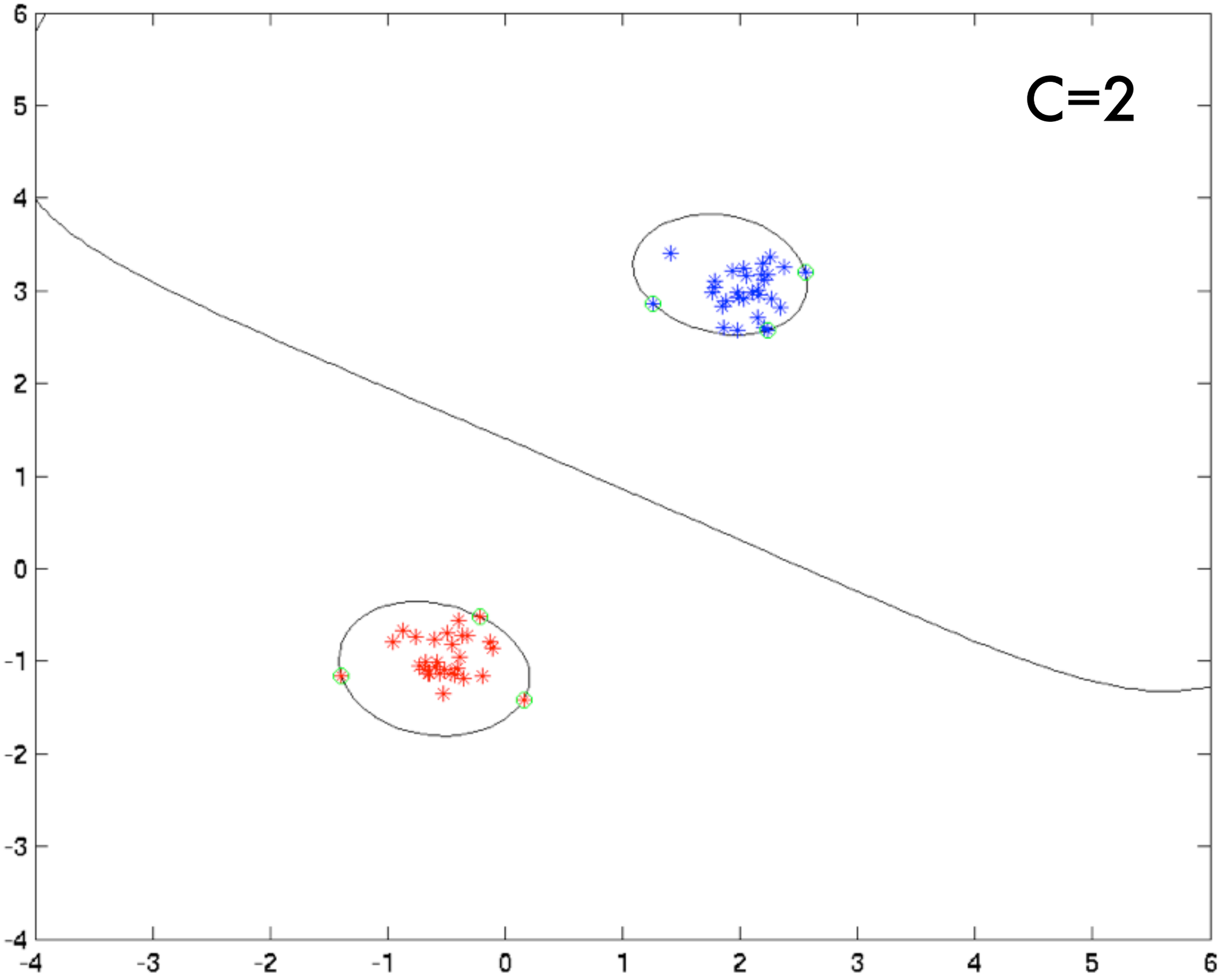
as eigensystem. Hence k is not a kernel.

SVM with a polynomial Kernel visualization

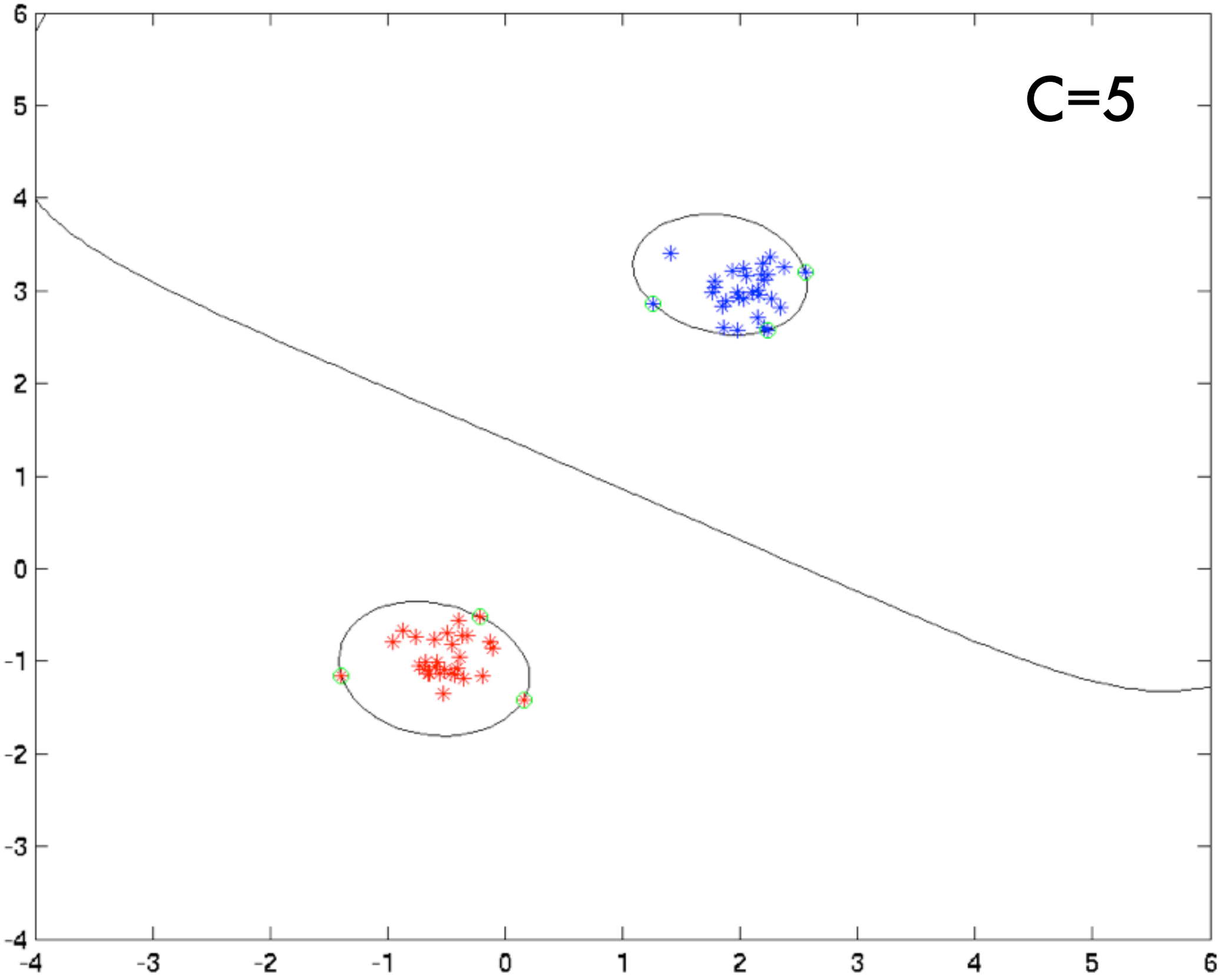
Created by:
Udi Aharoni



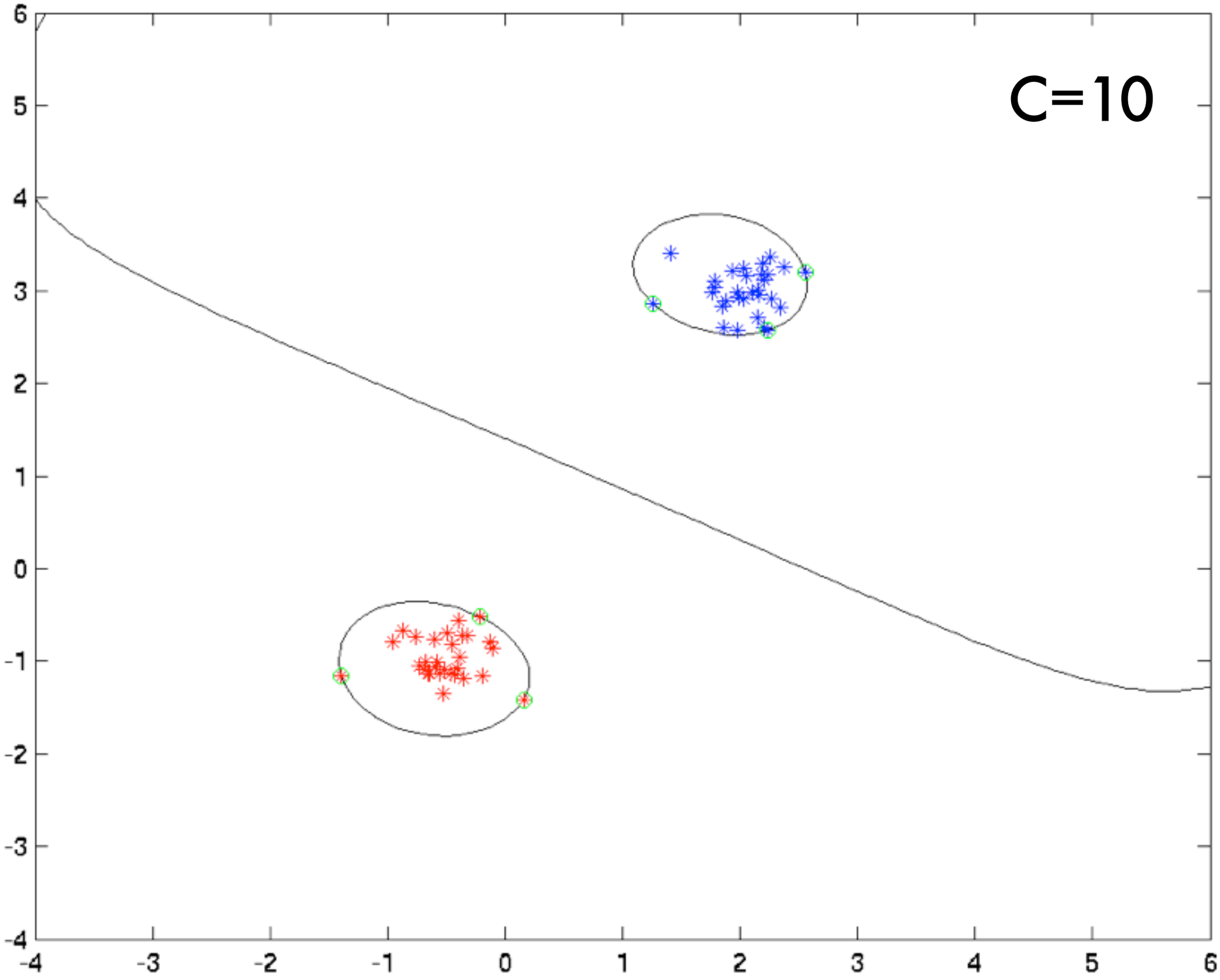
[thanks to: Alex Smola]



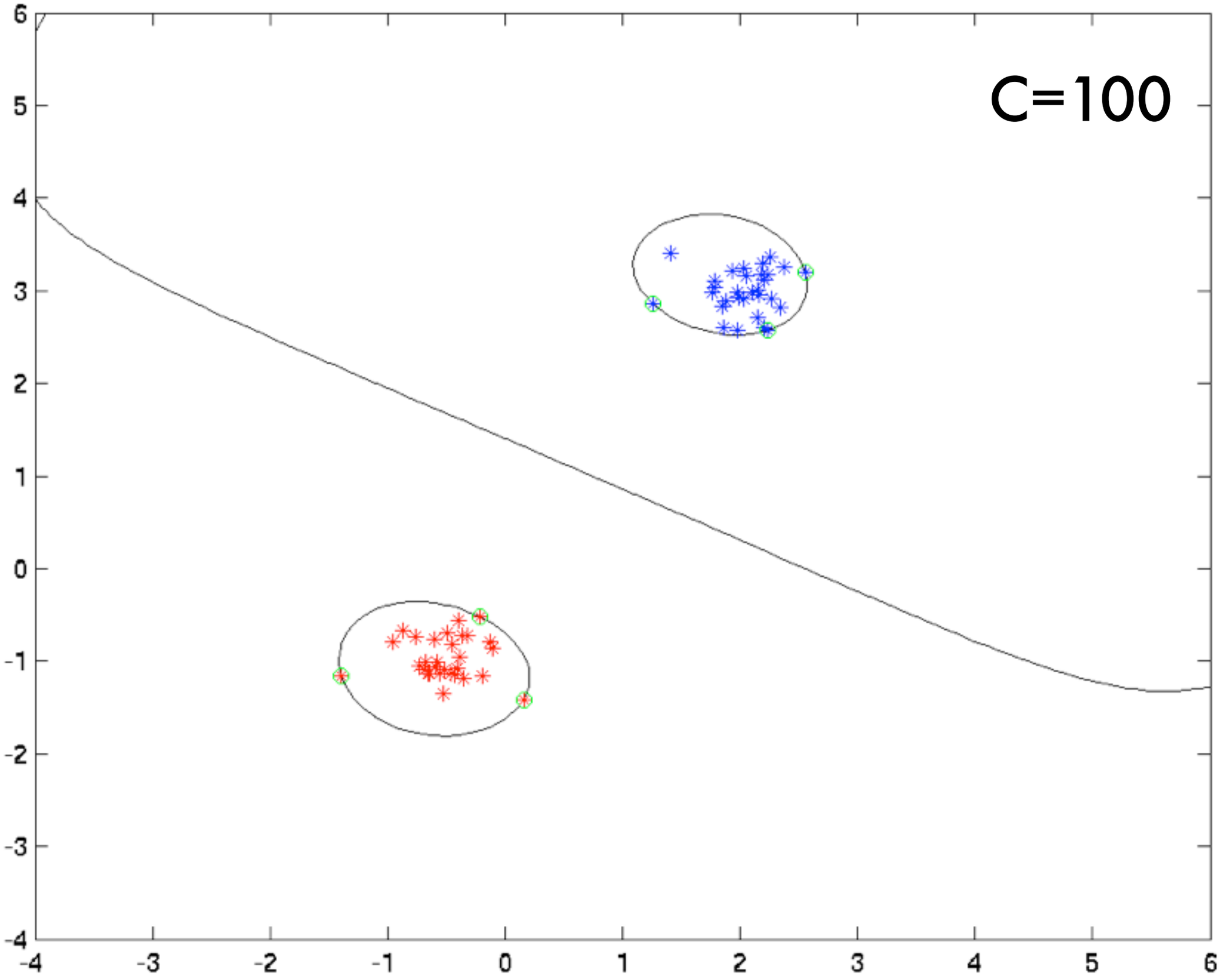
[thanks to: Alex Smola]



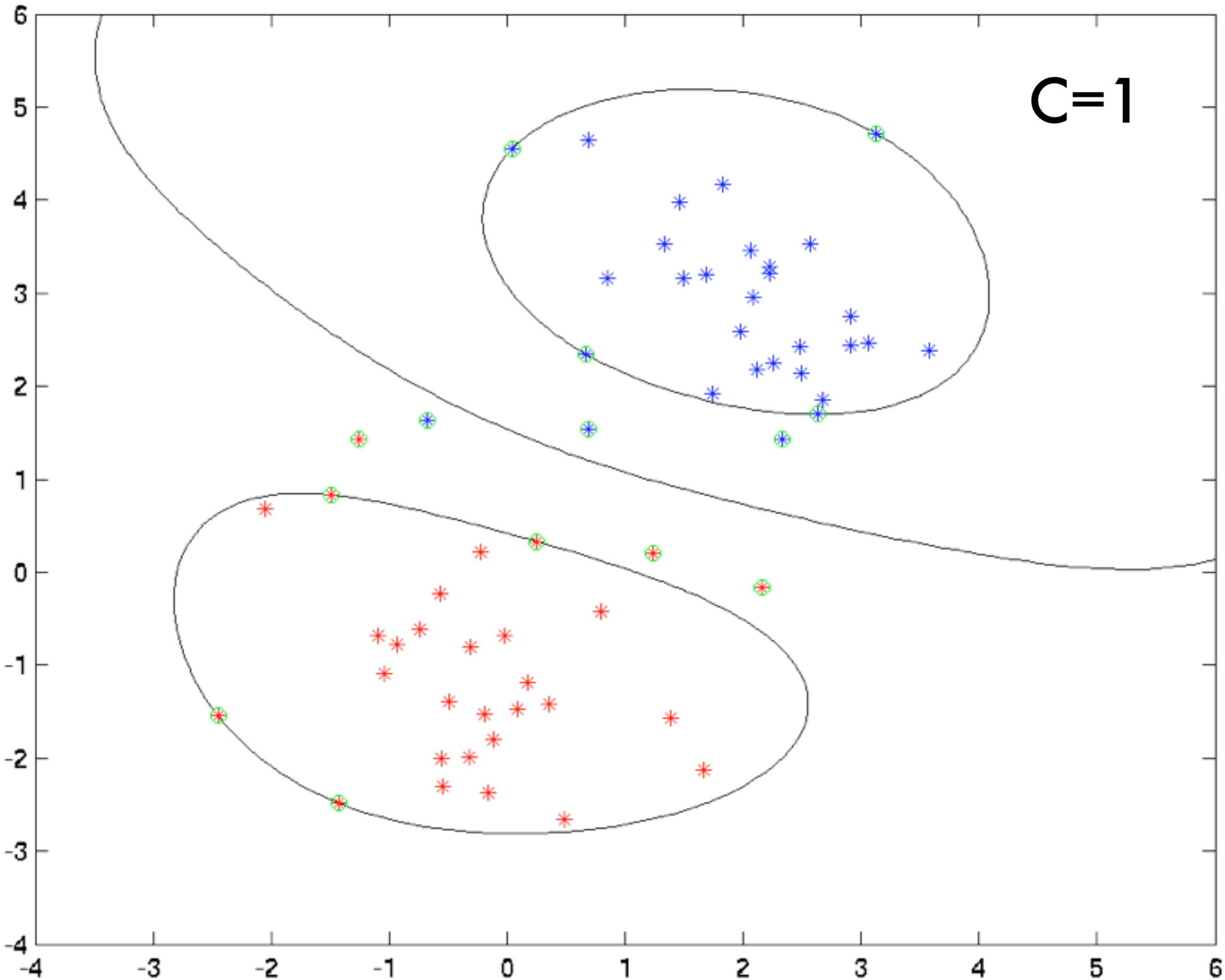
[thanks to: Alex Smola]



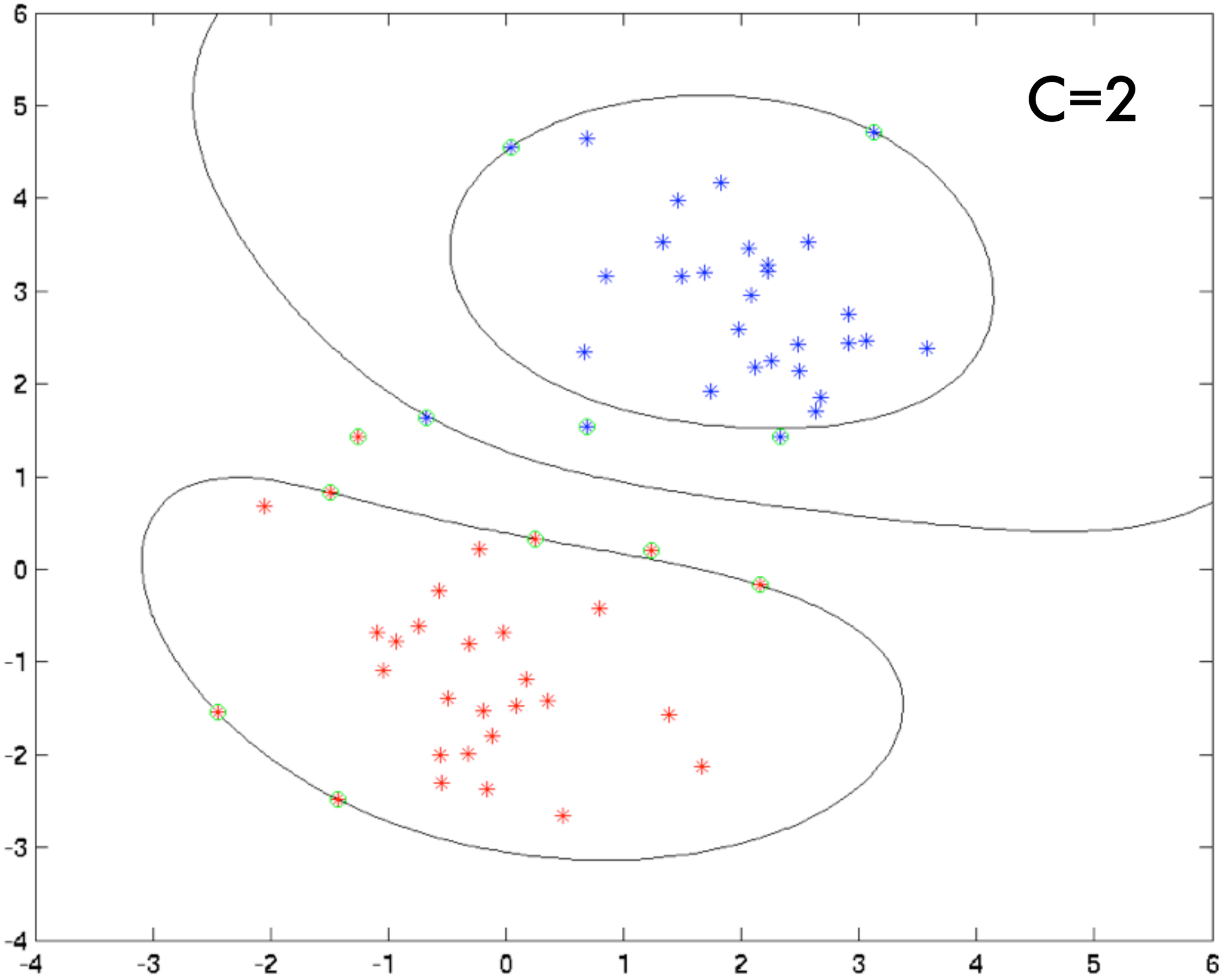
[thanks to: Alex Smola]



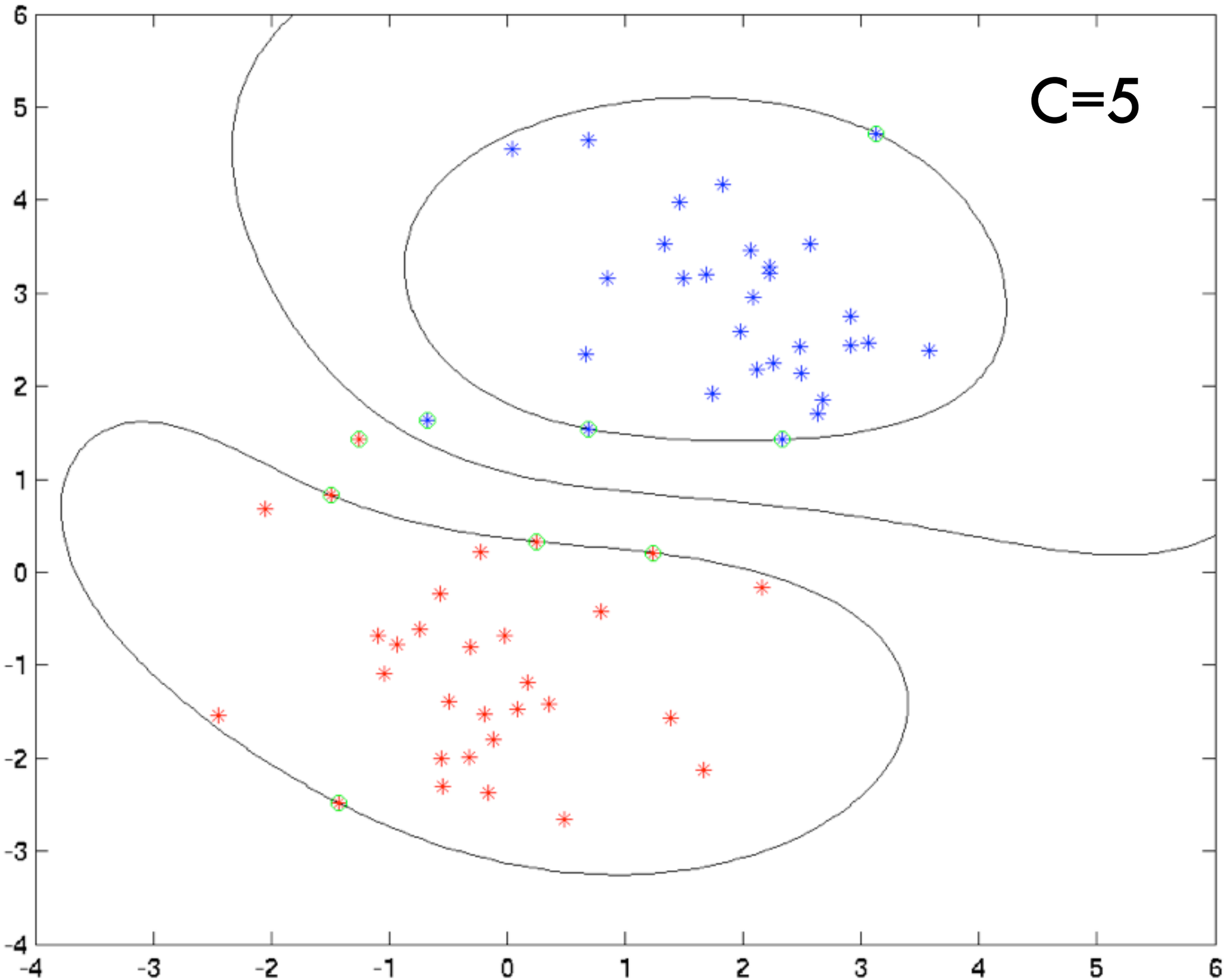
[thanks to: Alex Smola]



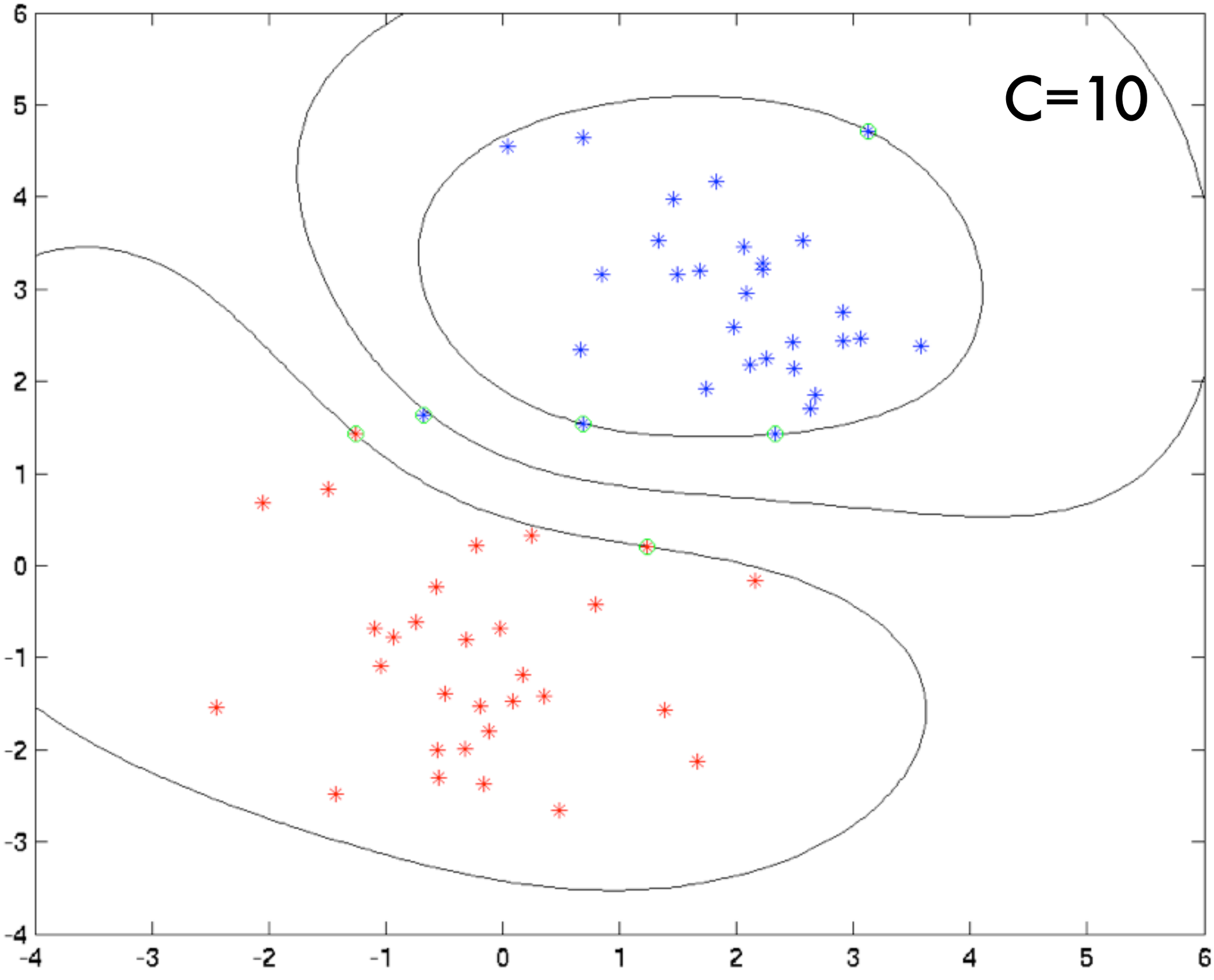
[thanks to: Alex Smola]



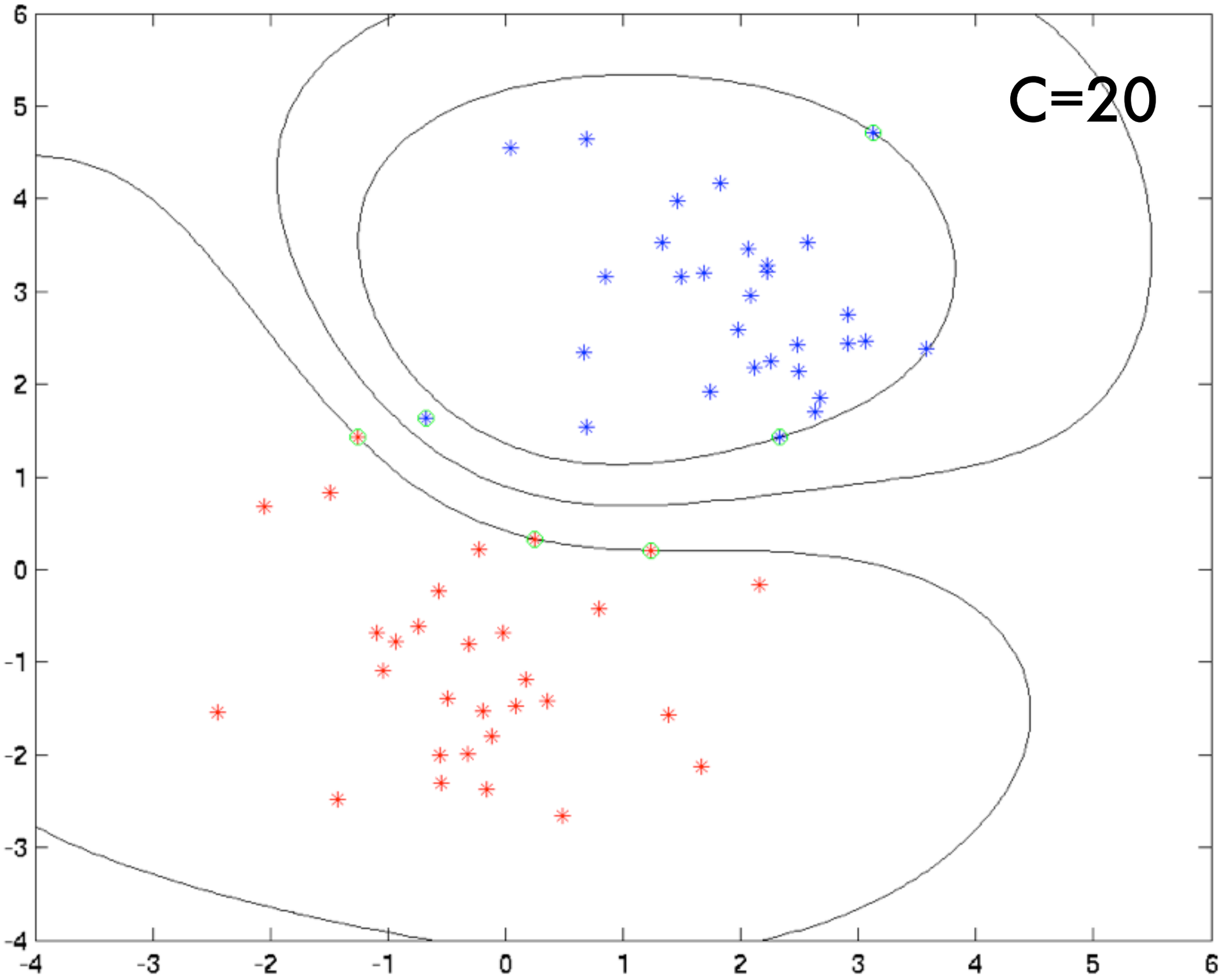
[thanks to: Alex Smola]



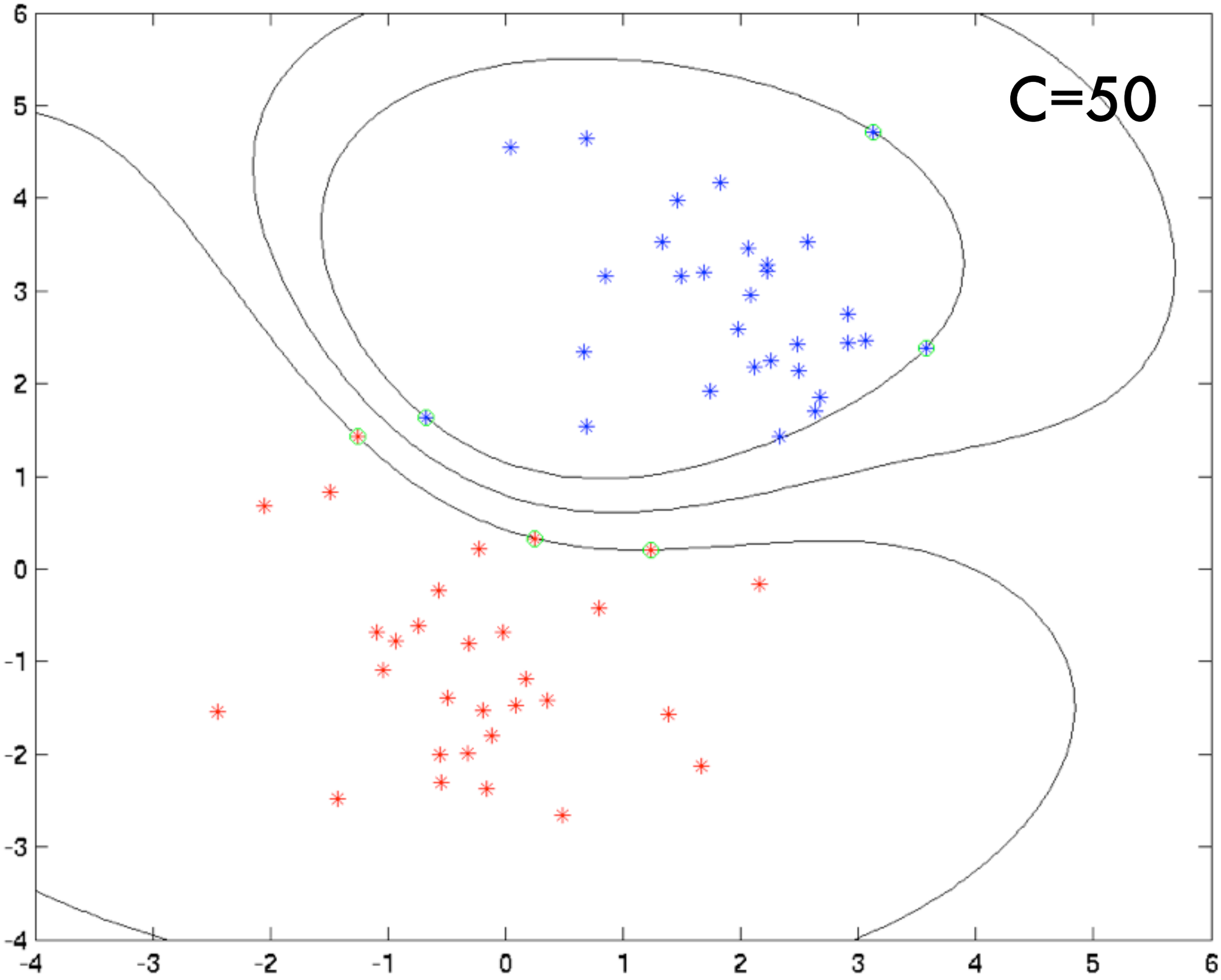
[thanks to: Alex Smola]



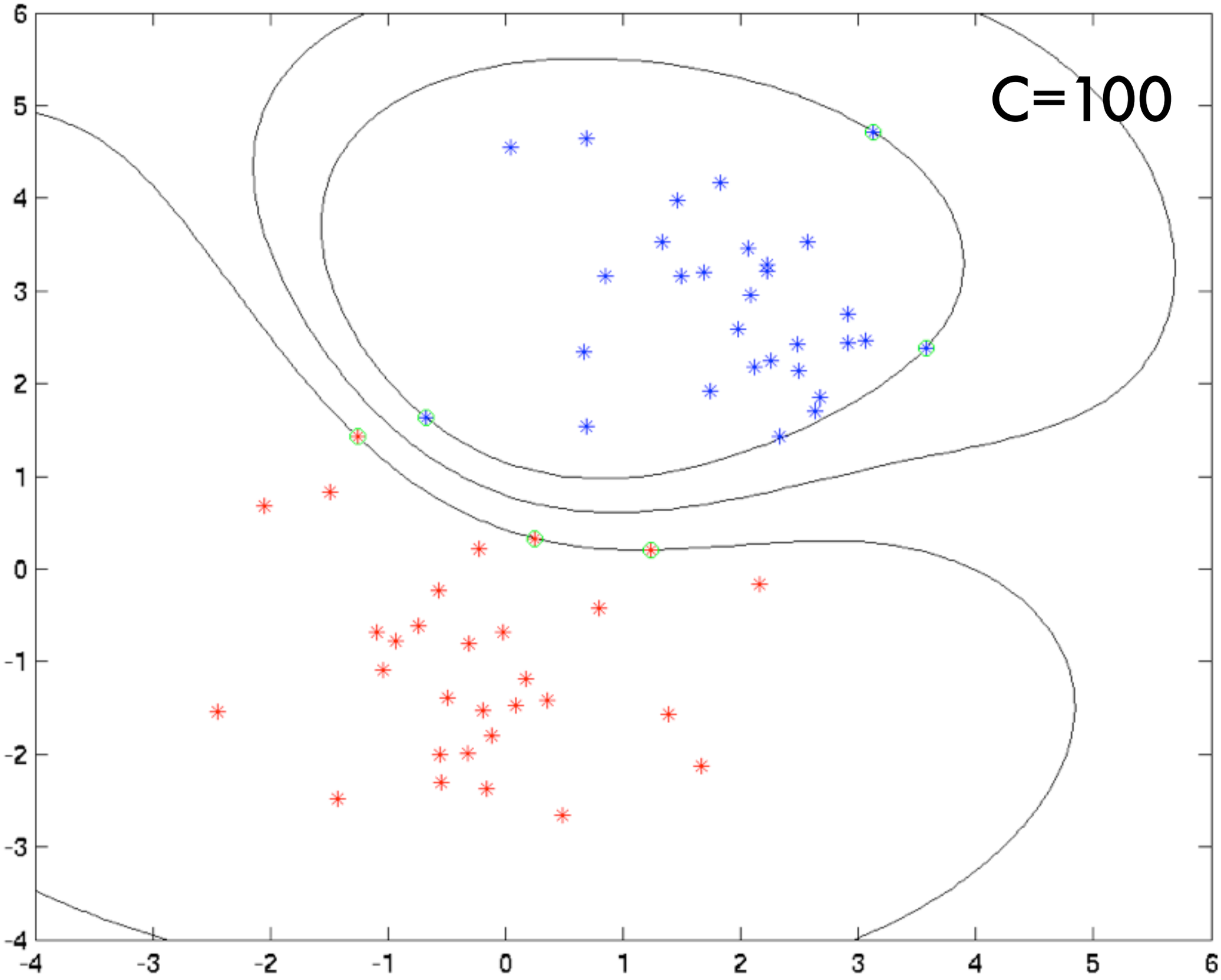
[thanks to: Alex Smola]



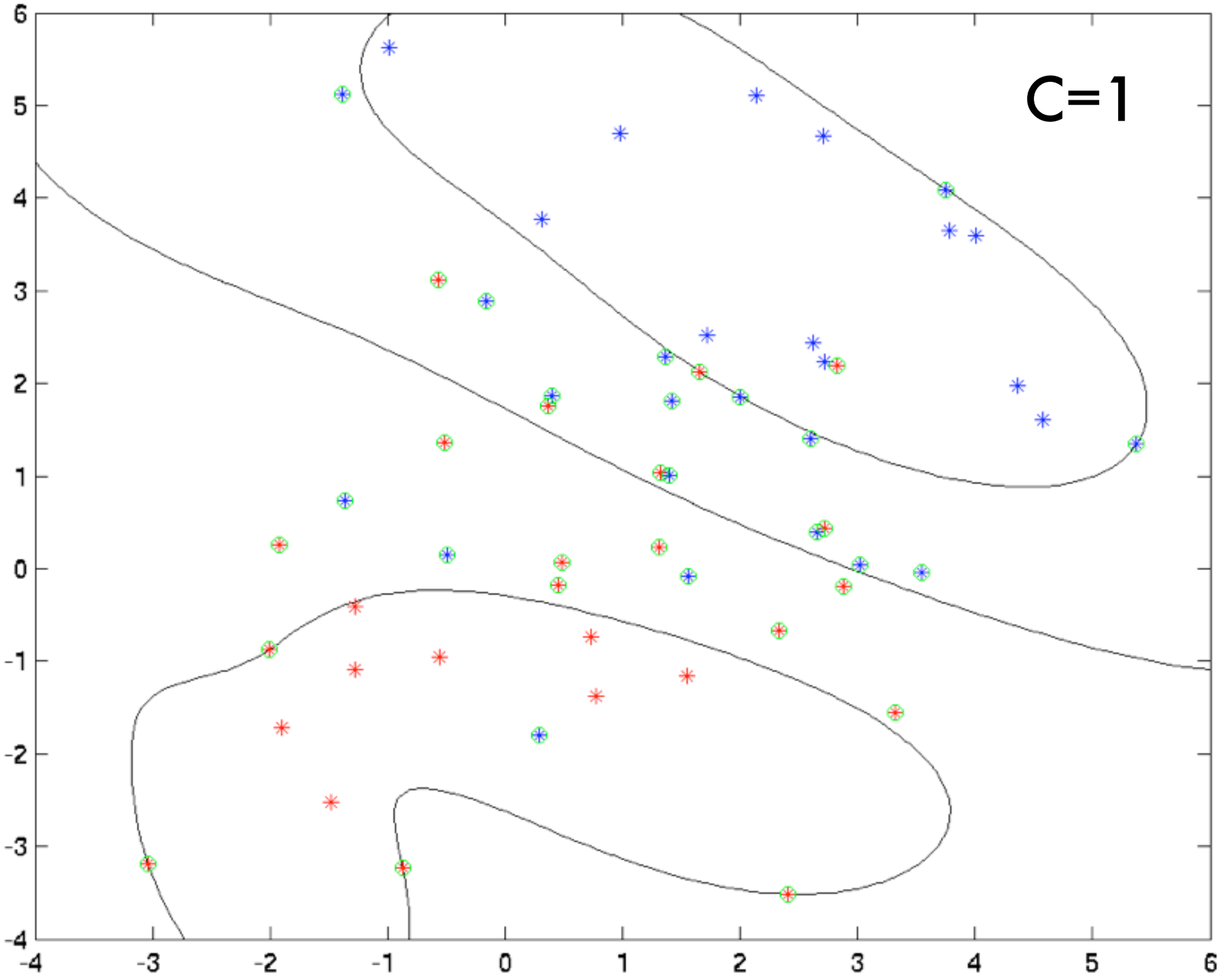
[thanks to: Alex Smola]



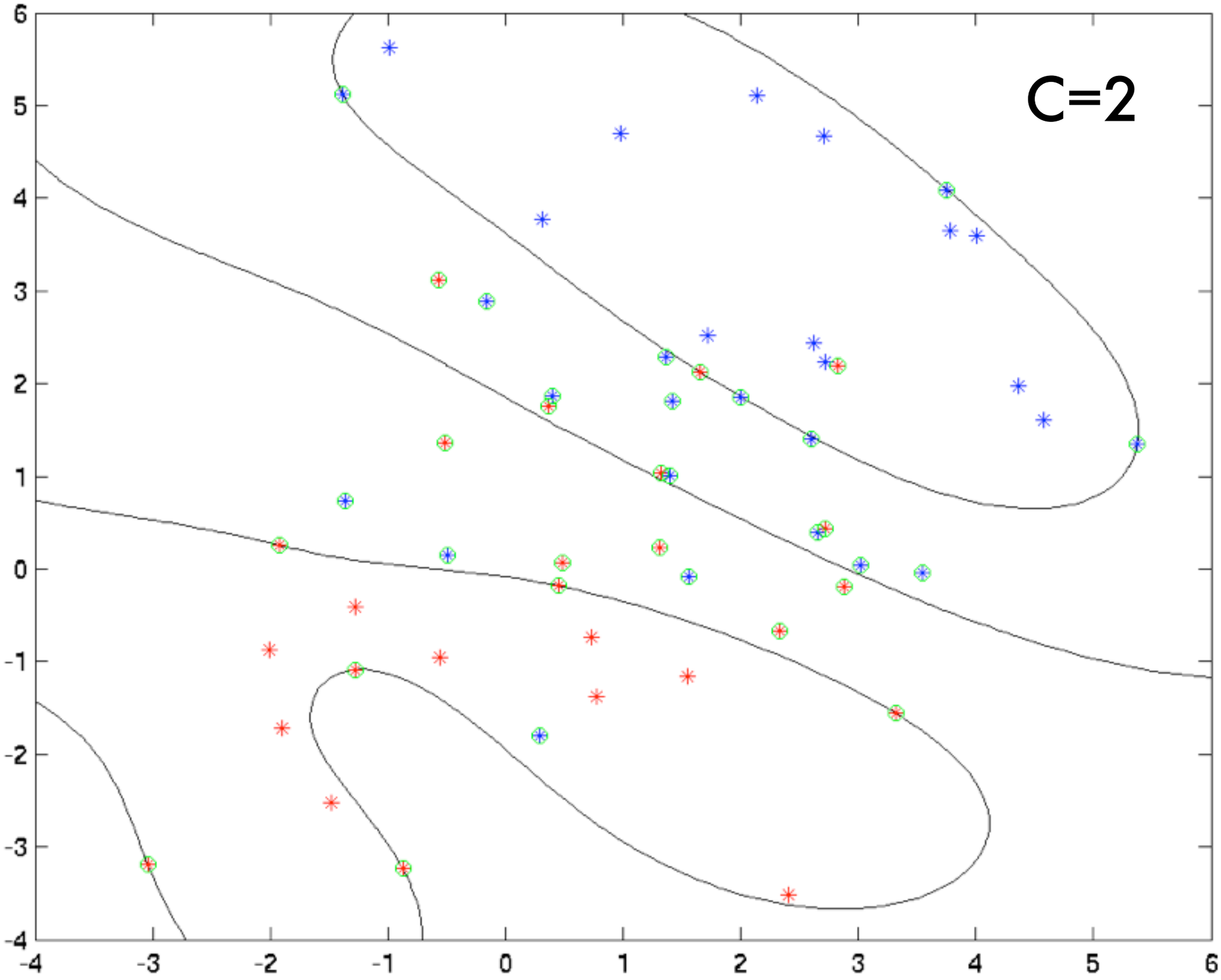
[thanks to: Alex Smola]



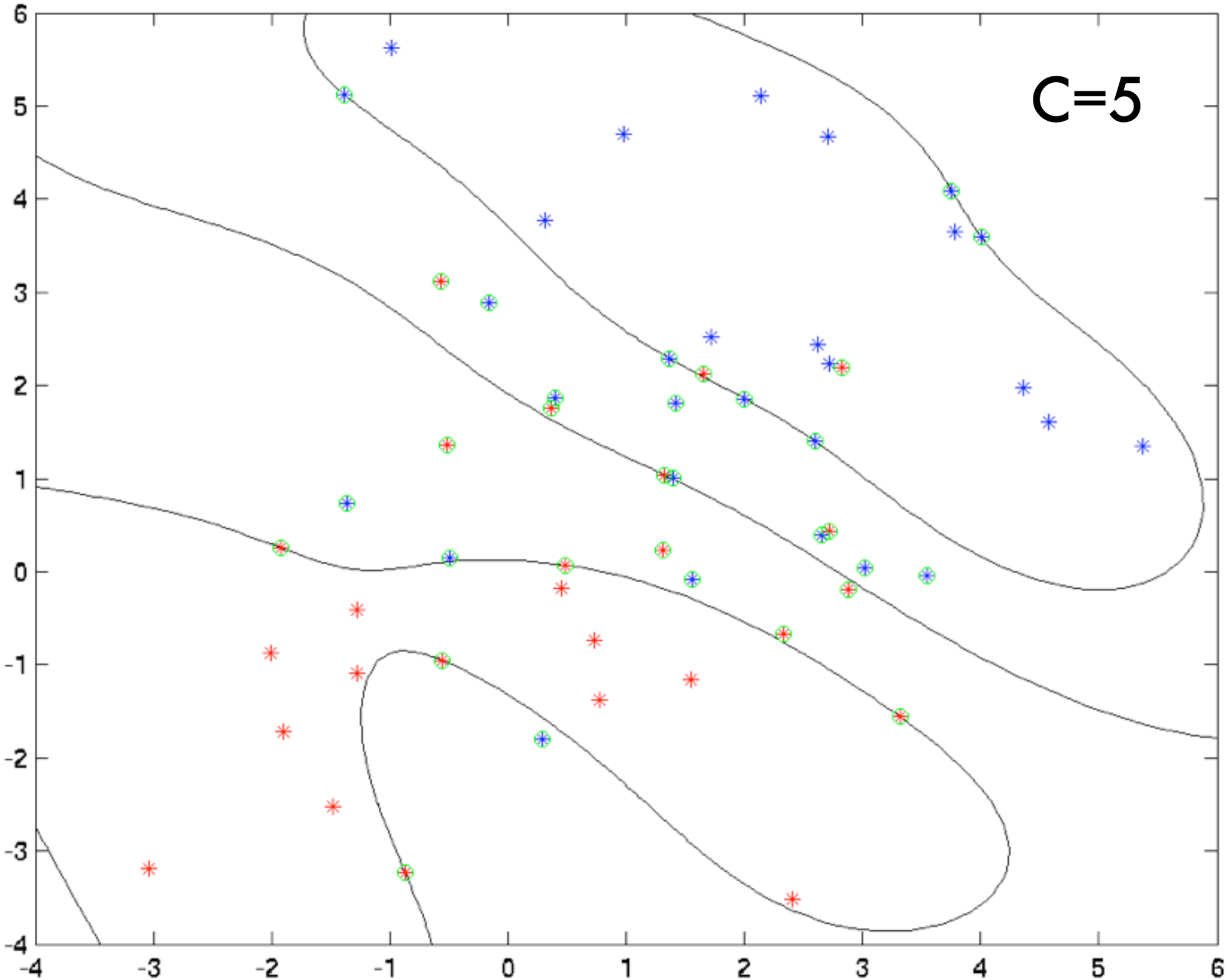
[thanks to: Alex Smola]



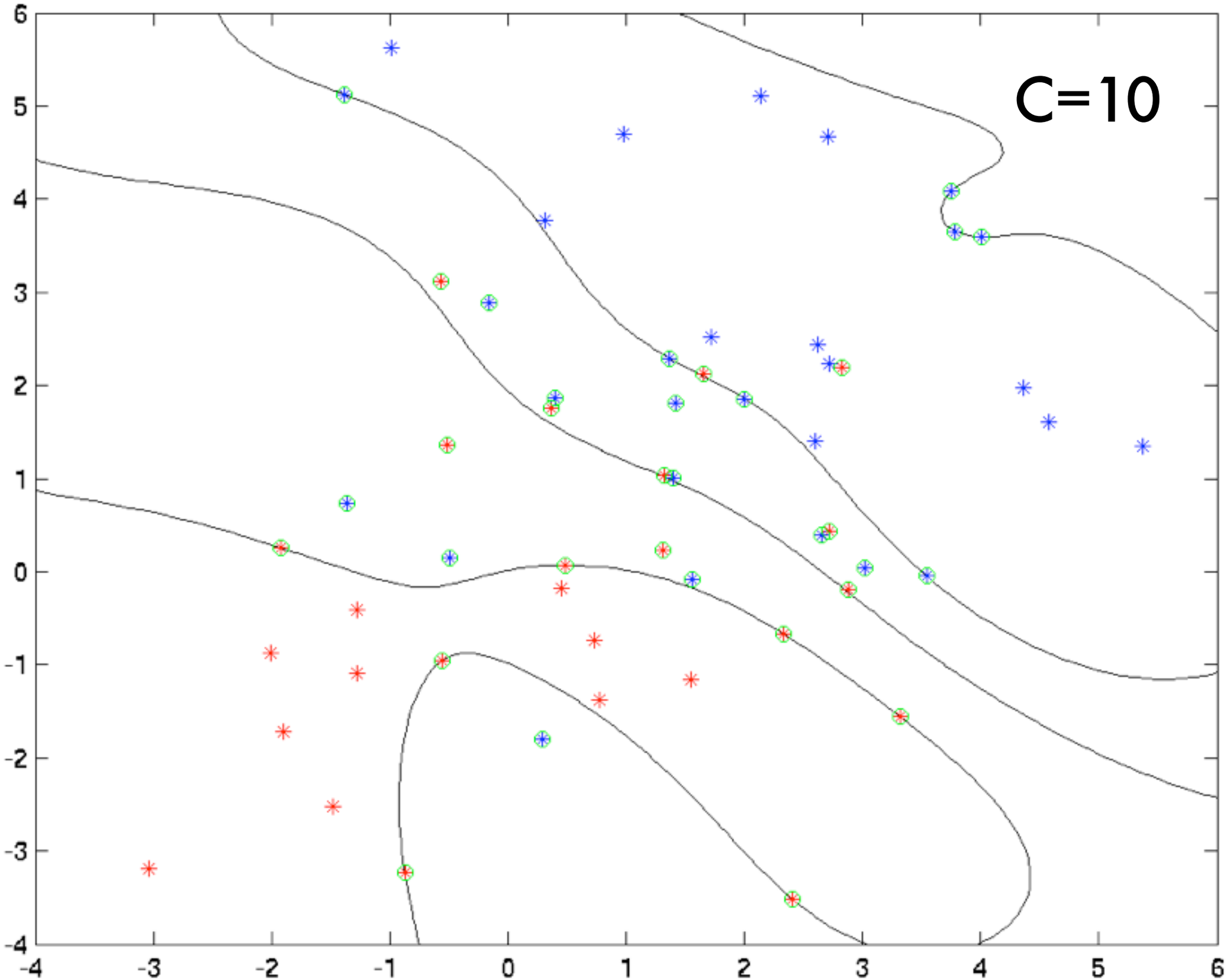
[thanks to: Alex Smola]



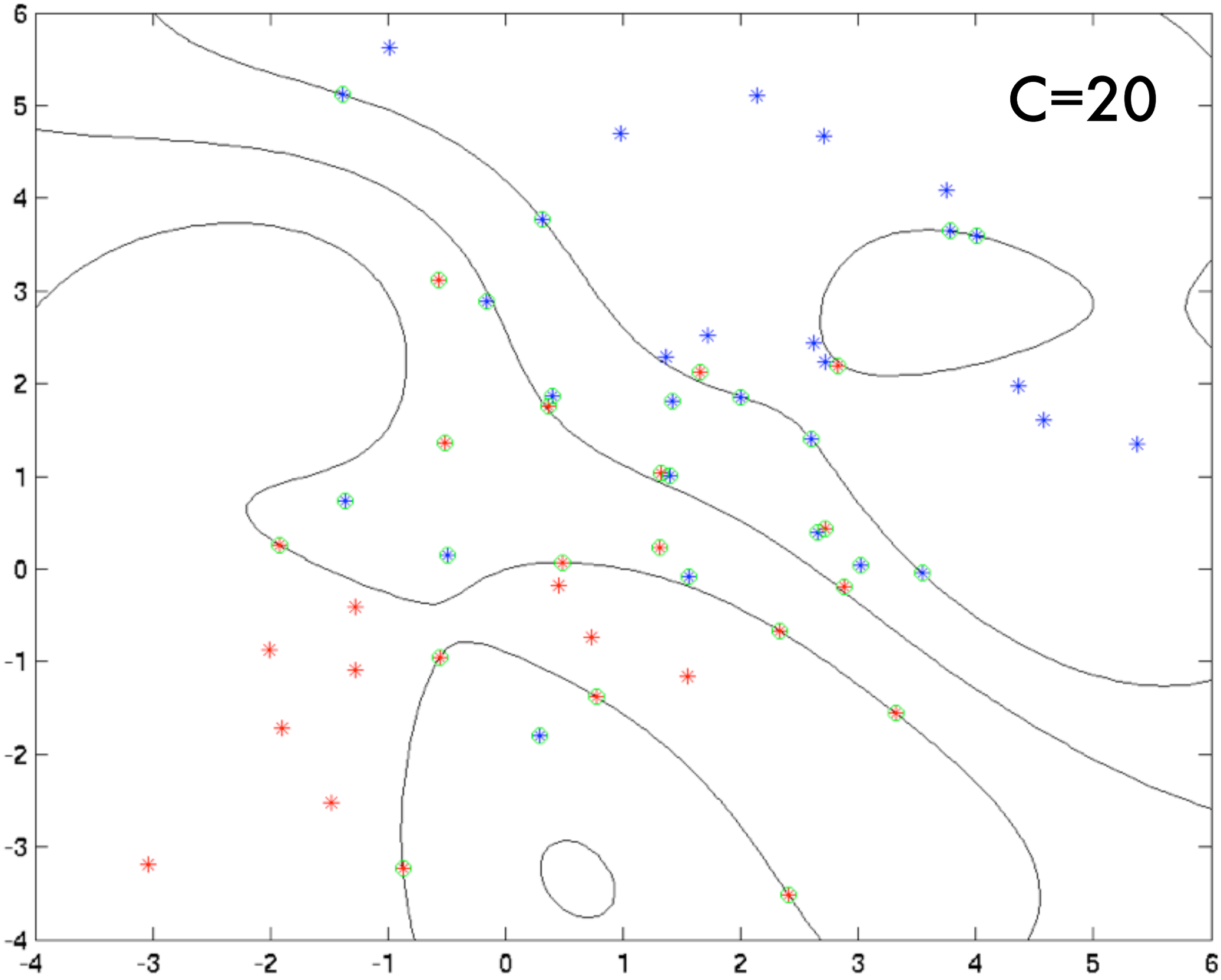
[thanks to: Alex Smola]



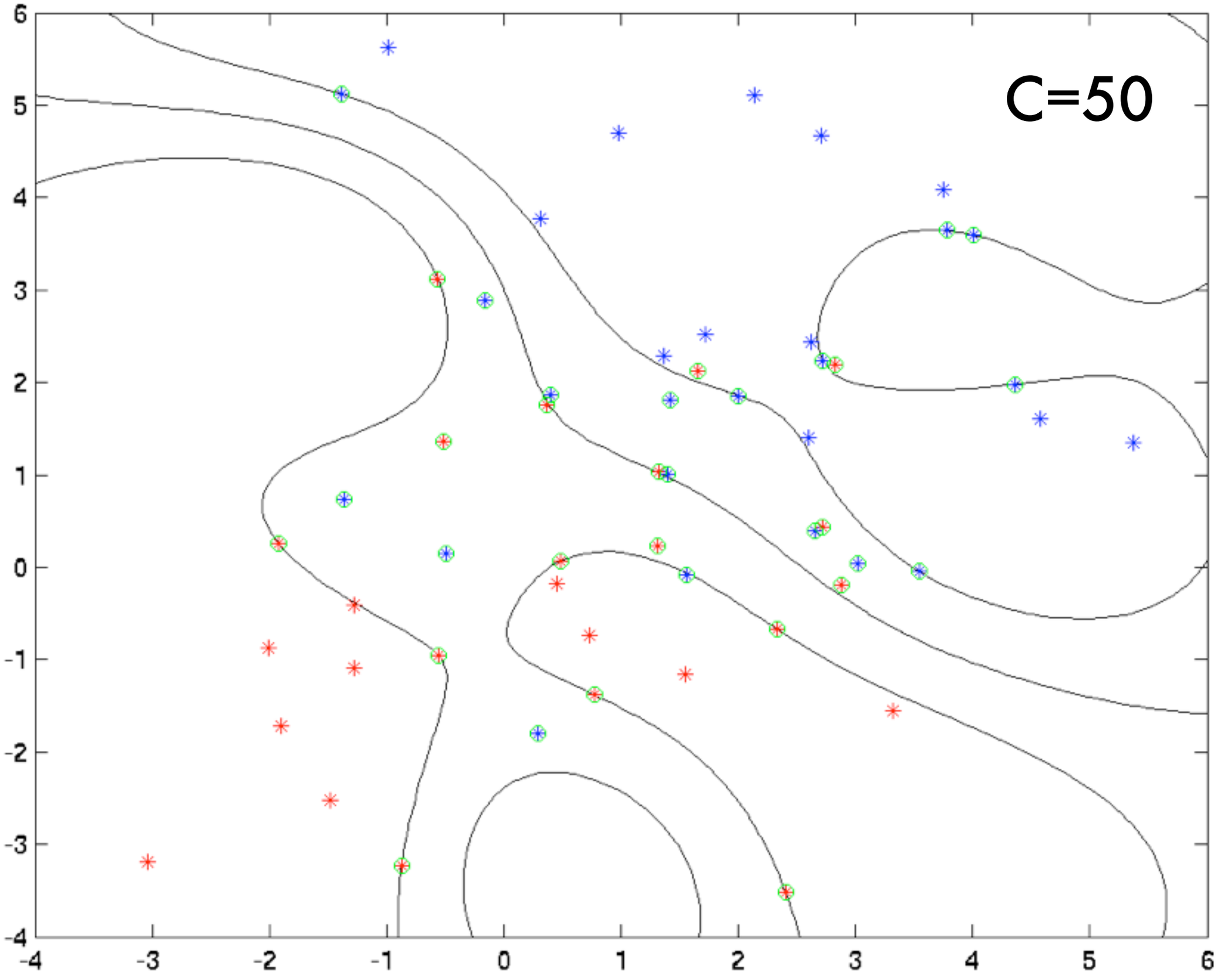
[thanks to: Alex Smola]



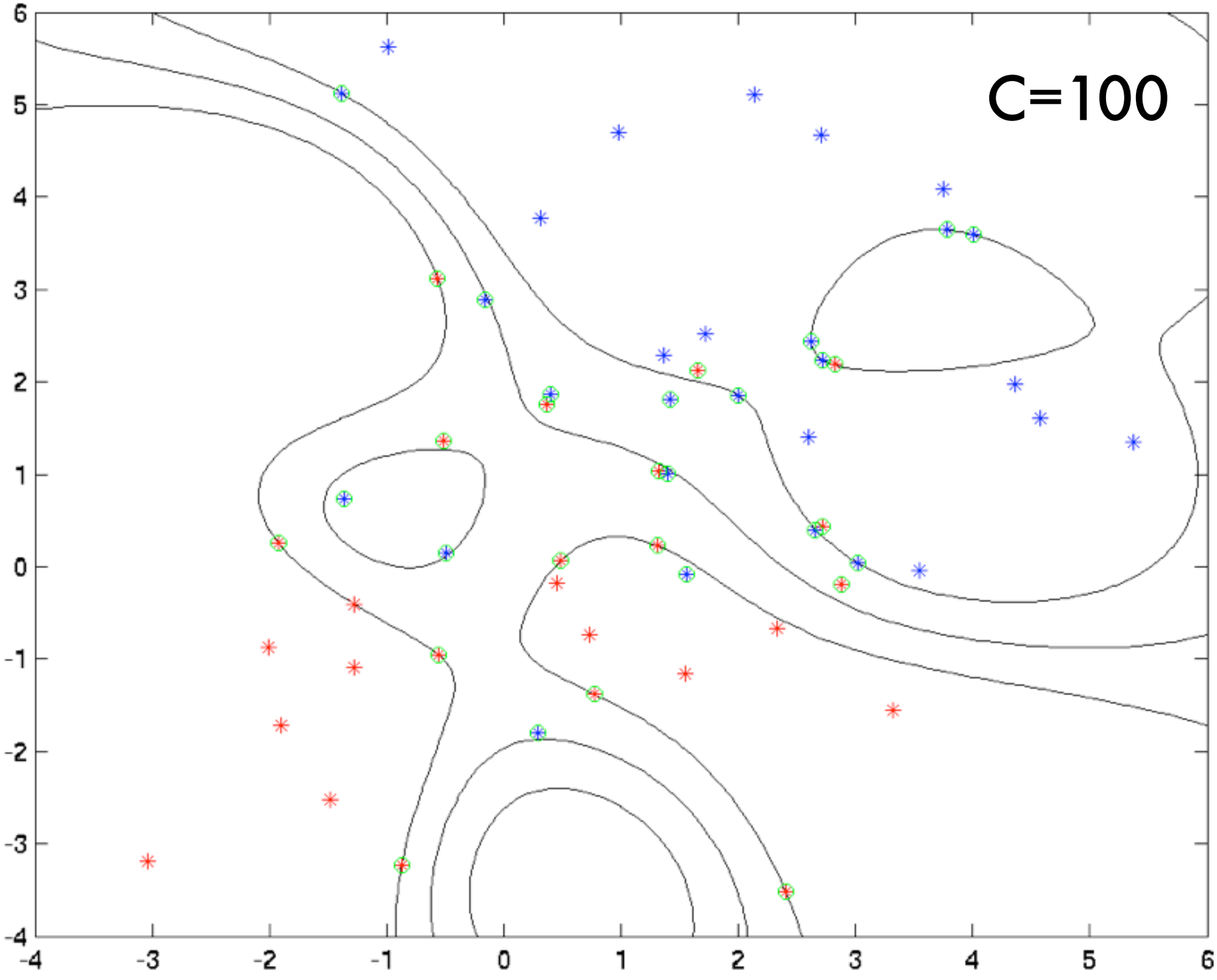
[thanks to: Alex Smola]



[thanks to: Alex Smola]



[thanks to: Alex Smola]

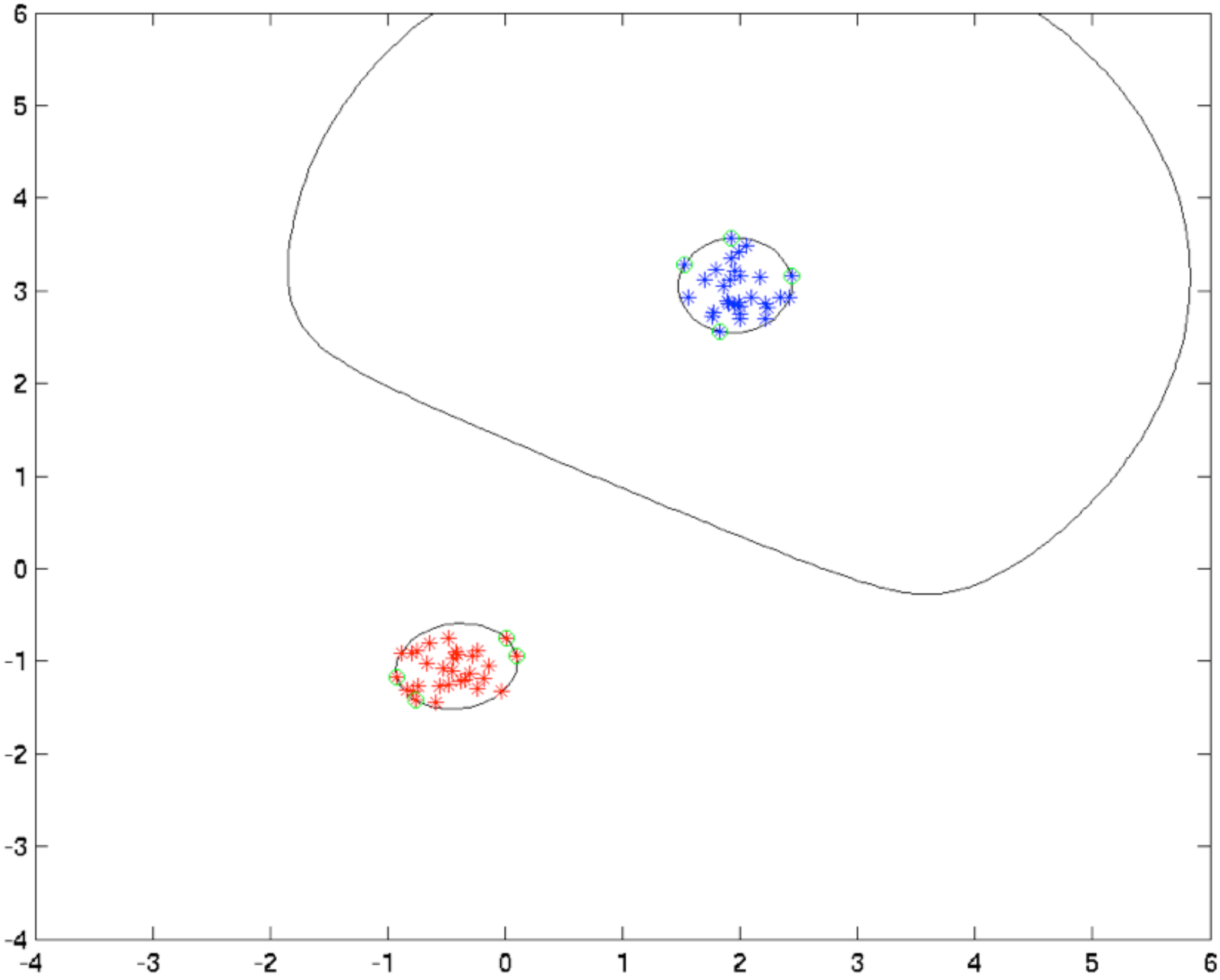


[thanks to: Alex Smola]

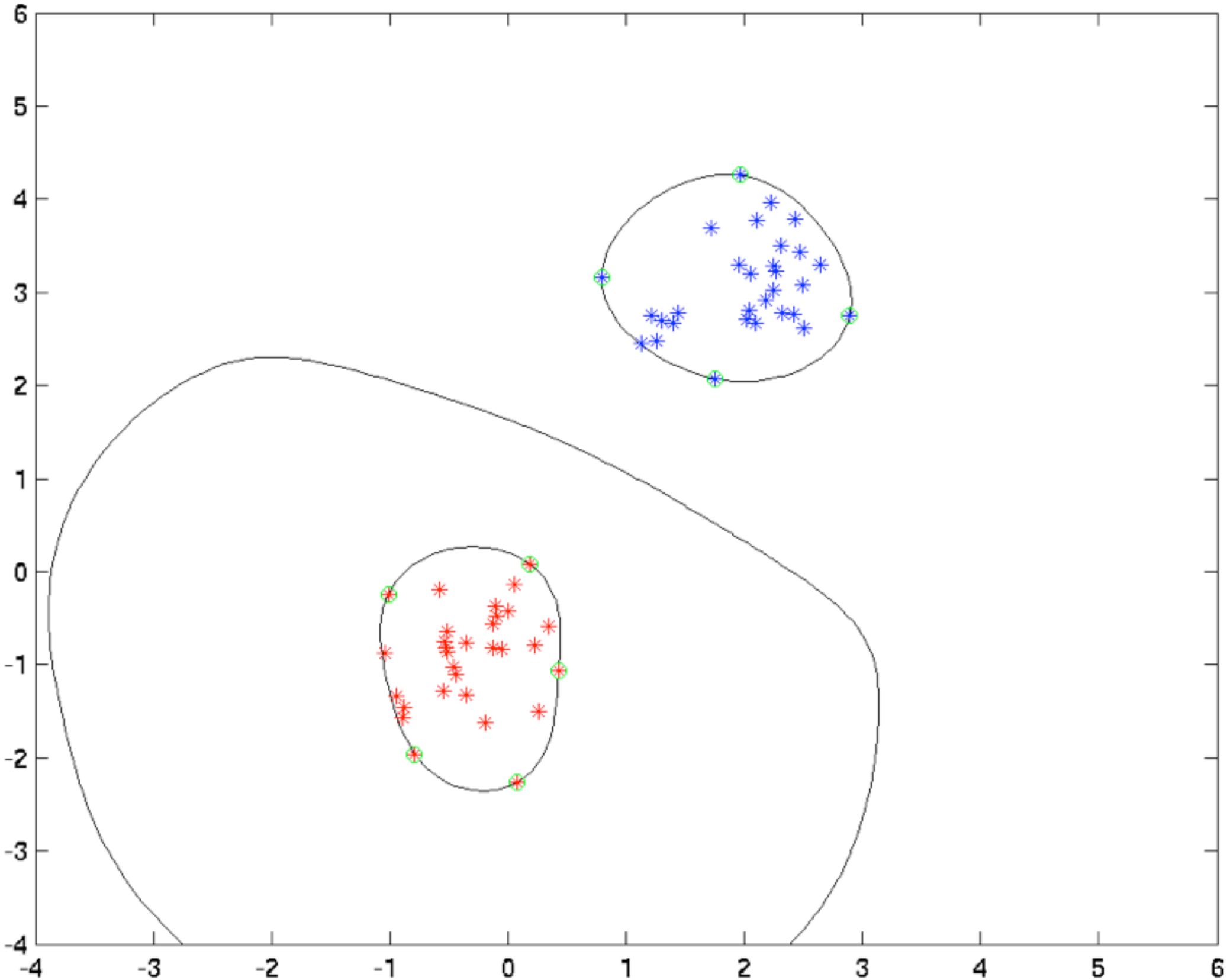
And now with a narrower kernel

(i.e., the 'spread' is small)

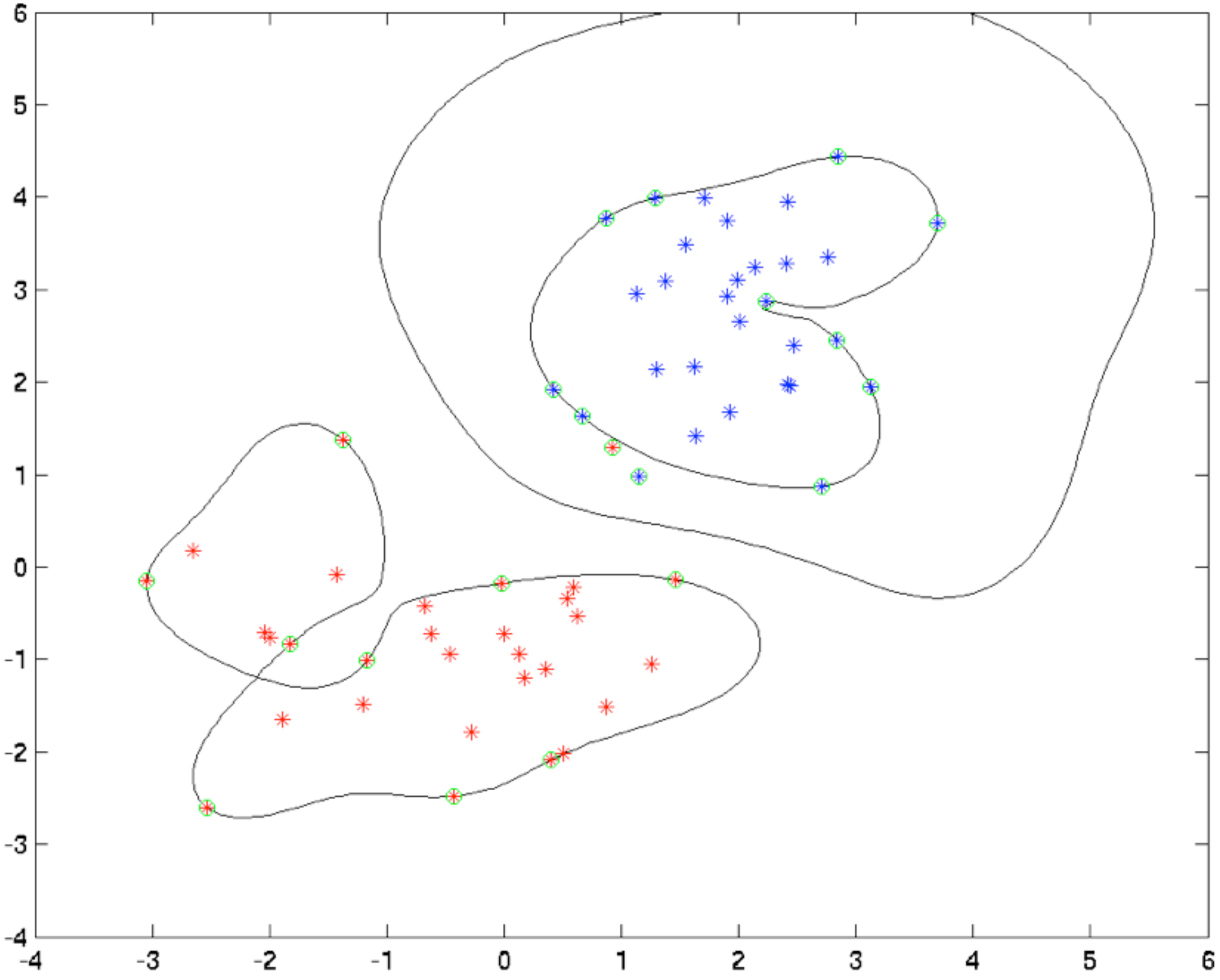
$$\exp(-\tfrac{1}{2\sigma^2}\|x - x'\|^2)$$



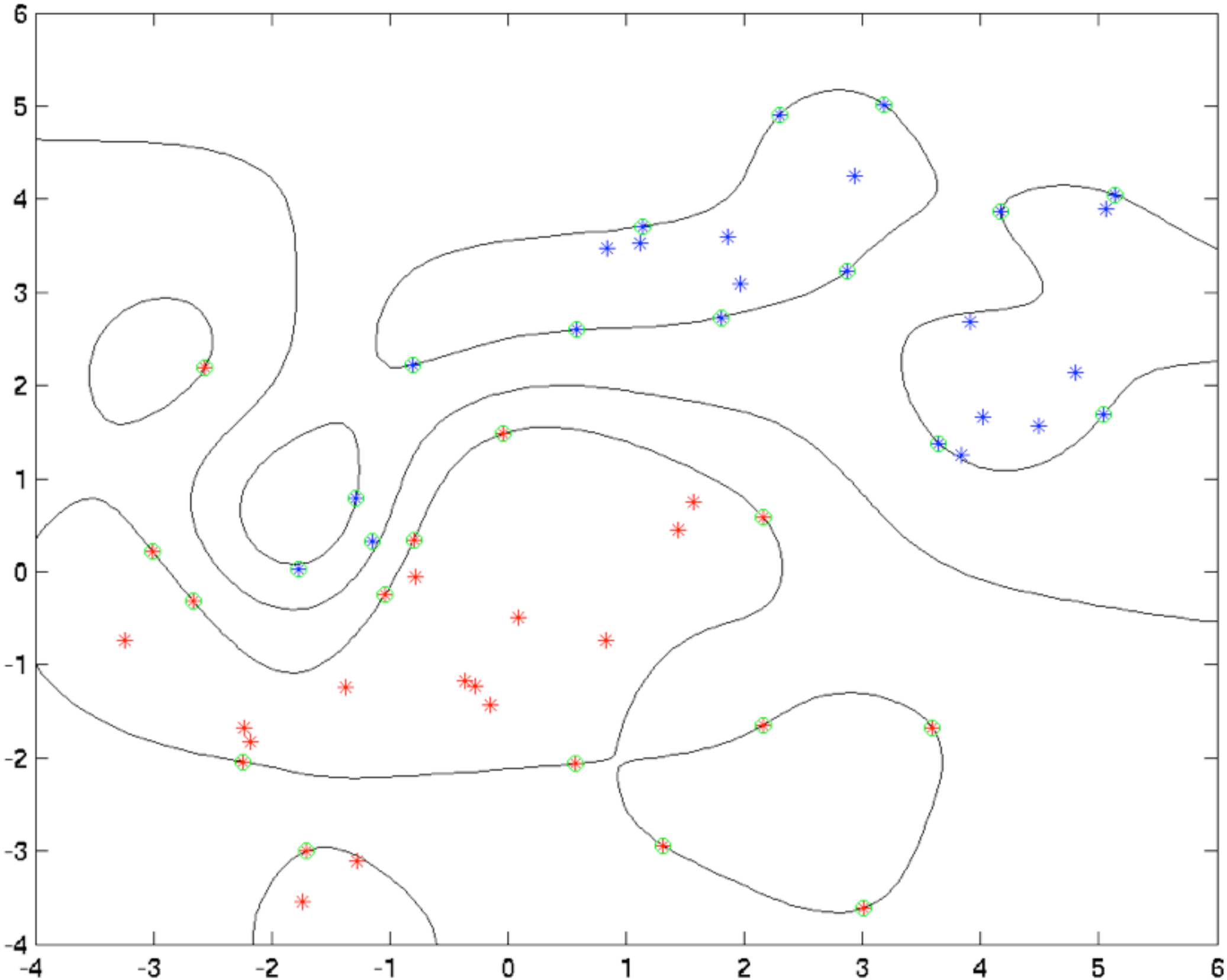
[thanks to: Alex Smola]



[thanks to: Alex Smola]



[thanks to: Alex Smola]

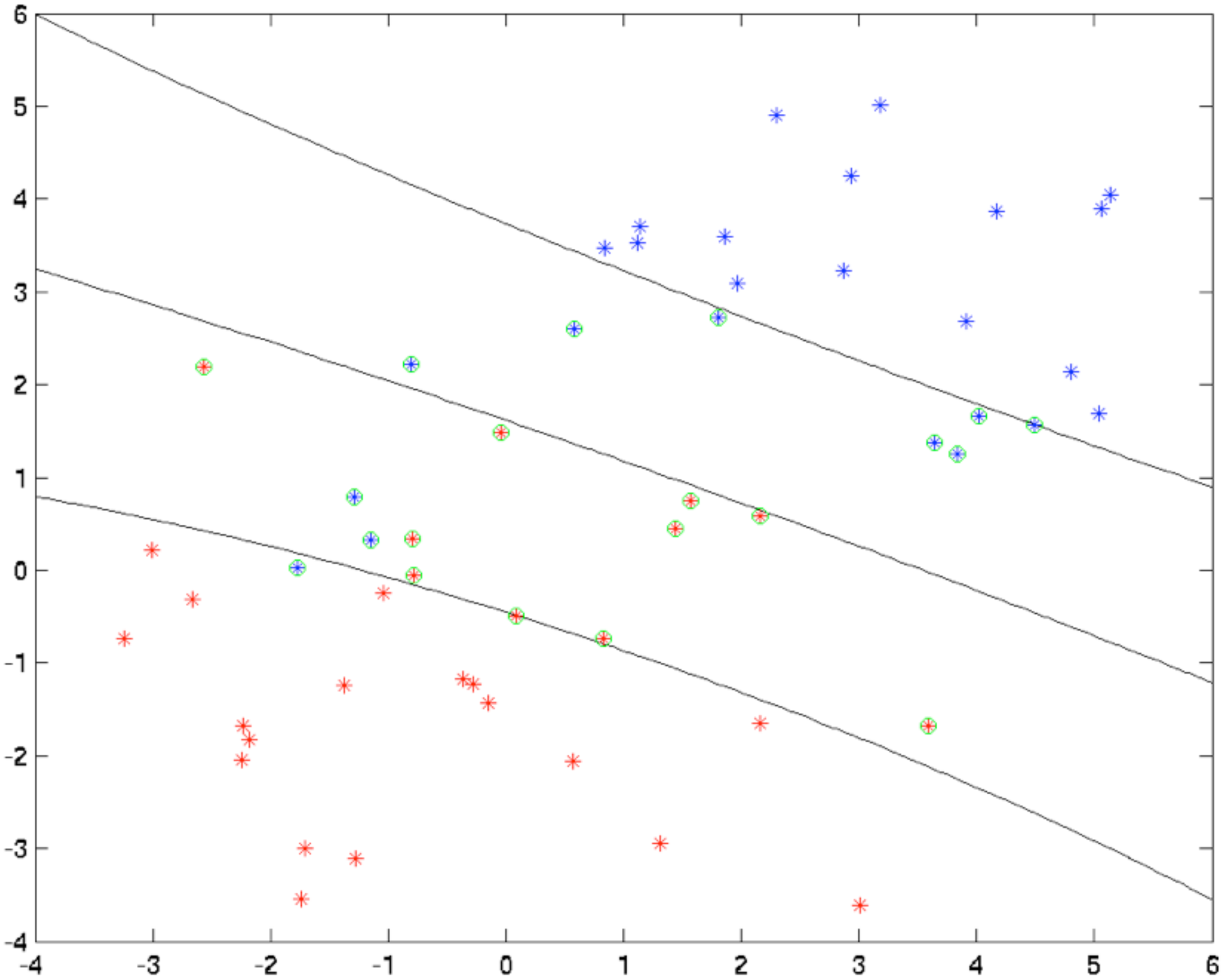


[thanks to: Alex Smola]

And now with a very wide kernel

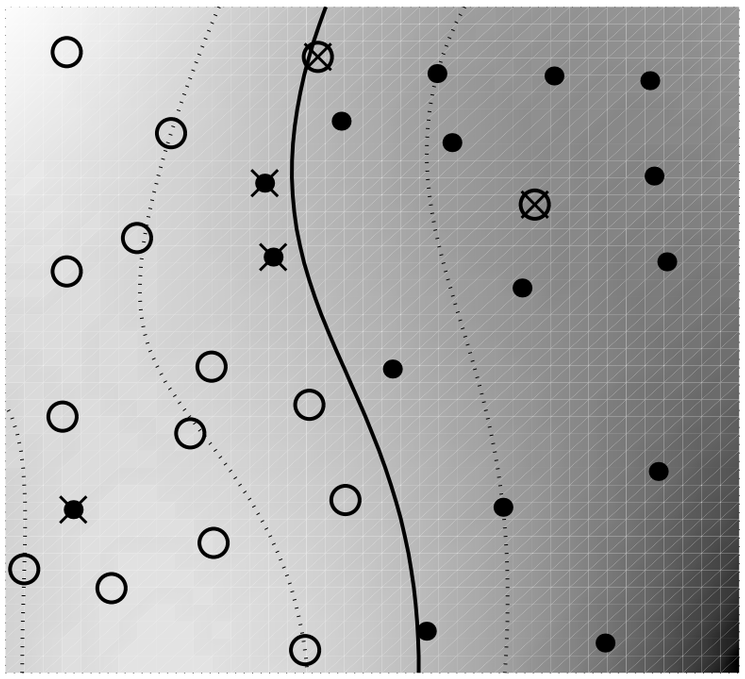
(i.e., the 'spread' is very large)

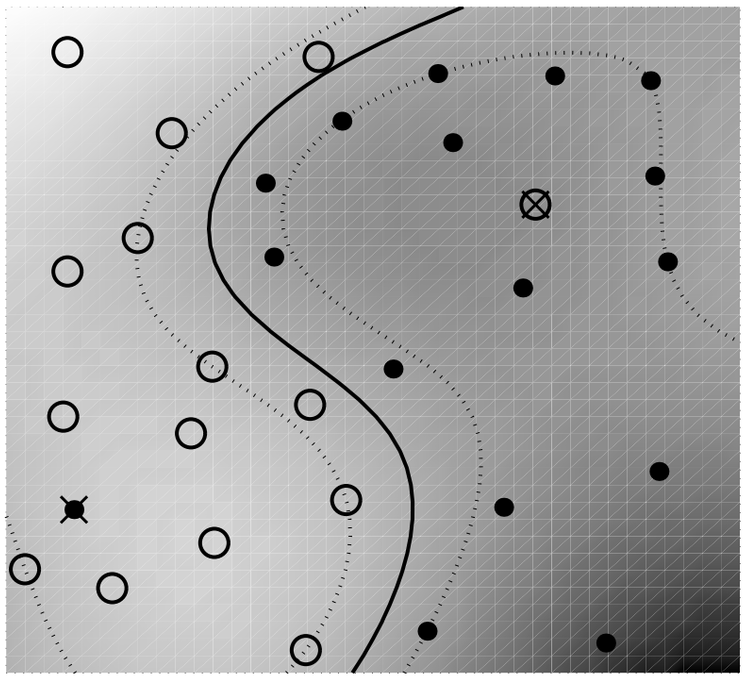
$$\exp(-\tfrac{1}{2\sigma^2}\|x - x'\|^2)$$

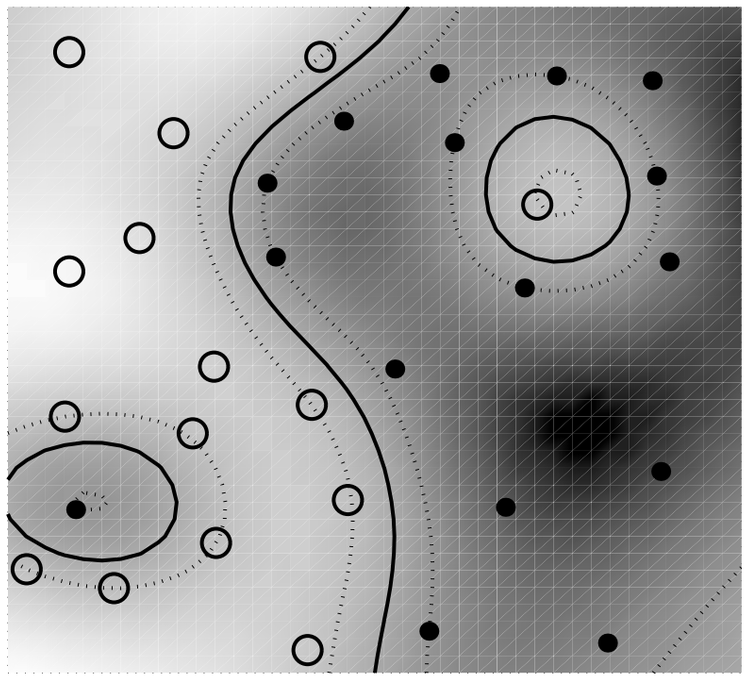


[thanks to: Alex Smola]

Nonlinear separation







- Increasing C allows for more nonlinearities
- Decreases number of training errors (overfitting?)
- SV boundary need not be contiguous
- Kernel width adjusts function class

[thanks to: Alex Smola]