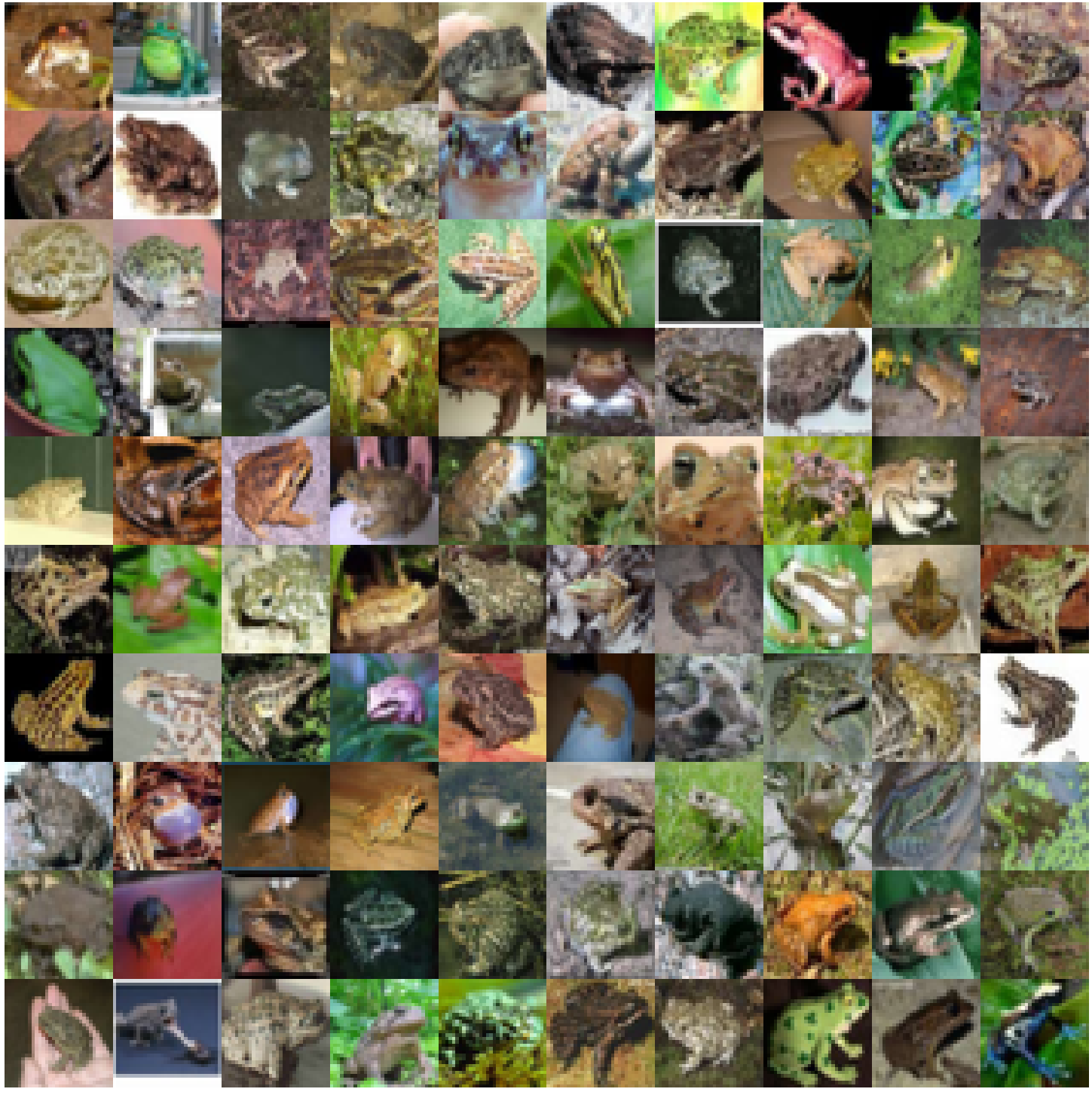
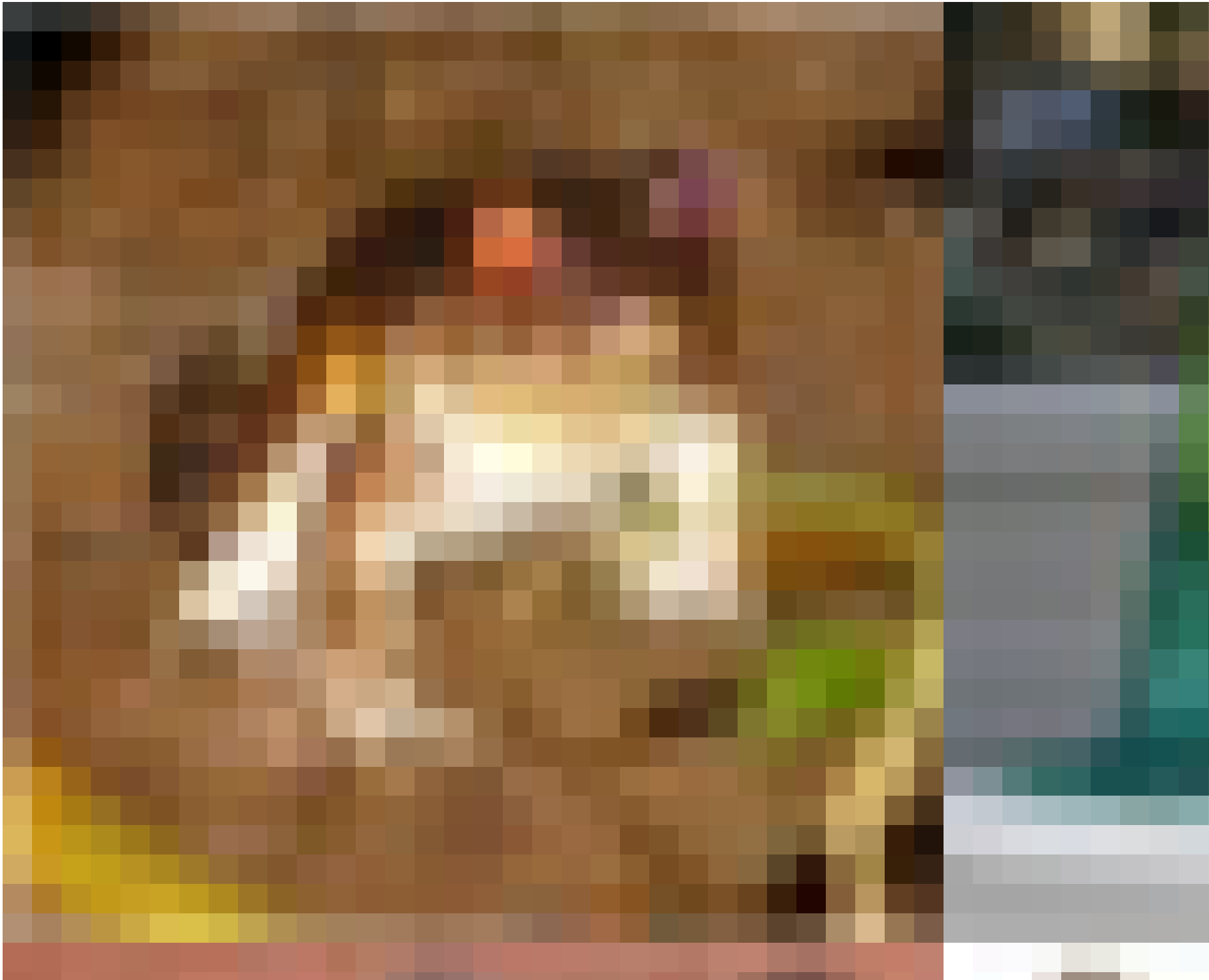
ST340: Support Vector Machines

Ben Graham
University of Warwick
Departments of Statistics and Complexity Science





SVMs

Vapnik (1979) developed a theory of statistical learning.

Use hyperplanes to cut up space.

Using the kernel trick you can control the shape of the hyperplane.

SVMs state of the art for problems with a strong geometrical flavour.

SVMs

Supervised learning

You have a bunch of points in a high dimensional space.

Each point belongs to a one of a (smallish) number of classes.

New points need to be classified.

Example: Cancer screening.

Data: Various measurements of cell samples + class labels.

Classes = { cancer, not cancer }

Notation

2 classes: Names $+1$ and -1

Data points: $\mathbf{x}_i \in \mathbb{R}^n$, $i = 1, ..., \ell$.

Labels/truth: $y_i \in \{-1, +1\}$.

Only 2 classes?

Suppose you have m classes $\{1, \ldots, m\}$.

Use voting.

One vs rest: Train m machines.

One vs one: Train $\binom{m}{2}$ machines.

WLOG, 2 classes $+1$ and -1.

Vapnik-Chervonenskis dimension

VC dimension h is a property of a class of functions $\{f(\cdot, \alpha)\}$.

Each $f(\cdot, \alpha)$ maps $\mathbb{R}^n \to \{-1, +1\}$.

i.e. it cuts up $\mathbb{R}^n$ into a $+$ and $-$ parts.

ℓ points can be labelled in 2^ℓ ways.

A class shatters the ℓ points if generates all the 2^ℓ labellings.

Capacity of a class: the maximum ℓ such that some ℓ points can be shattered.

N.B. A class having capacity ℓ does not mean it can shatter *any* ℓ points, only at least one set of ℓ points.

Oriented Hyperplanes in $\mathbb{R}^n$

Dot product: $\mathbf{x} \cdot \mathbf{y} = \sum_i x_i y_i$

$|\mathbf{x} \cdot \mathbf{y}| = \|\mathbf{x}\| \|\mathbf{y}\| \cos\theta$

If $\mathbf{w} \neq \mathbf{0}$, $\mathbf{x} \cdot \mathbf{w} + b = 0$ defines a hyperplane:
– it is $n - 1$ dimnesional,
– at right angles to $\mathbf{w}$,
– passing through the point $(-b/\|\mathbf{w}\|^2)\mathbf{w}$.

$\mathbb{R}^n$ is split into a $\mathbf{x} \cdot \mathbf{w} + b > 0$ side and a $\mathbf{x} \cdot \mathbf{w} + b < 0$ side.

i.e. in $\mathbb{R}^2$, $\mathbf{x} \cdot (1, 2) + 3 = 0$ defines the line $x_1 + 2x_2 + 3 = 0$.

Oriented hyperplanes in $\mathbb{R}^n$

$f(\mathbf{x}, \alpha) = \text{sign}(\mathbf{x} \cdot \mathbf{w} + b), \quad \alpha = (\mathbf{w}, b).$

The $n + 1$ points $\mathbf{0}$, $\mathbf{e}_1$, $\ldots \mathbf{e}_n$ in $\mathbb{R}^n$ can be wshattered by oriented hyperplanes.

The VC dimension is $n + 1$.

$(0, 1)$, $(0, 2)$, $(0, 3)$ cannot be shattered by a line in $\mathbb{R}^2$.

$(0, 0, 0)$, $(0, 1, 0)$, $(1, 0, 0)$, $(1, 1, 0)$ cannot be shattered by an oriented plane in $\mathbb{R}^3$.

Risk

Unknown probability distribution $\mathbb{P}(\mathbf{x}, y)$.

Actual risk is the expected test error

$$R(\alpha) = \tfrac{1}{2}\ \mathbb{E}[|\,Y - f(\mathbf{X}, \alpha)|].$$

Empirical risk is

$$R_{\text{emp}}(\alpha) = \frac{1}{2\ell} \sum_{i=1}^{\ell} |y_i - f(\mathbf{x}_i, \alpha)|$$

Theorem There is a function Q such that

$$\mathbb{P}[R(\alpha) \leq R_{\text{emp}}(\alpha) + Q(h, \ell, \varepsilon)] \geq 1 - \varepsilon.$$

Q is increasing with VC dimension h,
decreasing with number of samples ℓ,
decreasing with ε,
independent of the distribution $\mathbb{P}$.

A weird class $\{f(\cdot, \alpha)\}$ for $\mathbb{R} = \mathbb{R}^1$

$f(x, \alpha) = \sin(\alpha x)$

$x_i = 10^{-i}$, $i = 1, ..., \ell$

$y_i \in \{-1, +1\}$

$\alpha = \pi(1 + \frac{1}{2}\sum_{i=1}^{\ell}(1 - y_i)10^i)$

Infinite VC dimension!

Powerful? Cannot even shatter $\{+1, -1\}$!!

VC dimension intuition

The idea is that the higher the dimension, the more complex the learned models, so the greater the danger of overfitting.

However, VC dimension is an imperfect measure of model complexity.

The model on the previous slide has infinite VC dimension but cannot represent many very simple partitions of $\mathbb{R}$.

Some models have infinite VC dimension, and can fit very complicated partitions, but don't overfit because the various parameters can be set appropriately by validation.

Warm up: Linear SVMs—separable case

Training data is separable if there is an oriented hyperplane that splits $\mathbb{R}^n$ perfectly,

$$\exists \mathbf{w}, b: \quad \forall i, \operatorname{sign}(\mathbf{x}_i \cdot \mathbf{w} + b) = y_i.$$

Equivalently

$$\exists \mathbf{w}, b: \quad \forall i, y_i(\mathbf{x}_i \cdot \mathbf{w} + b) > 0.$$

Equivalently,

$$\exists \mathbf{w}, b: \quad \forall i, y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 \quad \text{and} \quad \|\mathbf{w}\| \text{ is minimal.}$$

Why $\|\mathbf{w}\|$ minimal? It gives the maximum margin hyperplane.

[The inequality is tight for some $y_i = +1$ and some $y_i = -1$.]

Warm up: Linear SVMs—separable case

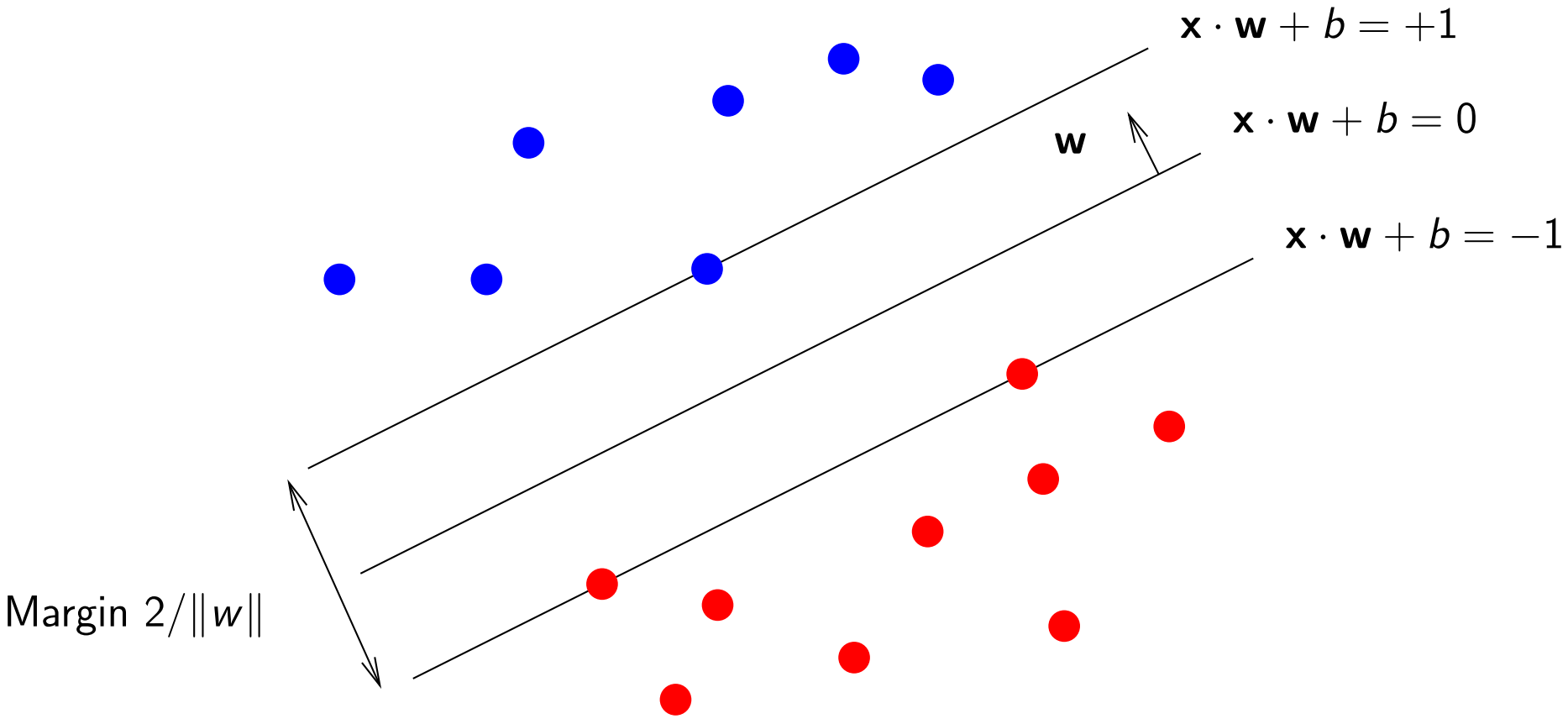
Training data is separable if there is an oriented hyperplane that splits $\mathbb{R}^n$ appropriately,

$$\exists \mathbf{w}, b: \quad \forall i, y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 \quad \text{and} \quad \|\mathbf{w}\| \text{ is minimal.}$$

Let d_+, d_- denote the minimum distances from the $+1$ and -1 labels to $\mathbf{x} \cdot \mathbf{w} + b = 0$, respectively.

$$\textit{Margin} := d_+ + d_- = 2/\|\mathbf{w}\|.$$

Warm up: Linear SVMs—separable case



Separable data

```
x1 <- c(rnorm(50,mean=0),rnorm(50,mean=3))
x2 <- c(rnorm(50,mean=0),rnorm(50,mean=3))
y  <- c(rep(+1,50),rep(-1,50))
plot(x2,x1,col=(y+3)/2,asp=1)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="linear",cost=10^6)
summary(s)
plot(s,d,grid=200,asp=1)
```

Linear SVMs, non-separable case

Introduce extra variables (ξ_i) to allow for some errors.

Choose $(\mathbf{w}, b, (\xi_i))$ to minimize $\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i$

subject to the constraints

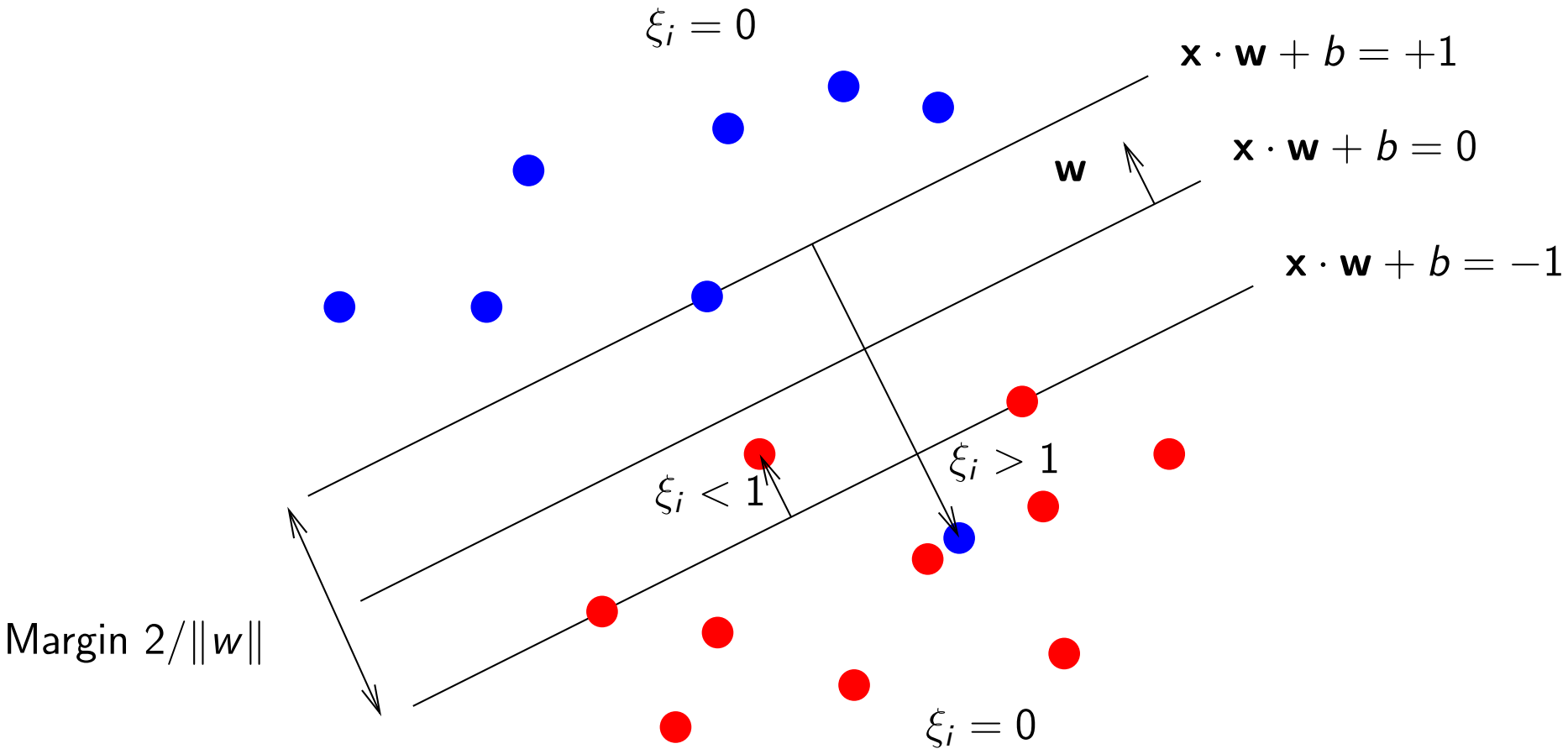
$\forall i,\ \xi_i \geq 0$
$\forall i,\ y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 - \xi_i.$

$\#\text{errors} = \sum_i 1_{\xi_i > 1} \leq \sum_i \xi_i.$

N.B. $\xi_i \in (0, 1)$ is punished, even though the label is correct.

Choice of C—validation.

Linear SVMs, non-separable case



Separable data

```
x1 <- c(rnorm(50,mean=0),rnorm(50,mean=3))
x2 <- c(rnorm(50,mean=0),rnorm(50,mean=3))
y  <- c(rep(+1,50),rep(-1,50))
plot(x2,x1,col=(y+3)/2,asp=1)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="linear",cost=1)
summary(s)
plot(s,d,grid=200,asp=1)
```

Non-separable data

```
x1 <- c(rnorm(50,mean=0),rnorm(50,mean=2))
x2 <- c(rnorm(50,mean=0),rnorm(50,mean=2))
y  <- c(rep(+1,50),rep(-1,50))
plot(x2,x1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="linear",cost=1)
summary(s)
plot(s,d,grid=200)
```

Non-separable data

```
x1 <- c(rnorm(50,mean=0),rnorm(50,mean=1))
x2 <- c(rnorm(50,mean=0),rnorm(50,mean=1))
y  <- c(rep(+1,50),rep(-1,50))
plot(x2,x1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="linear",cost=1)
summary(s)
plot(s,d,grid=200)
```

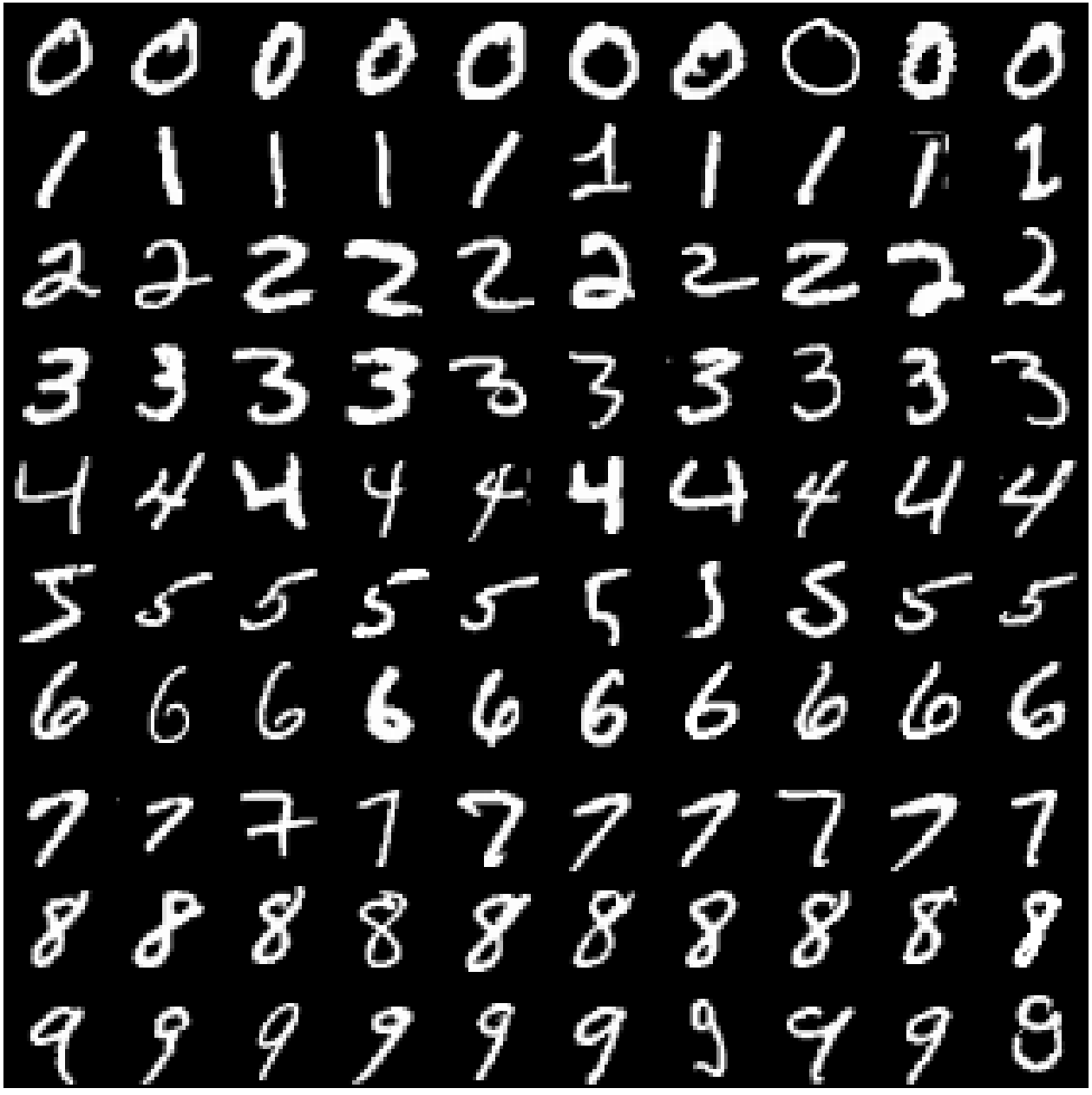
Bad

```
x1 <- rnorm(200,mean=rep(c(1,2,-2,-2),50))
x2 <- rnorm(200,mean=rep(c(1,-2,-2,2),50))
y  <- rep(c(+1,-1,-1,-1),50)
plot(x2,x1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="linear",cost=1)
summary(s)
plot(s,d,grid=200)

mean(predict(s,d)==y)
```

Disaster

```
x1 <- rnorm(200,mean=rep(c(2,2,-2,-2),50))
x2 <- rnorm(200,mean=rep(c(2,-2,-2,2),50))
y  <- rep(c(+1,-1),100)
plot(x2,x1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="linear",cost=1)
summary(s)
plot(s,d,grid=200)

mean(predict(s,d)==y)
#plot(x1*x2,x2,col=(y+3)/2) ##How to find?
```

10 Digits



10 Digits

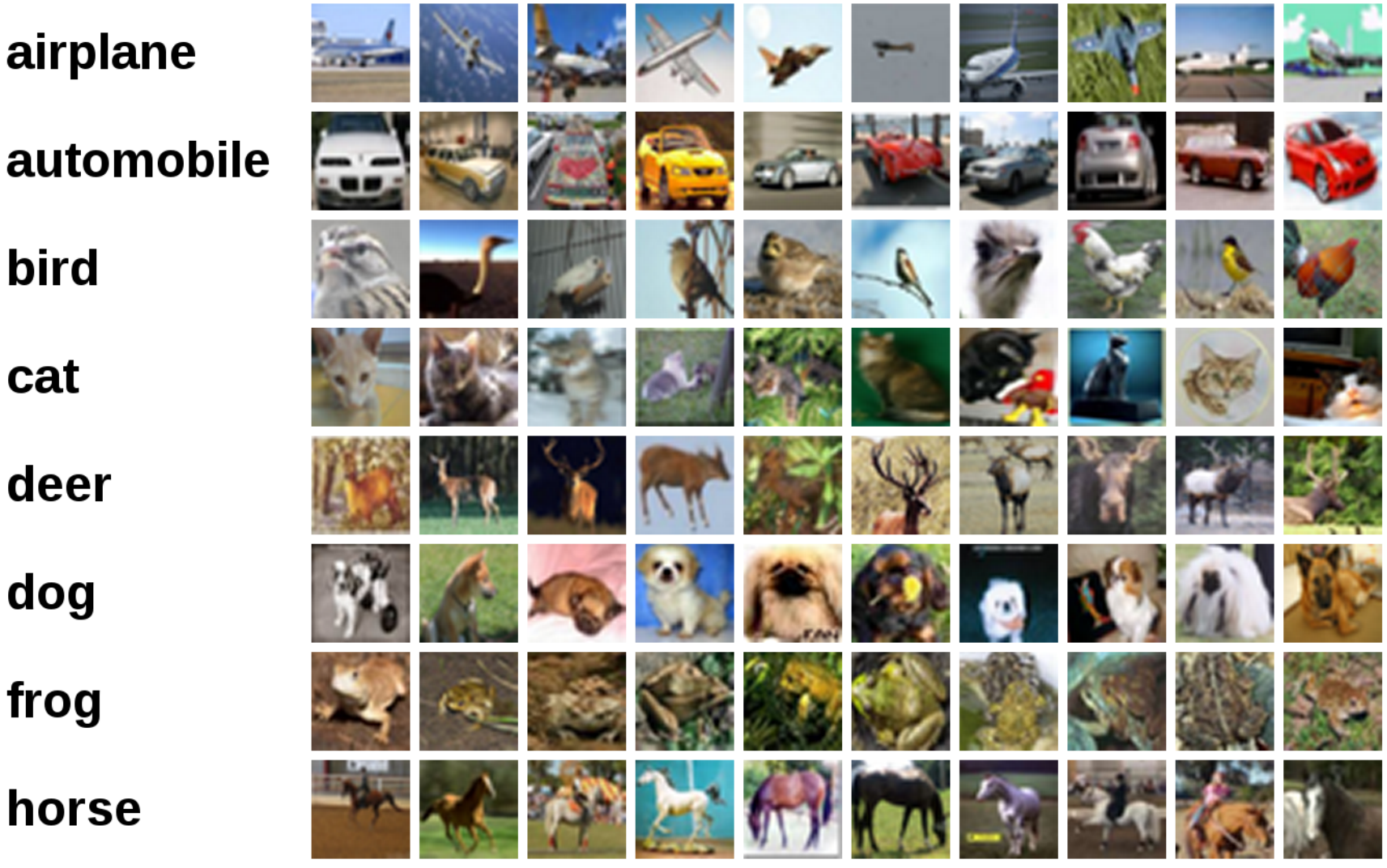
```
require(e1071)
load("mnist.small.RData")
features <- (apply(train.X,2,sd)>0)
train.X=train.X[,features]
test.X=test.X[,features]
s=svm(train.X,train.labels,type="C",
      kernel="l",cost=1)
mean(predict(s,train.X)==train.labels)
mean(predict(s,test.X)==test.labels)
```

Frog versus Horse

Frog versus Horse

```
require(e1071)
load("frog-horse.RData")
s=svm(train.X,train.labels,type="C",
      kernel="l",cost=1)
mean(predict(s,train.X)==train.labels)
mean(predict(s,test.X)==test.labels)
```

10 Categories

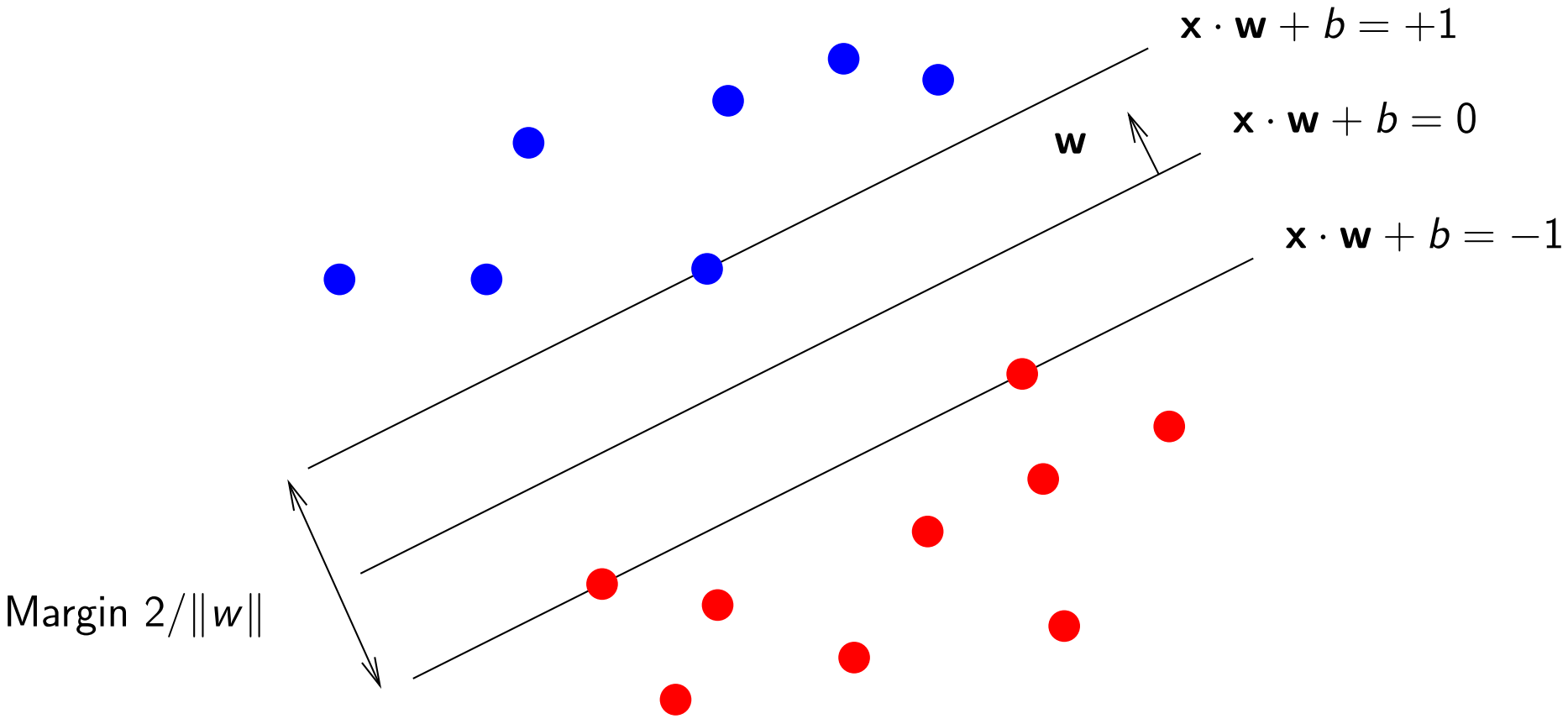


10 Catagories

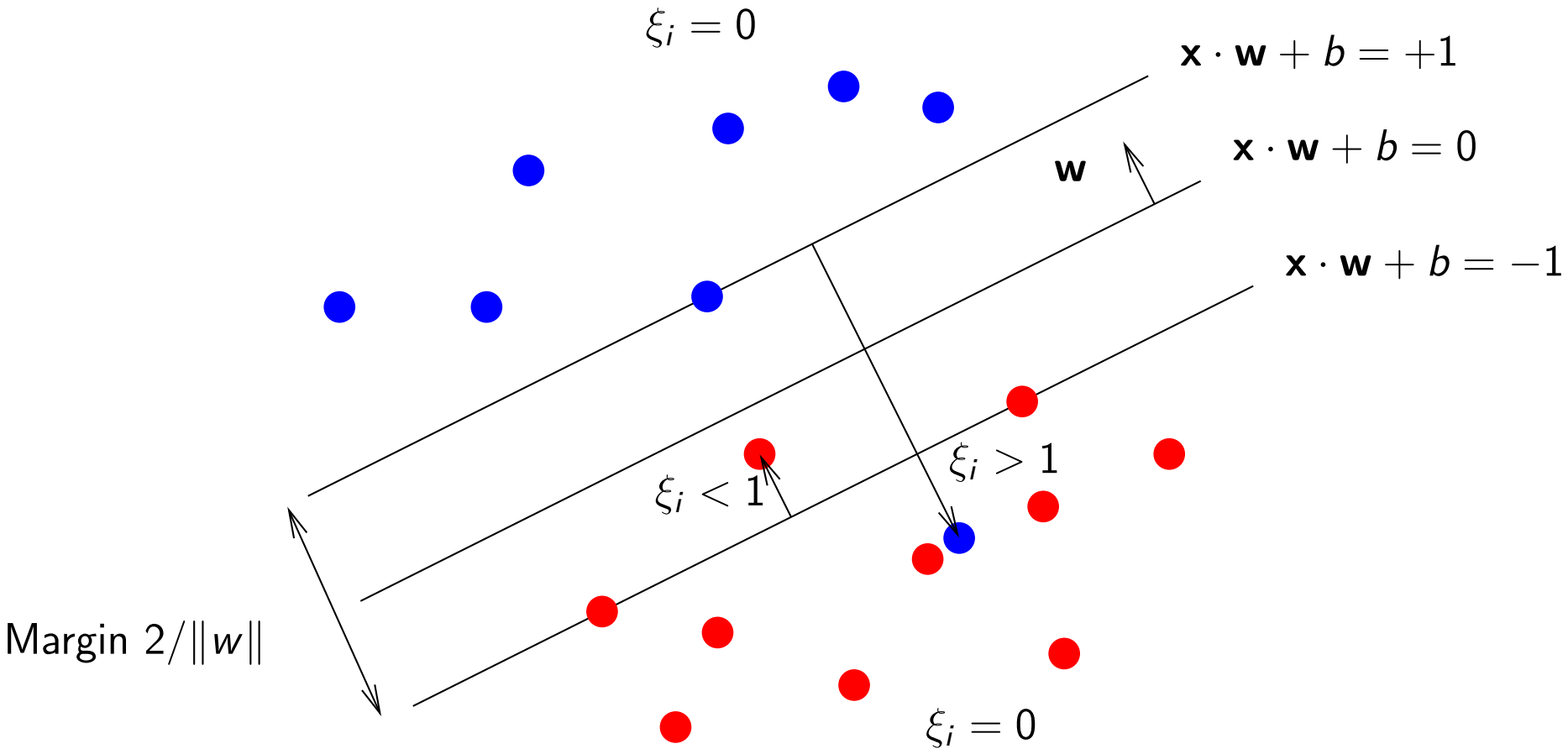
```
require(e1071)
load("cifar10-small.RData")
s=svm(train.X,train.labels,type="C",
      kernel="l",cost=1)
mean(predict(s,train.X)==train.labels)
mean(predict(s,test.X)==test.labels)
```

Recap

Linear SVMs—separable case



Linear SVMs, non-separable case



Linear SVMs, non-separable case

Introduce extra variables (ξ_i) to allow for some errors.

Choose $(\mathbf{w}, b, (\xi_i))$ to minimize $\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i$

subject to the constraints

$\forall i,\ \xi_i \geq 0$
$\forall i,\ y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 - \xi_i.$

$\#\text{errors} = \sum_i 1_{\xi_i > 1} \leq \sum_i \xi_i.$

N.B. $\xi_i \in (0, 1)$ is punished, even though the label is correct.

Choice of C—validation.

Non-linear SVMs?

```
x1=rnorm(1000)
x2=rnorm(1000)
y=factor(x1^2+x2^2<1)
plot(x1,x2,col=y,asp=1)

require(e1071)
d=data.frame(x1=x1,x2=x2,y=y)
s=svm(y~x1+x2,d,type="C",kernel="l",cost=1)
mean(predict(s,d)==y)
```

Equation of a circle

$\mathbf{x} \in \mathbb{R}^2$.

$$(x_1 - a_1)^2 + (x_2 - a_2) = r^2$$

Augment dataset

$$\hat{\mathbf{x}} = (x_1, x_2, x_1^2, x_2^2)$$

Hyperplane

$$\hat{\mathbf{x}} \cdot w + b = 0$$

i.e.

$$w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + b = 0$$

Non-linear SVMs?

```
x1=rnorm(1000)
x2=rnorm(1000)
y=factor(x1^2+x2^2<1)
plot(x1,x2,col=y,asp=1)

require(e1071)
d=data.frame(x1=x1,x2=x2,x1s=x1^2,x2s=x2^2,y=y)
s=svm(y~x1+x2+x1s+x2s,d,type="C",kernel="l",cost=1)
mean(predict(s,d)==y)
```

Linear SVMs, separable case—Properties of the solution

Theorem: The solution takes the form

$$\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i \mathbf{x}_i.$$

Proof: Suppose

$$\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i \mathbf{x}_i + \mathbf{v}.$$

with $\mathbf{v}$ orthogonal to the $\mathbf{x}_i$. Getting rid of $\mathbf{v}$:
—does not affect any of the inequalities,
—reduces $\|\mathbf{w}\|$.

□

Kernel trick

The SVM solver thus has a specific challenge:

find $\alpha_1, \ldots, \alpha_\ell$ and $\xi_1, \ldots, \xi_\ell \geq 0$ such that

$$\frac{1}{2} \left\| \sum_{i=1}^{\ell} \alpha_i \mathbf{x}_i \right\|^2 + C \sum_i \xi_i \text{ is minimal subject to}$$

$$\forall i, y_i \left(\mathbf{x}_i \cdot \left(\sum_{j=1}^{\ell} \alpha_j \mathbf{x}_j \right) + b \right) \geq 1 - \xi_i.$$

Kernel trick

The SVM solver thus has a specific challenge:

find $\alpha_1, \ldots, \alpha_\ell$ and $\xi_1, \ldots, \xi_\ell \geq 0$ such that

$$\frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j) + C \sum_i \xi_i \text{ is minimal subject to}$$

$$\forall i, y_i \left(\left(\sum_{j=1}^{\ell} \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j) \right) + b \right) \geq 1 - \xi_i.$$

Kernel trick

The SVM solver thus has a specific challenge:

find $\alpha_1, \ldots, \alpha_\ell$ and $\xi_1, \ldots, \xi_\ell \geq 0$ such that

$$\frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j) + C \sum_i \xi_i \text{ is minimal subject to}$$

$$\forall i, y_i \left(\left(\sum_{j=1}^{\ell} \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j) \right) + b \right) \geq 1 - \xi_i.$$

Amazing! The algorithm only depends on the $(\mathbf{x}_i)_{1 \leq i \leq \ell}$ through the dot products $(\mathbf{x}_i \cdot \mathbf{x}_j)_{1 \leq i,j \leq \ell}$.

Kernel trick

So far, we have had to hope that the data is nearly separable.

Let $\phi : \mathbb{R}^n \to V$ denote a function mapping $\mathbb{R}^n$ into a higher dimensional space. Let $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ denote the function

$$K(\mathbf{x}, \hat{\mathbf{x}}) = \phi(\mathbf{x}) \cdot \phi(\hat{\mathbf{x}}).$$

(i) We can choose the kernel K without dealing with the corresponding ϕ.
(ii) 'K's are much simpler than 'ϕ's.
(ii) In V-space, separability can becomes a useful concept.

Kernels

Linear Kernel

$$K(\mathbf{x}, \hat{\mathbf{x}}) = \mathbf{x} \cdot \hat{\mathbf{x}}$$

Polynomial Kernel

$$K(\mathbf{x}, \hat{\mathbf{x}}) = (c + \gamma \mathbf{x} \cdot \hat{\mathbf{x}})^p, \qquad c \geq 0, p = 2, 3, \ldots$$

Radial Basis Function (RBF) Kernel

$$K(\mathbf{x}, \hat{\mathbf{x}}) = \exp(-\gamma \|\mathbf{x} - \hat{\mathbf{x}}\|^2), \qquad \gamma \geq 0$$

If in doubt, use an RBF kernel. Use validation to find C and γ.

Quadratic Kernel $K(\mathbf{x}, \hat{\mathbf{x}}) = (\mathbf{x} \cdot \hat{\mathbf{x}})^2$

$$\phi(\mathbf{x}) = (x_1^2,\ \sqrt{2}x_1x_2,\ x_2^2)$$

Or

$$\phi(\mathbf{x}) = ((x_1 - x_2)^2, 2x_1x_2, (x_1 + x_2)^2)$$

Check $K(\mathbf{x}, \hat{\mathbf{x}}) = \phi(\mathbf{x}) \cdot \phi(\hat{\mathbf{x}})$.

What do hyperplanes $\mathbf{w} \cdot \phi(\mathbf{x}) + b = 0$ look like now?

"Hyperplanes"

Curves $Ax_1^2 + Bx_1x_2 + Cx_2^2 + D = 0$

If $B^2 - 4AC < 0$: ellipse or circle
If $B^2 - 4AC = 0$: 2 parallel lines
If $B^2 - 4AC > 0$: hyperbola, or two intersecting lines

All centered at the origin.

Quadratic Kernel $K(\mathbf{x}, \hat{\mathbf{x}}) = (1 + \mathbf{x} \cdot \hat{\mathbf{x}})^2$

$$\phi(\mathbf{x}) = (1,\ \sqrt{2}x_1,\ \sqrt{2}x_2,\ x_1^2,\ \sqrt{2}x_1x_2,\ x_2^2)$$

What do hyperplanes $\mathbf{w} \cdot \phi(\mathbf{x}) + b = 0$ look like now?

"Hyperplanes"

Curves $Ax_1^2 + Bx_1x_2 + Cx_2 + D + Ex + Fy = 0$

If $B^2 - 4AC < 0$: ellipse or circle
If $B^2 - 4AC = 0$: parabola, 2 parallel lines
If $B^2 - 4AC > 0$: hyperbola, or two intersecting lines.

Can shift the curve in the x and y directions.

Non-separable data—Linear Kernel $K(\mathbf{x}, \hat{\mathbf{x}}) = \mathbf{x} \cdot \hat{\mathbf{x}}$

```
th=runif(100,-pi,pi)
y=sign(th)
x1=sin(th)-y/3+rnorm(100,sd=0.1)
x2=cos(th)+y/2+rnorm(100,sd=0.1)
plot(x2,x1,asp=1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="linear",cost=1)
mean(predict(s,d)==y)
plot(s,d,grid=200,asp=1)
```

Non-separable data—Quadratic Kernel $K(\mathbf{x}, \hat{\mathbf{x}}) = (1 + \mathbf{x} \cdot \hat{\mathbf{x}})^2$

```
th=runif(100,-pi,pi)
y=sign(th)
x1=sin(th)-y/3+rnorm(100,sd=0.1)
x2=cos(th)+y/2+rnorm(100,sd=0.1)
plot(x2,x1,asp=1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="polynomial",degree=2,coef=1,cost=1)
mean(predict(s,d)==y)
plot(s,d,grid=200,asp=1)
```

Non-separable data—Cubic Kernel $K(\mathbf{x}, \hat{\mathbf{x}}) = (1 + \mathbf{x} \cdot \hat{\mathbf{x}})^3$

```
th=runif(100,-pi,pi)
y=sign(th)
x1=sin(th)-y/3+rnorm(100,sd=0.1)
x2=cos(th)+y/2+rnorm(100,sd=0.1)
plot(x2,x1,asp=1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="polynomial",degree=3,coef=1,cost=1)
mean(predict(s,d)==y)
plot(s,d,grid=200,asp=1)
```

Non-separable data—Polynomial Kernel $K(\mathbf{x}, \hat{\mathbf{x}}) = (1 + \mathbf{x} \cdot \hat{\mathbf{x}})^7$

```
th=runif(100,-pi,pi)
y=sign(th)
x1=sin(th)-y/3+rnorm(100,sd=0.1)
x2=cos(th)+y/2+rnorm(100,sd=0.1)
plot(x2,x1,asp=1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="polynomial",degree=7,coef=1,cost=1)
mean(predict(s,d)==y)
plot(s,d,grid=200,asp=1)
```

Non-separable data—RBF Kernel $K(\mathbf{x}, \hat{\mathbf{x}}) = \exp(-\gamma\|\mathbf{x} - \hat{\mathbf{x}}\|^2)$

```
th=runif(100,-pi,pi)
y=sign(th)
x1=sin(th)-y/3+rnorm(100,sd=0.1)
x2=cos(th)+y/2+rnorm(100,sd=0.1)
plot(x2,x1,asp=1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="radial",gamma=1,cost=1)
mean(predict(s,d)==y)
plot(s,d,grid=200,asp=1)
```

Non-separable data—RBF Kernel $K(\mathbf{x}, \hat{\mathbf{x}}) = \exp(-\gamma\|\mathbf{x} - \hat{\mathbf{x}}\|^2)$

```
th=runif(100,-pi,pi)
y=sign(th)
x1=sin(th)-y/3+rnorm(100,sd=0.1)
x2=cos(th)+y/2+rnorm(100,sd=0.1)
plot(x2,x1,asp=1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="radial",gamma=10,cost=1)
mean(predict(s,d)==y)
plot(s,d,grid=200,asp=1)
```

Non-separable data—RBF Kernel $K(\mathbf{x}, \hat{\mathbf{x}}) = \exp(-\gamma\|\mathbf{x} - \hat{\mathbf{x}}\|^2)$

```
th=runif(100,-pi,pi)
y=sign(th)
x1=sin(th)-y/3+rnorm(100,sd=0.1)
x2=cos(th)+y/2+rnorm(100,sd=0.1)
plot(x2,x1,asp=1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="radial",gamma=0.1,cost=1)
mean(predict(s,d)==y)
plot(s,d,grid=200,asp=1)
```

Non-separable data—RBF Kernel $K(\mathbf{x}, \hat{\mathbf{x}}) = \exp(-\gamma\|\mathbf{x} - \hat{\mathbf{x}}\|^2)$

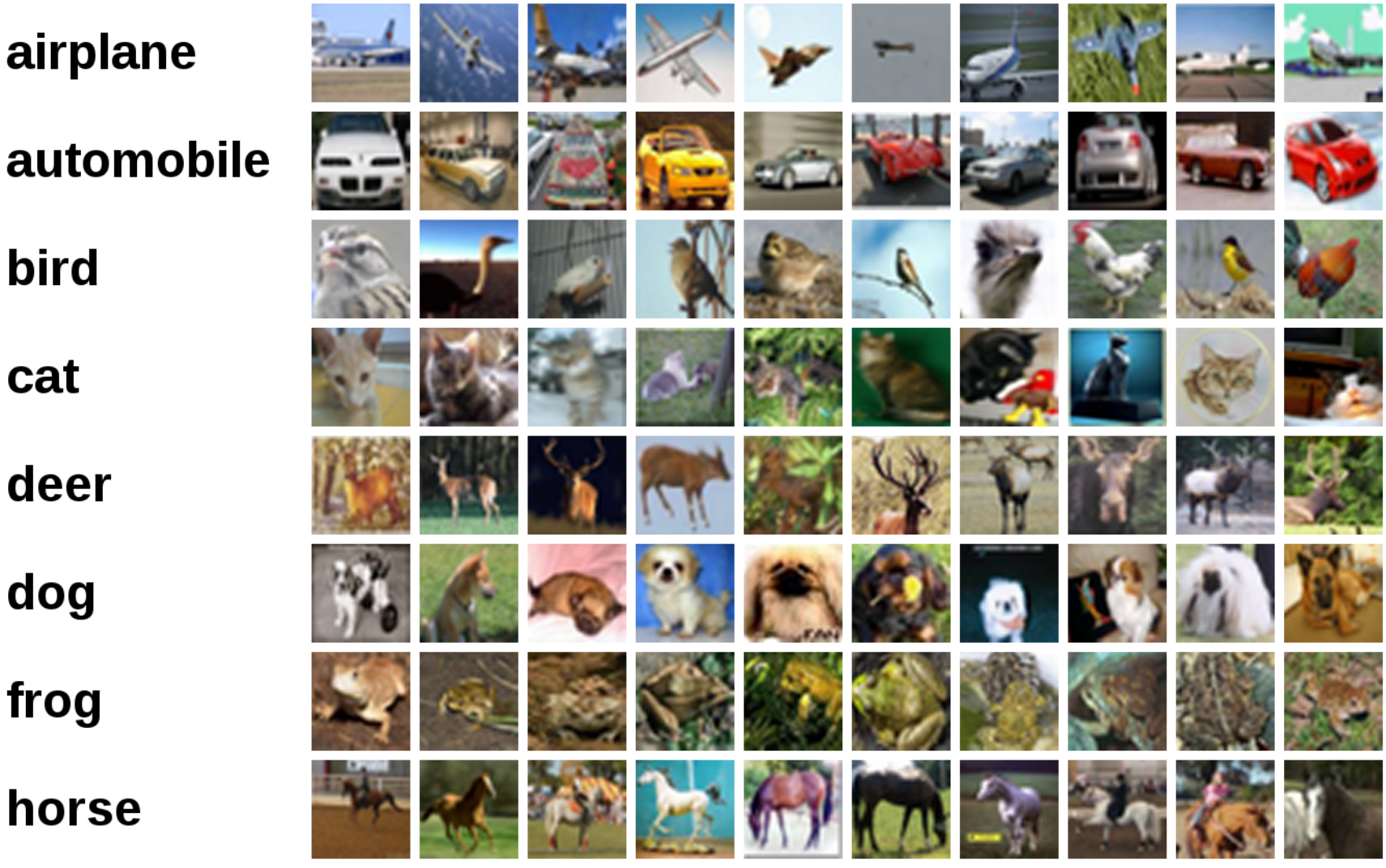
```
th=runif(100,-pi,pi)
y=sign(th)
x1=sin(th)-y/3+rnorm(100,sd=0.1)
x2=cos(th)+y/2+rnorm(100,sd=0.1)
plot(x2,x1,asp=1,col=(y+3)/2)

require(e1071)
d <- data.frame(x1=x1,x2=x2,y=factor(y))
s <- svm(y~x1+x2,d,type="C",
         kernel="radial",gamma=1,cost=100)
mean(predict(s,d)==y)
plot(s,d,grid=200,asp=1)
```

Frog versus Horse 500 each

Frog versus Horse Linear 73.4%

```
require(e1071)
load("frog-horse.RData")
s=svm(train.X,train.labels,type="C",kernel="p",
      degree=3,coef=1,cost=1)
mean(predict(s,train.X)==train.labels)
mean(predict(s,test.X)==test.labels)
```

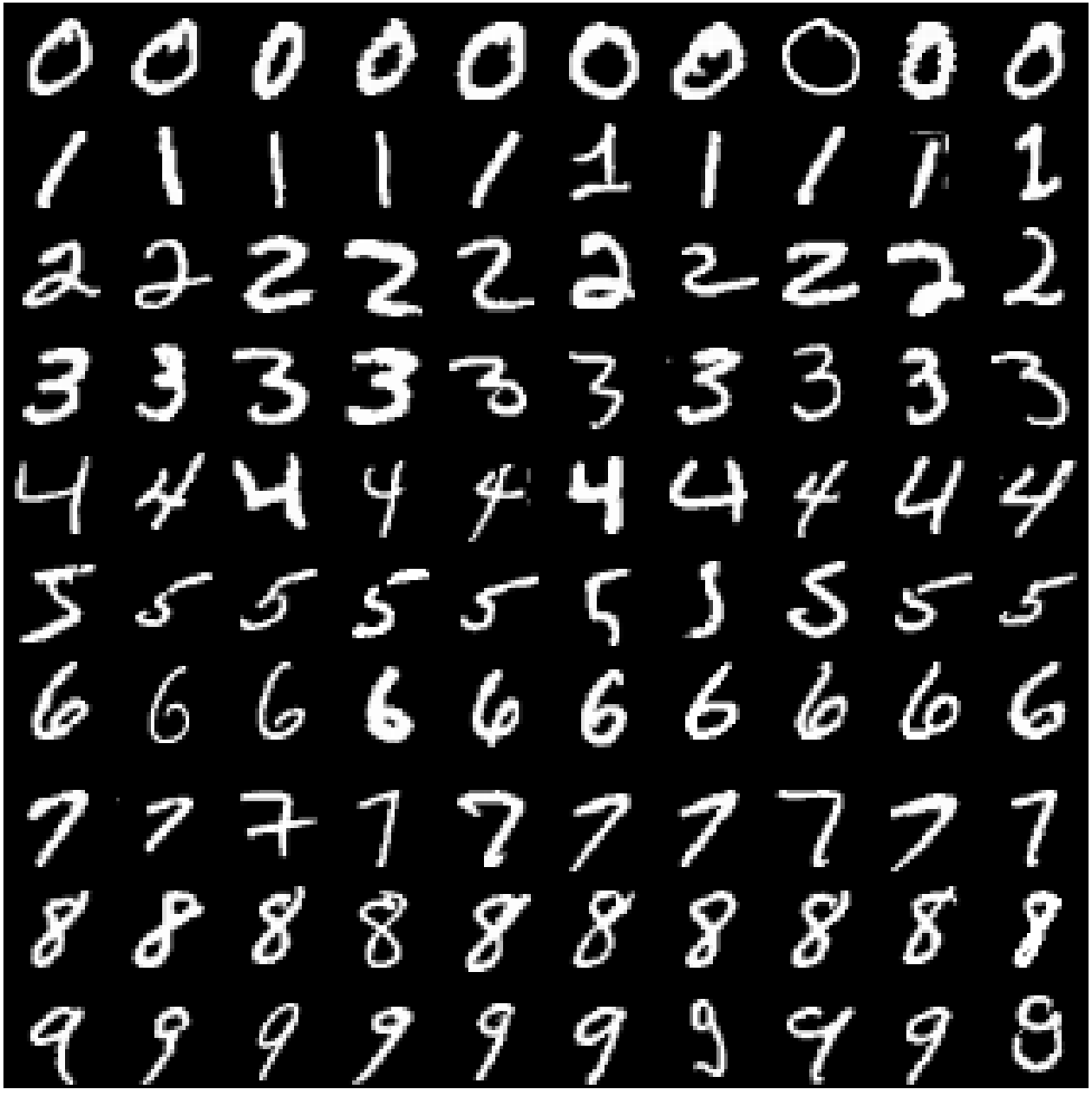
10 Catagories 100 each



10 Catagories Linear 28.5%

```
require(e1071)
load("cifar10-small.RData")
s=svm(train.X,train.labels,type="C",kernel="p",
      degree=2,coef=1,cost=1)
mean(predict(s,train.X)==train.labels)
mean(predict(s,test.X)==test.labels)
```

10 Digits 1000 each



10 Digits Linear 91.6%

```
require(e1071)
load("mnist.small.RData")
features <- (apply(train.X,2,sd)>0)
train.X=train.X[,features]
test.X=test.X[,features]
s=svm(train.X,train.labels,type="C",kernel="p",
      degree=4,coef=1,cost=1)
mean(predict(s,train.X)==train.labels)
mean(predict(s,test.X)==test.labels)
```

Cross validation

Integer $k > 1$

Split the data G into k groups $G_1, \ldots, G_k$ of size ℓ/k.

For each $i = 1, \ldots, k$, test G_i against training data $G_1, \ldots, G_{i-1}, G_{i+1}, \ldots, G_k$.

Stratergy

Try different parameters. Choose the one with the best average cross validation accuracy.

Train an SVM using the best parameters and all the training data.

Wisconsin Breast Cancer Diagnostic Data Set

```
load("wisconsin-breast-cancer.RData")
dim(x)
summary(Y)
s=svm(x,y,type="C",kernel="l",cost=1,cross=10)
s$accuracies
mean(s$accuracies)

s=svm(x,y,type="C",
      kernel="radial",gamma=1,cost=1,cross=10)
s$accuracies
mean(s$accuracies)
```

Cross-validation with grid search

Radial Basis Function Kernels require fine tuning.

Use cross validation on the training set. Use the best parameters to test the test set.

$\log\gamma \backslash \log C$	-4	-3	-2	-1	0
-6	64.95	82.84	94.72	96.48	97.06
-5	78.59	94.72	96.62	97.06	97.06
-4	94.86	96.48	97.21	97.06	96.92
-3	96.92	96.92	97.06	97.06	97.06
-2	96.48	96.48	96.48	96.48	96.48

The best value should be near the centre of the grid!