Machine Learning and Data Mining
Support Vector Machines

Patricia Hoffman, PhD

Slides are from
- Tan, Steinbach, and Kumar
- Ethem Alpaydin

Support Vector Machines
Main Sources

- Patricia Hoffman – Mathematics behind SVM
  - http://patriciahoffmanphd.com/resources/papers/SVM/SupportVectorJune3.pdf
- Andrew Ng – Lecture Notes
  - http://cs229.stanford.edu/notes/cs229-notes3.pdf
- John Platt – Fast Training of SVM
  - http://research.microsoft.com/apps/pubs/default.aspx?id=68391
- Kernel Methods
  - http://www.kernel-machines.org/
- C.J.C. Burges. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2(2):955-974, 1998. http://citeseer.nj.nec.com/burges98tutorial.html

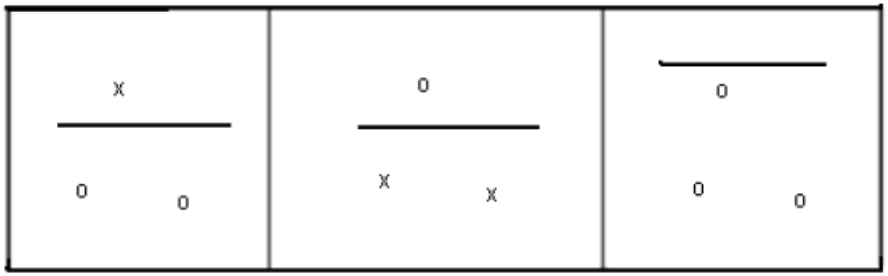
Example Code & R Package

- e1071
  - http://cran.r-project.org/web/packages/e1071/e1071.pdf
- svm1.r - Example of SVM
  - classification plots, tuning plots, confusion tables
  - Classification using SVM on
    - Iris Data Set, Glass Data Set
  - Regression using SVM on
    - Toy data set

SVM Main Advantage

- There are NO assumptions
  - Understanding Distribution – NOT necessary
  - NO distribution parameter estimation

Vapnik – Chervonenkis (VC Dimension)



Linearly Separable

The line Shatters the three points

Any three points in a plane can be separated by a line

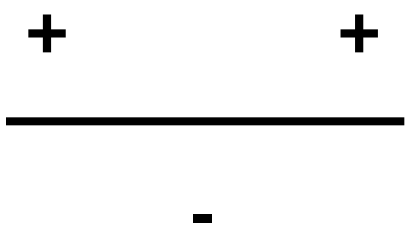
VC Dimension Example

Can't Shatter with line + - +

observation	target
-1	+
0	-
+1	+

Higher Dimension

Shattered with a line



observation	squared	target
-1	1	+
0	0	-
+1	1	+

VC (Vapnik-Chervonenkis) Dimension

- N points can be labeled in 2^N ways as +/–
- H SHATTERS N if there exists a set of N points such that $h \in$ H is consistent with all of these possible labels:
  - Denoted as: VC(H) = N
  - Measures the capacity of H
- Any learning problem definable by N examples can be learned with no error by a hypothesis drawn from H

Formal Definition

The VC Dimension

Definition: the VC dimension of a set of functions $H = \{h(\mathbf{x}, \alpha)\}$ is d if and only if there exists a set of points $\{x^i\}_{i=1}^{d}$ such that these points can be labeled in all 2^d possible configurations, and for each labeling, a member of set H can be found which correctly assigns those labels, but that no set $\{x^i\}_{i=1}^{q}$ exists where $q > d$ satisfying this property.

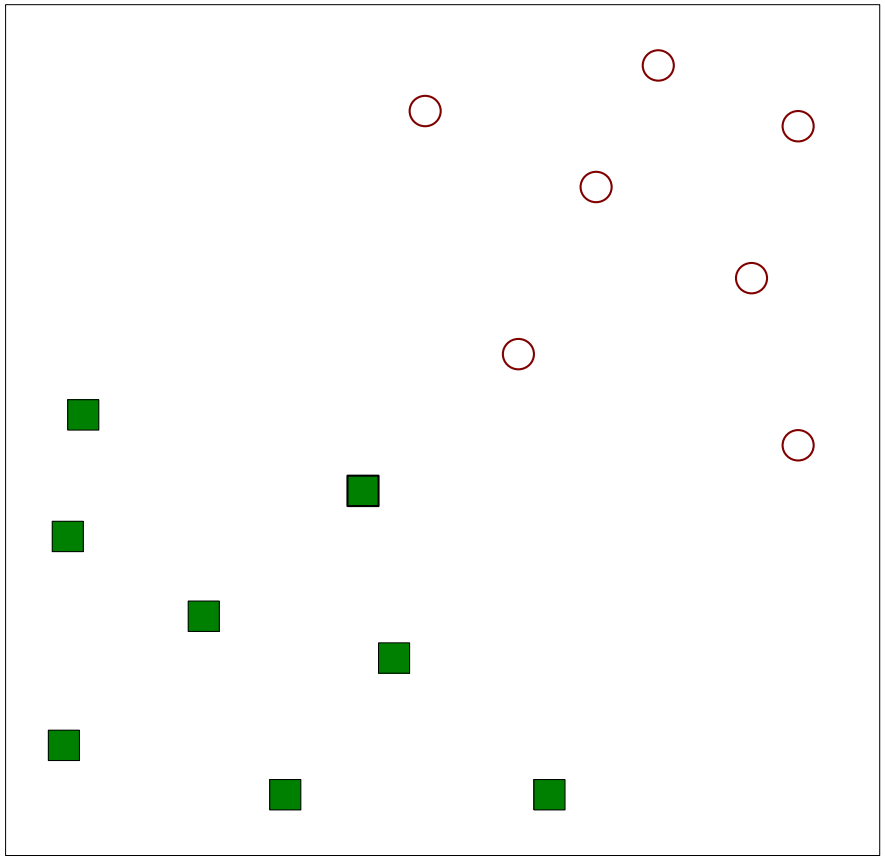
VC dim of linear classifiers in m-dimensions

If input space is m-dimensional and if **f** is *sign(**w**.**x**-b)*, what is the VC-dimension?

- h=m+1

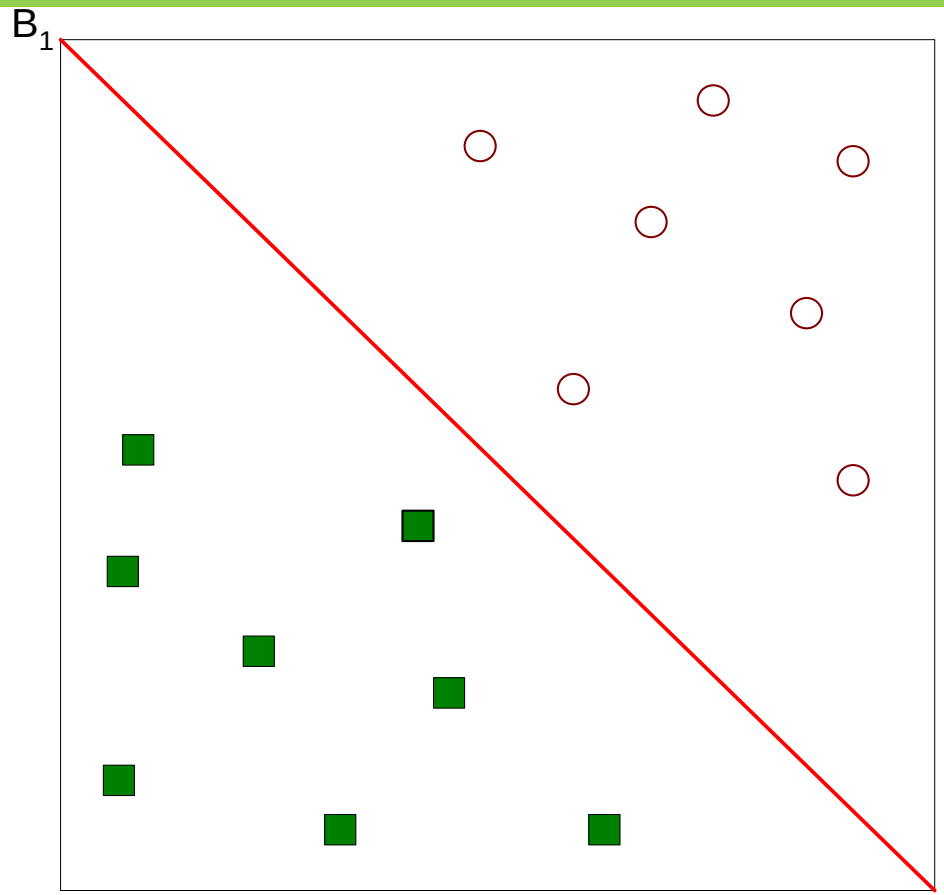
- Lines in 2D can shatter 3 points
- Planes in 3D space can shatter 4 points
- ...

Support Vector Machines



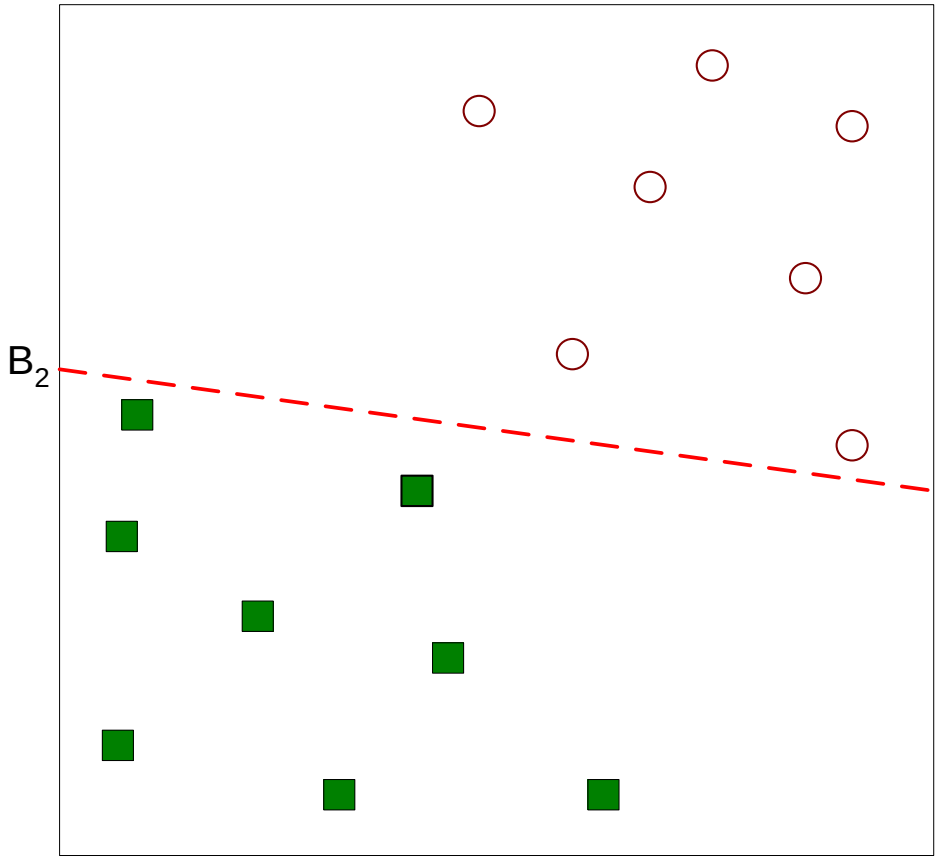
- Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



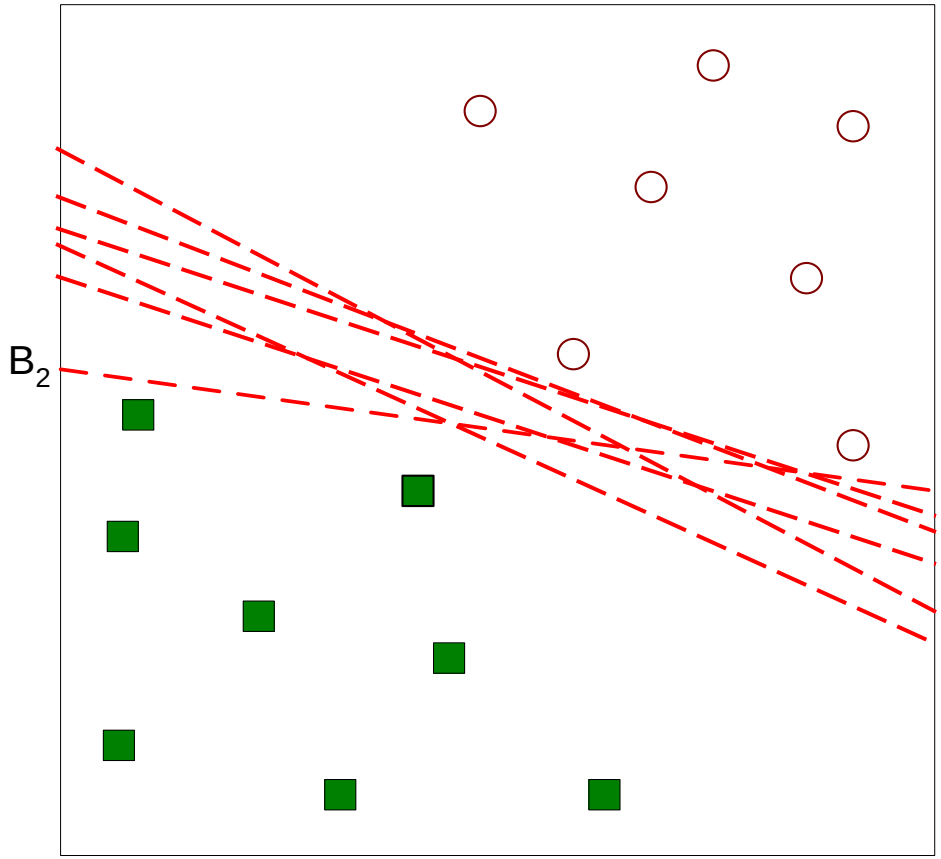
- One Possible Solution

Support Vector Machines



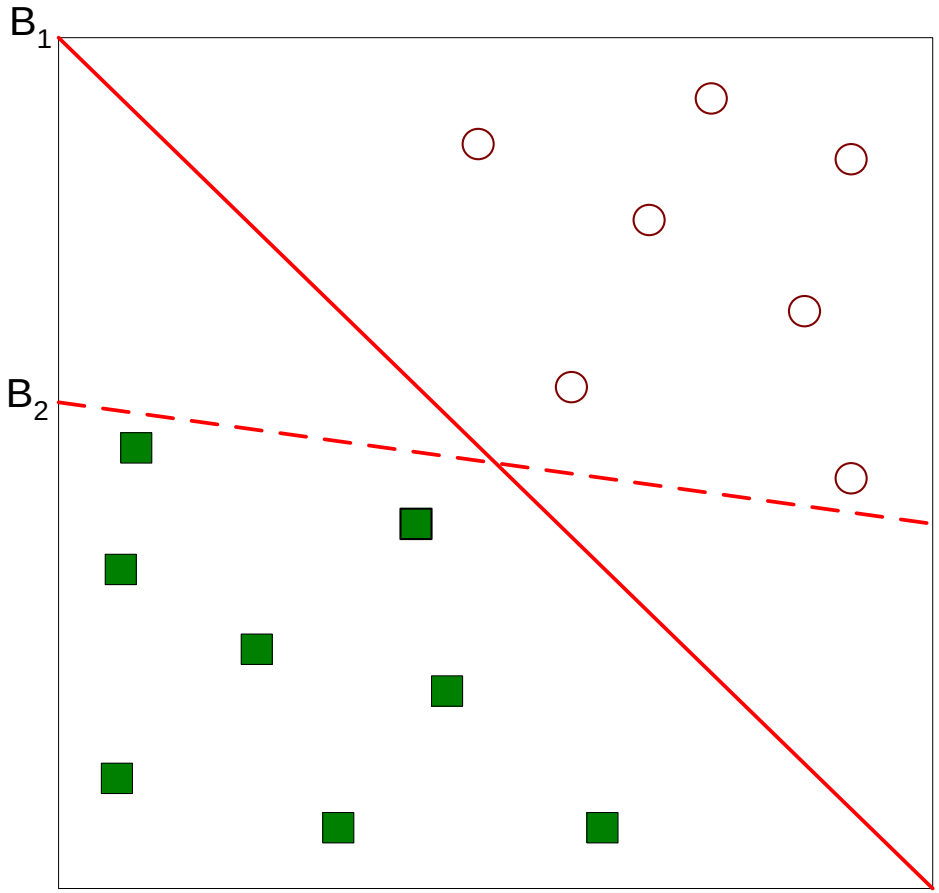
- Another possible solution

Support Vector Machines



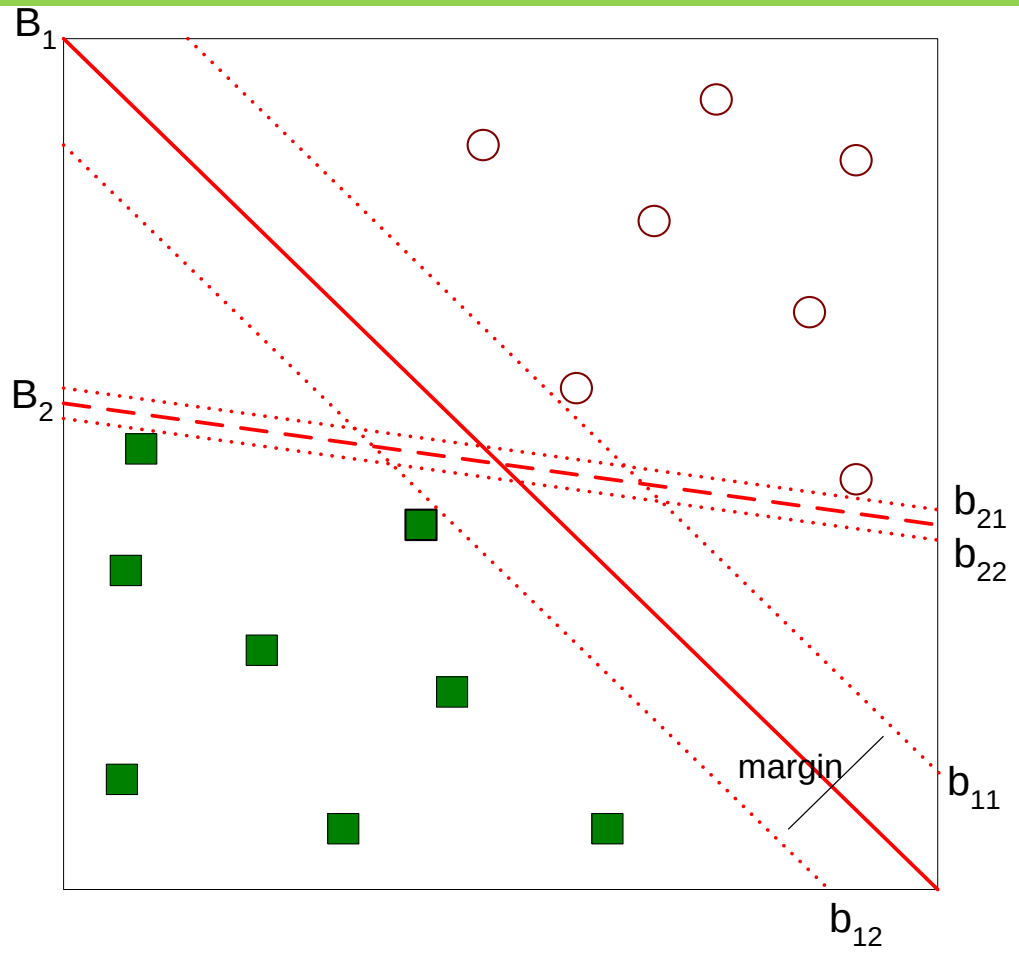
- Other possible solutions

Support Vector Machines



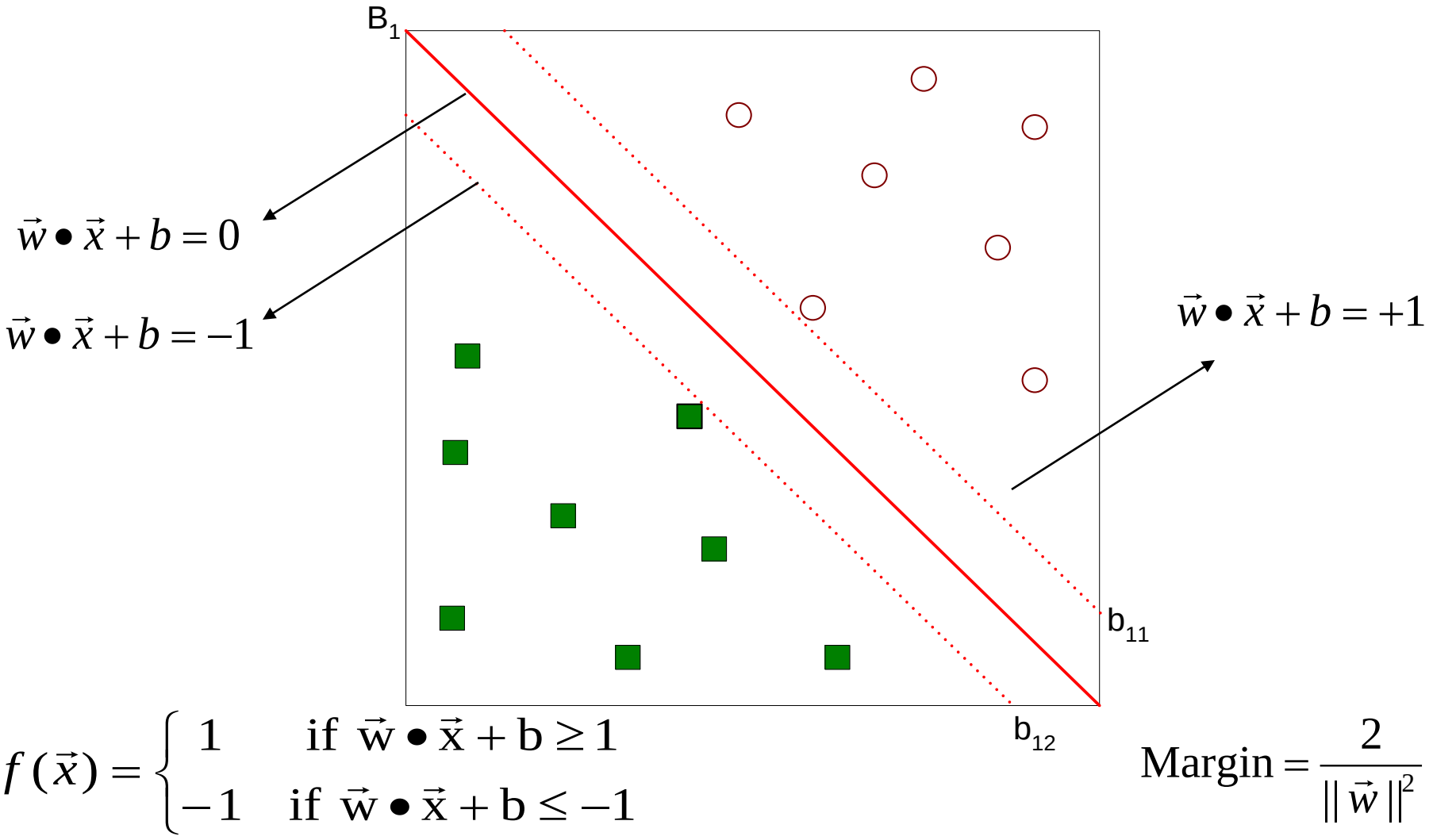
- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machines



- Find hyperplane maximizes the margin => B1 is better than B2

Support Vector Machines



$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\| \vec{w} \|^2}$$

Support Vector Machines

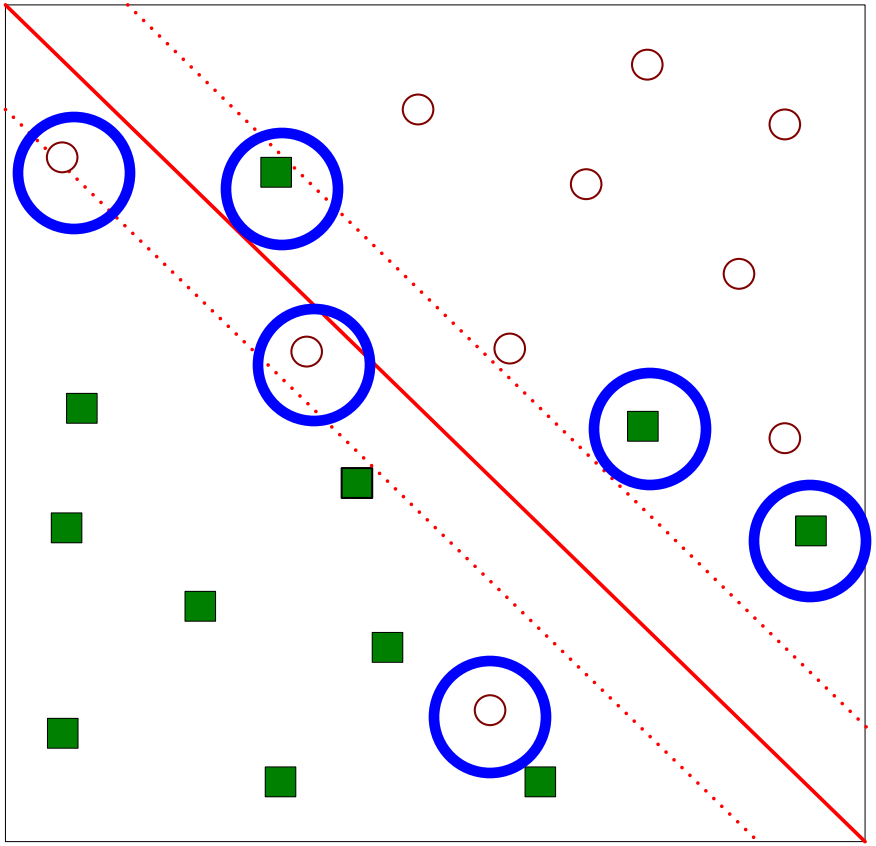
- We want to maximize: $\text{Margin} = \frac{2}{\| \vec{w} \|^2}$
  - Which is equivalent to minimizing: $L(w) = \frac{\| \vec{w} \|^2}{2}$
  - But subjected to the following constraints:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 \end{cases}$$

    - This is a constrained optimization problem
      - Numerical approaches to solve it (e.g., quadratic programming)

Support Vector Machines

- What if the problem is not linearly separable?



Support Vector Machines

- What if the problem is not linearly separable?
  - Introduce slack variables
    - Need to minimize:

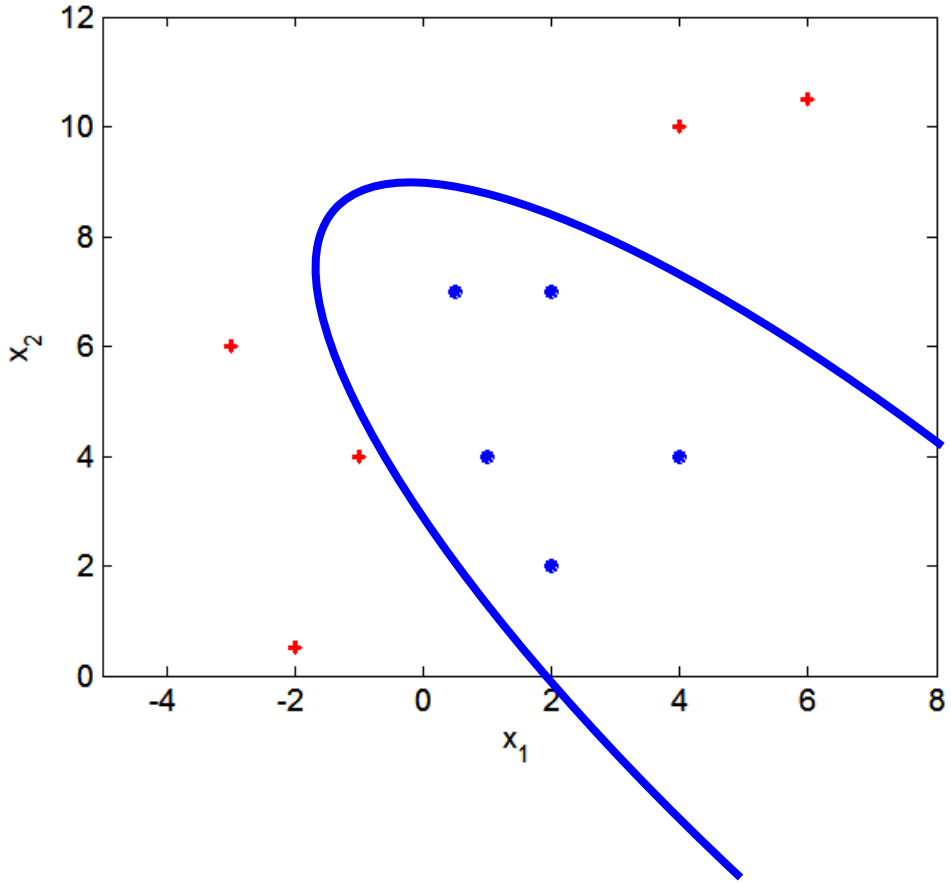
$$L(w) = \frac{\| \vec{w} \|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i^k \right)$$

    - Subject to:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 - \xi_i \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 + \xi_i \end{cases}$$

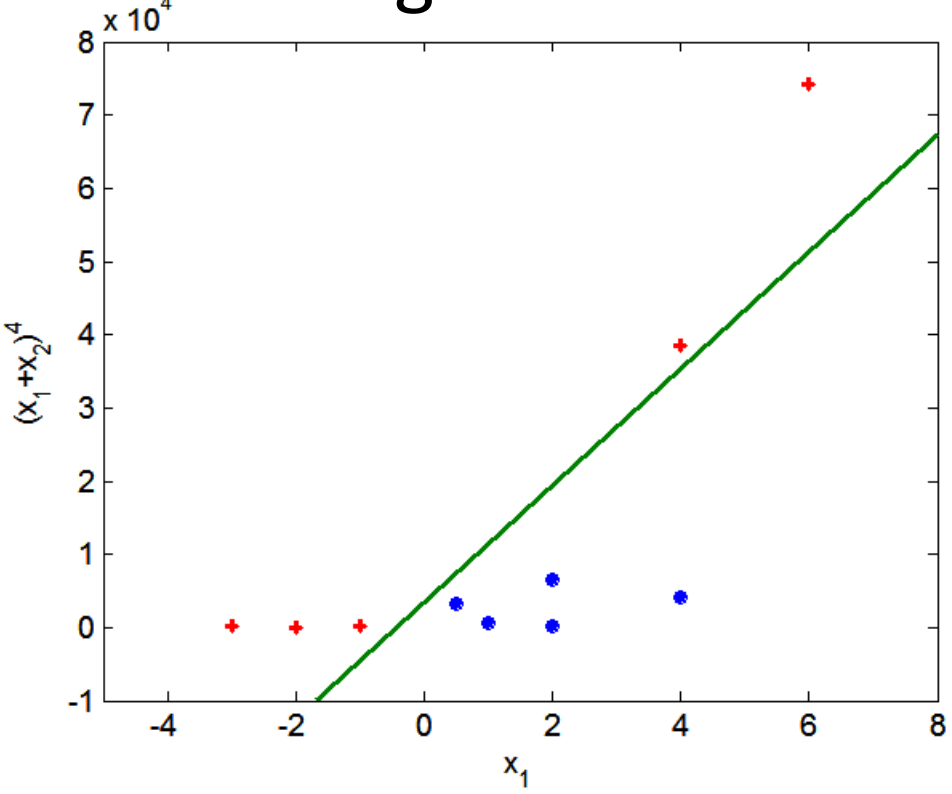
Nonlinear Support Vector Machines

- What if decision boundary is not linear?



Nonlinear Support Vector Machines

- Transform data into higher dimensional space



Kernel Machines

- Preprocess input $\boldsymbol{x}$ by basis functions

$$\boldsymbol{z} = \boldsymbol{\varphi}(\boldsymbol{x}) \qquad g(\boldsymbol{z})=\boldsymbol{w}^T\boldsymbol{z}$$

$$g(\boldsymbol{x})=\boldsymbol{w}^T\boldsymbol{\varphi}(\boldsymbol{x})$$

- The SVM solution

$$\boldsymbol{w} = \sum_t \alpha^t r^t \boldsymbol{z}^t = \sum_t \alpha^t r^t \boldsymbol{\varphi}\left(\boldsymbol{x}^t\right)$$

$$g\left(\boldsymbol{x}\right) = \boldsymbol{w}^T \boldsymbol{\varphi}\left(\boldsymbol{x}\right) = \sum_t \alpha^t r^t \boldsymbol{\varphi}\left(\boldsymbol{x}^t\right)^T \boldsymbol{\varphi}\left(\boldsymbol{x}\right)$$

$$g\left(\boldsymbol{x}\right) = \sum_t \alpha^t r^t K\left(\boldsymbol{x}^t, \boldsymbol{x}\right)$$

Kernel Functions

- Polynomials of degree q: $K(\boldsymbol{x}^t, \boldsymbol{x}) = (\boldsymbol{x}^T \boldsymbol{x}^t + 1)^q$

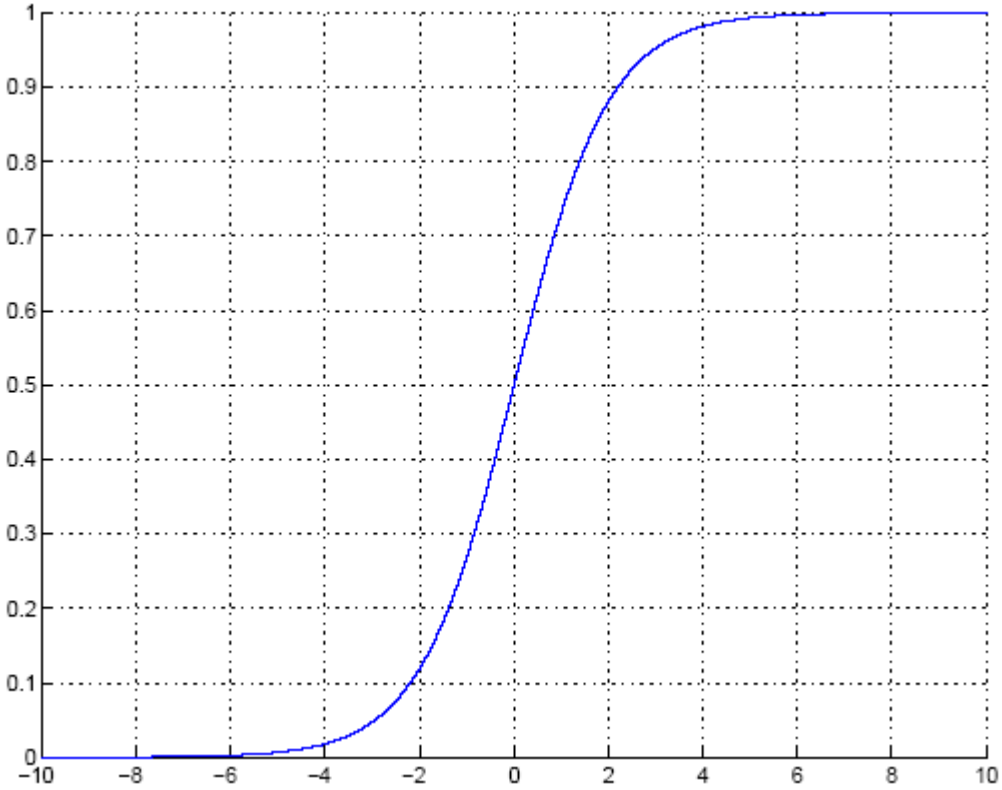
$$K(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}^T \boldsymbol{y} + 1)^2$$
$$= (x_1 y_1 + x_2 y_2 + 1)^2$$
$$= 1 + 2x_1 y_1 + 2x_2 y_2 + 2x_1 x_2 y_1 y_2 + x_1^2 y_1^2 + x_2^2 y_2^2$$
$$\phi(\boldsymbol{x}) = \left[1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1 x_2, x_1^2, x_2^2\right]^T$$

- Radial-basis functions: $K(\boldsymbol{x}^t, \boldsymbol{x}) = \exp\left[-\frac{\|\boldsymbol{x}^t - \boldsymbol{x}\|^2}{\sigma^2}\right]$

- Sigmoidal functions: $K(\boldsymbol{x}^t, \boldsymbol{x}) = \tanh(2\boldsymbol{x}^T \boldsymbol{x}^t + 1)$

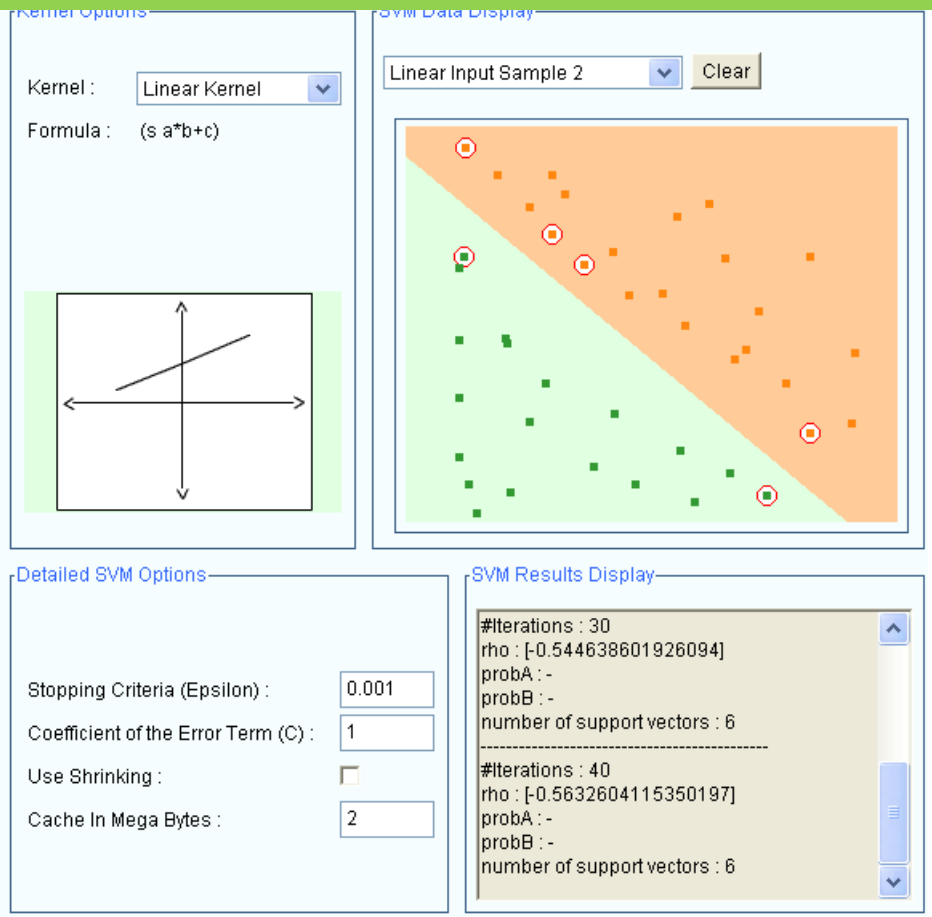
(Cherkassky and Mulier, 1998)

Sigmoid (Logistic) Function



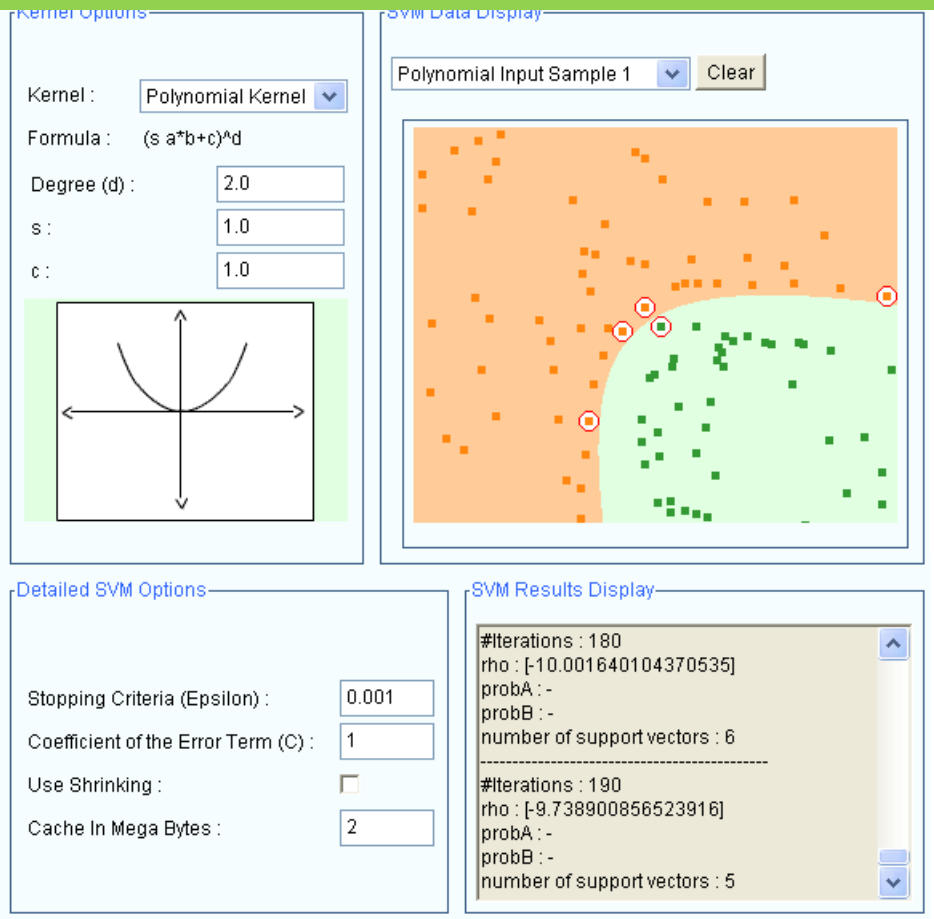
1. Calculate $g(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + w_0$ and choose C_1 if $g(\boldsymbol{x}) > 0$, or
2. Calculate $y = \text{sigmoid}(\boldsymbol{w}^T \boldsymbol{x} + w_0)$ and choose C_1 if $y > 0.5$

Linear Kernel



http://www.eee.metu.edu.tr/~alatan/Courses/Demo/AppletSVM.html

Polynomial Kernel

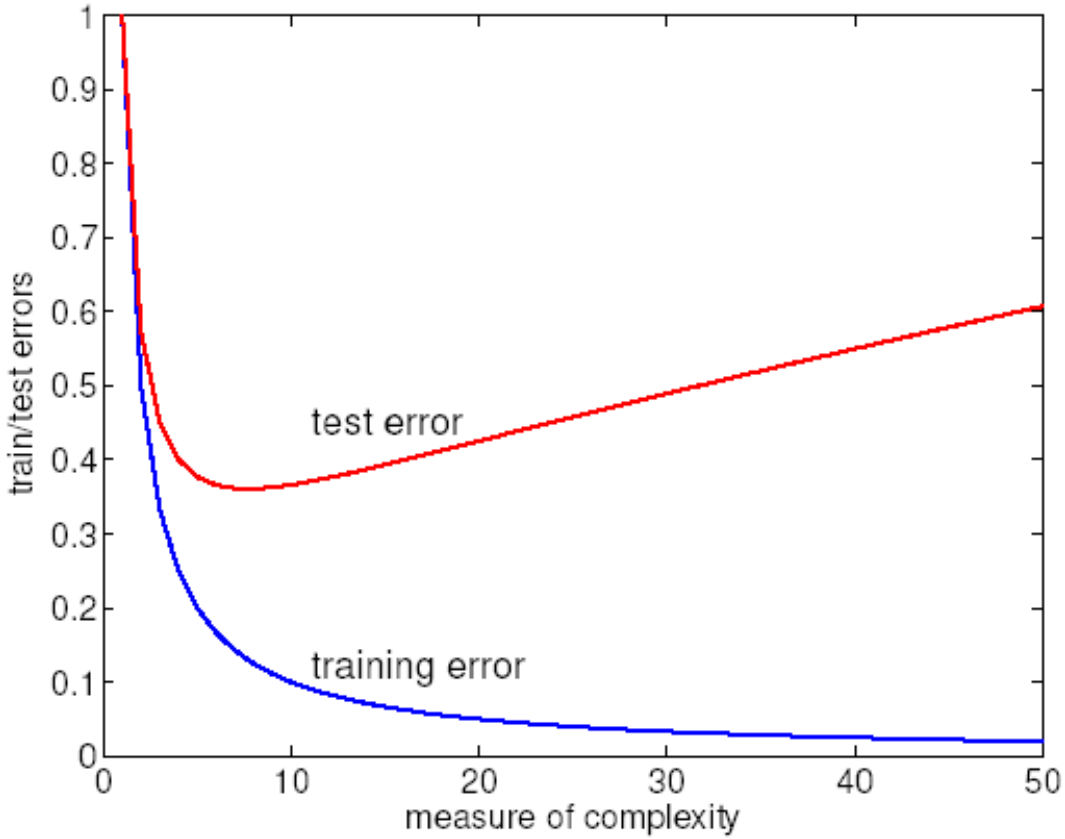


http://www.eee.metu.edu.tr/~alatan/Courses/Demo/AppletSVM.html

Radial Basis Function

http://www.eee.metu.edu.tr/~alatan/Courses/Demo/AppletSVM.html

Why care about "complexity"?



- We need a quantitative measure of complexity in order to b
  able to relate the training error (which we can observe) an
  the test error (that we'd like to optimize)

Insight into Kernels

There are four basic kernels that are currently in use. The linear kernel in which K is just the identity matrix and the result is just the regular inner product. As a summary, the four most common kernels with parameters γ, r, and d are given as

- Linear Kernel: $K(x, z) = x^T z$
- Polynomial Kernel: $K(x, z) = (\gamma x^T z + r)^d, \gamma > 0$
- Radial Basis Function Kernel: $K(x, z) = \exp(-\gamma ||x - z||^2), \gamma > 0$
- Sigmoid: $K(x, z) = \tanh(\gamma x^T z + r)$

The Gaussian Kernel is a special case of the Radial Basis Function (RBF) kernel. The Gaussian Kernel is given as

$$K(x, z) = \exp\left(\frac{-||x - z||^2}{2\sigma^2}\right)$$

Use Cross Validation to determine gamma, r, d, and C (the cost of constraint violation for soft margin)

http://patriciahoffmanphd.com/resources/papers/SVM/SupportVectorJune3.pdf

Complexity
Alternative to Cross Validation

- "Complexity" is a measure of a set of classifiers, not any

specific (fixed) classifier

- Many possible measures
  - degrees of freedom
  - description length
  - Vapnik-Chervonenkis (VC) dimension
  - etc.

Expected and Empirical error

$$\hat{\mathcal{E}}_n(i) = \frac{1}{n}\sum_{t=1}^{n} \overbrace{\text{Loss}(y_t, h_i(\mathbf{x}_t))}^{=0,1} = \text{empirical error of } h_i(\mathbf{x})$$

$$\mathcal{E}(i) = E_{(\mathbf{x},y)\sim P}\{\text{Loss}(y, h_i(\mathbf{x}))\} = \text{expected error of } h_i(\mathbf{x})$$

Learning and VC-dimension

- Let d_{VC} be the VC-dimension of our set of classifiers F.

Theorem: With probability at least $1-\delta$ over the choice of the training set, for all $h \in F$

$$\mathcal{E}(h) \leq \hat{\mathcal{E}}_n(h) + \epsilon(n, d_{VC}, \delta)$$

where

$$\epsilon(n, d_{VC}, \delta) = \sqrt{\frac{d_{VC}(\log(2n/d_{VC}) + 1) + \log(1/(4\delta))}{n}}$$

Model selection

- We try to find the model with the best balance of complexity and the fit to the training data
- Ideally, we would select a model from a nested sequence of models of increasing complexity (VC-dimension)

  Model 1 d_1
  Model 2 d_2
  Model 3 d_3

  where $d_1 \leq d_2 \leq d_3 \leq \ldots$

- The model selection criterion is: find the model class that achieves the lowest upper *bound* on the expected loss

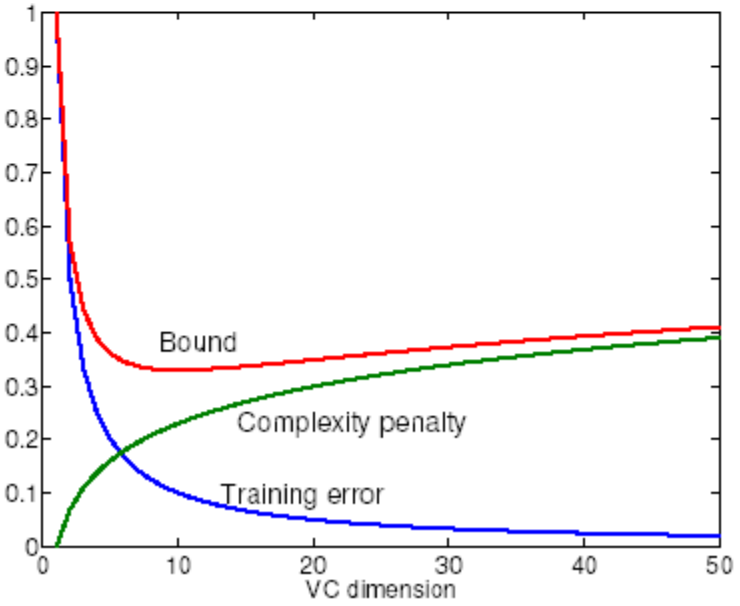
  $$\text{Expected error} \leq \text{Training error} + \text{Complexity penalty}$$

Structural risk minimization cont'd

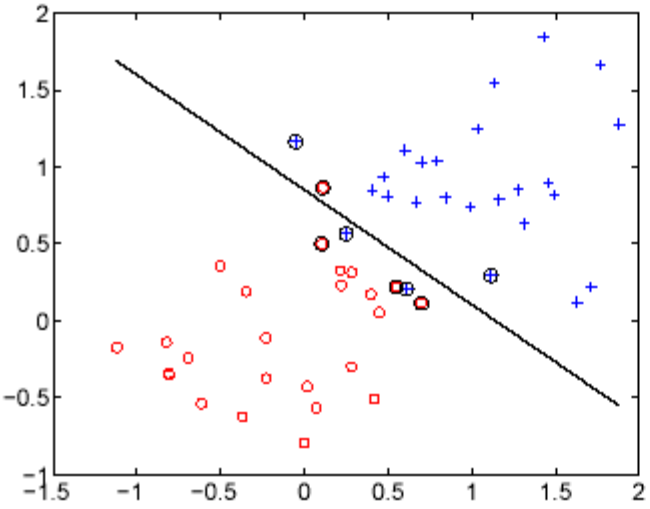
- We choose the model class F_i that minimizes the upper bound on the expected error:

$$\mathcal{E}(\hat{h}_i) \leq \hat{\mathcal{E}}_n(\hat{h}_i) + \sqrt{\frac{d_i(\log(2n/d_i)+1) + \log(1/(4\delta))}{n}}$$

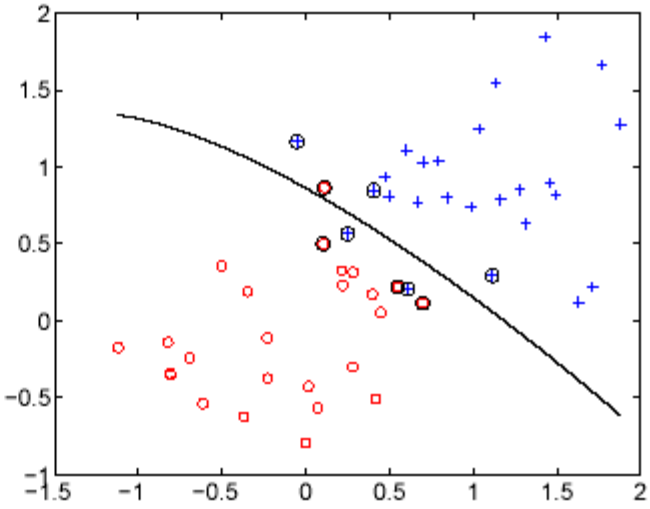
where $\hat{h}_i$ is the best classifier from F_i selected on the basis of the training set.



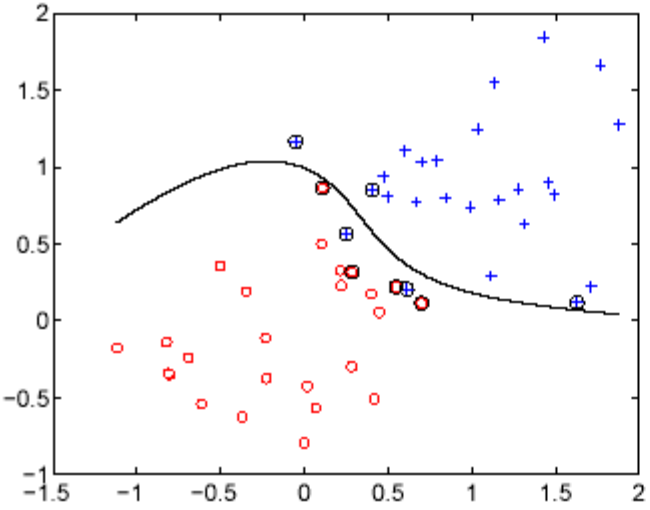
Structural risk minimization: example



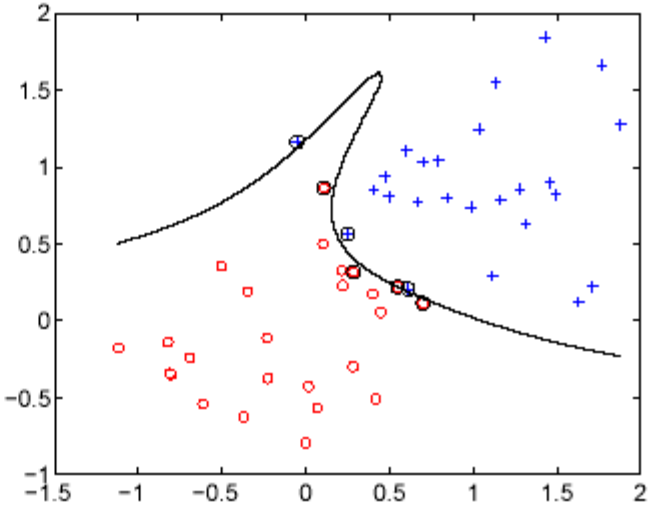
linear



2^{nd} order polynomial



4^{th} order polynomial



8^{th} order polynomial

Structural risk minimization: example cont'd

- Number of training examples $n = 50$, confidence parameter $\delta = 0.05$.

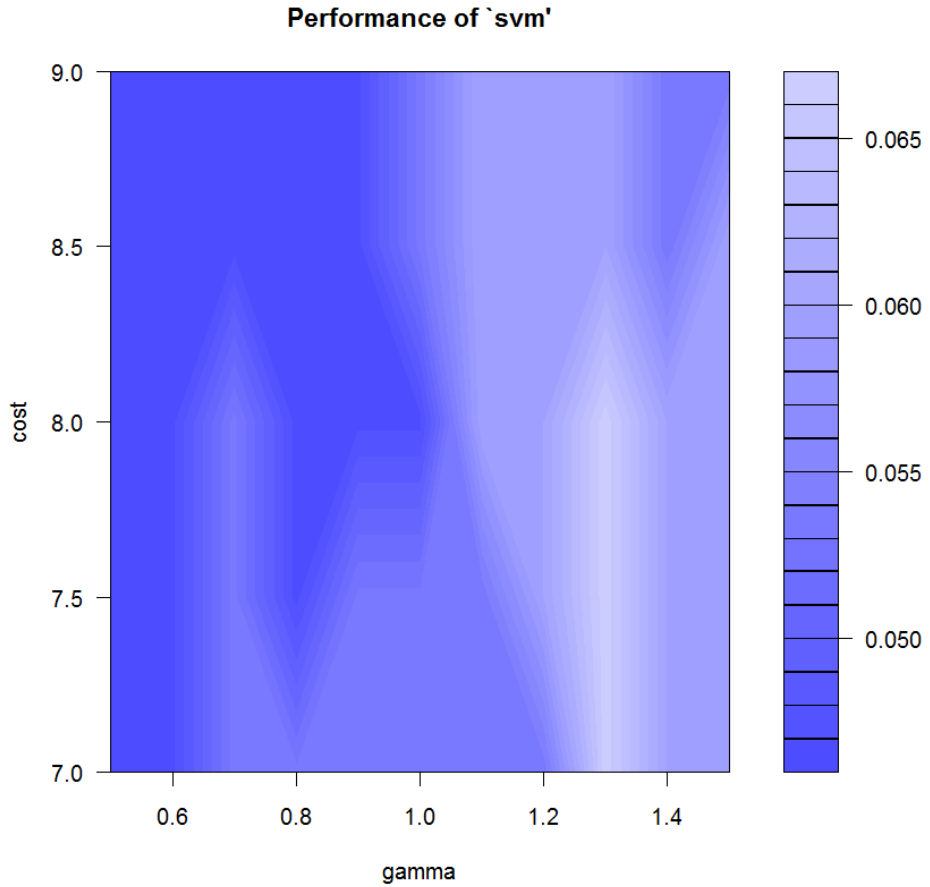
Model	d_{VC}	Empirical fit	$\epsilon(n, d_{VC}, \delta)$
1^{st} order	3	0.06	0.5501
2^{nd} order	6	0.06	0.6999
4^{th} order	15	0.04	0.9494
8^{th} order	45	0.02	1.2849

- Structural risk minimization would select the simplest (linear) model in this case.

Example svm1.r

- svm1.r - Example of SVM
  - classification plots, tuning plots, confusion tables
  - Classification using SVM on
    - Iris Data Set, Glass Data Set
  - Regression using SVM on
    - Toy data set

svm1.r Iris Tuning Plot (heat map)



gamma	cost	error	dispersion
0.5	7.0	0.04666667	0.04499657
1.0	8.0	0.04666667	0.03220306

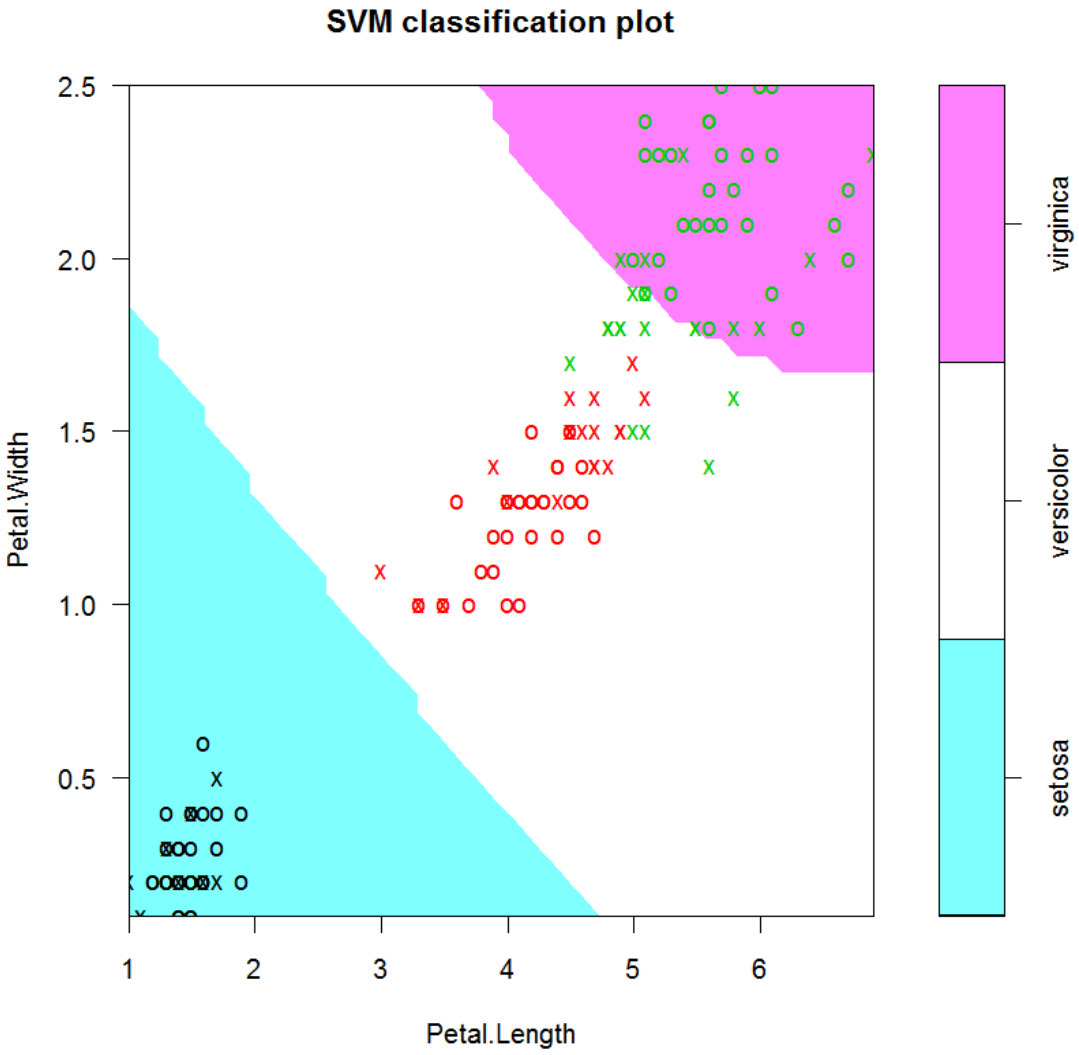
Tune SVM via Cross Validation

```
obj <- tune(svm, Species~., data = iris,
ranges = list(gamma = seq(.5,1.5,0.1), cost =
   seq(7,9,0.5)),
tunecontrol = tune.control(sampling = "cross")
)
```

- gamma = parameter for Radial Basis Function
- cost = cost of constraint violation (soft margin)

SVM Classification Plot

```
# visualize (classes by color)
plot(model, iris,
Petal.Width ~ Petal.Length,
slice = list(Sepal.Width = 2,
Sepal.Length = 3))

#support vectors  = “x“
# data points       = "o"
```

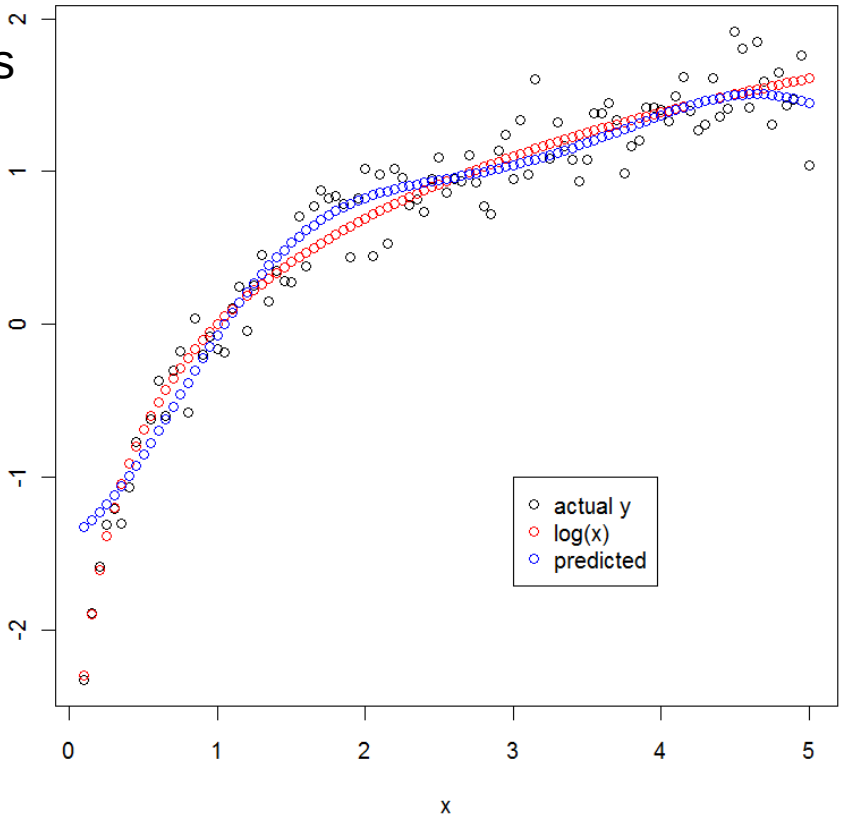


Results of SVM for Iris Data

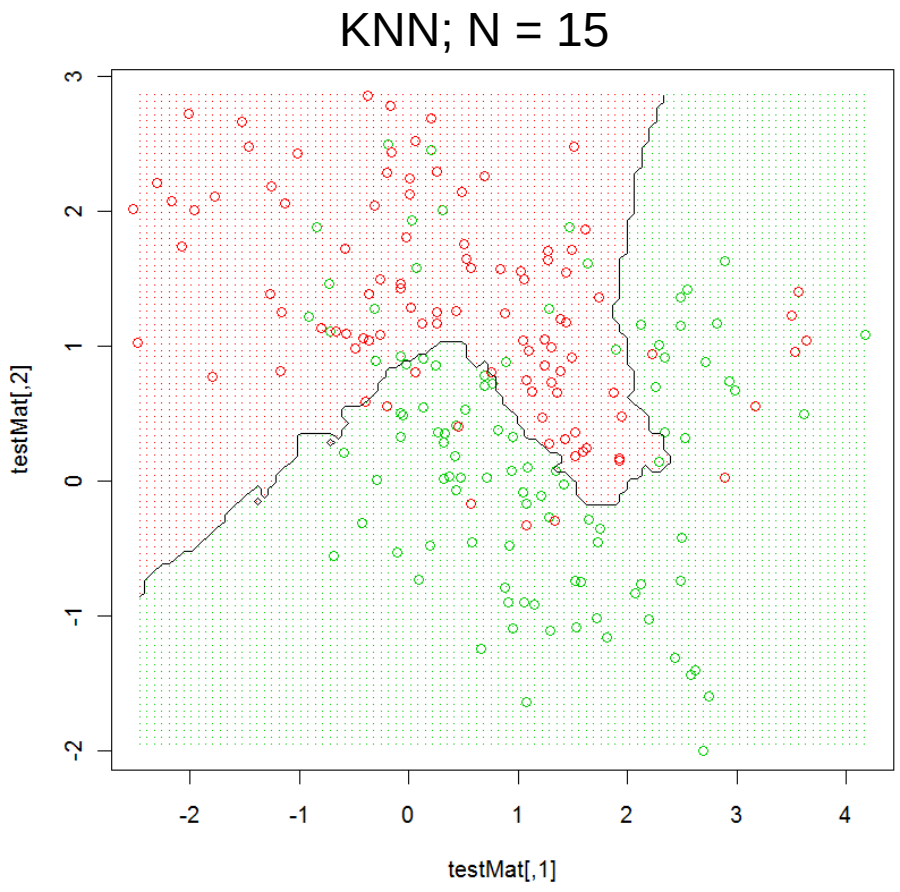
```
model <- svm(Species ~ ., data = iris, gamma = 1.0, cost = 8)

# test with train data
pred <- predict(model, x)
# Check accuracy:
table(pred, y)

#
#gamma = 1.0, cost = 8
#                y
#pred        setosa versicolor virginica
#setosa         50          0         0
#versicolor      0         49         0
#virginica       0          1        50
```

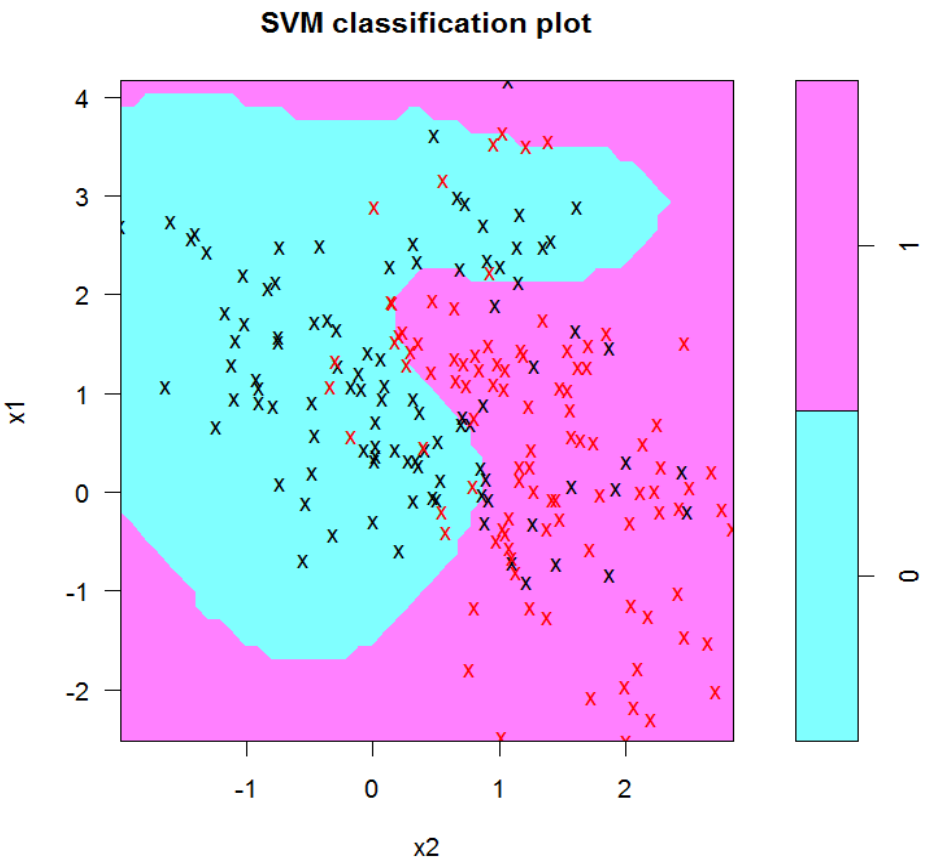
Regression using SVM

```
## try regression mode on two dimensions
# create data
x <- seq(0.1, 5, by = 0.05)
y <- log(x) + rnorm(x, sd = 0.2)

# estimate model and predict input values
m <- svm(x, y)
new <- predict(m, x)

# visualize
plot(x, y, col = 1)
points(x, log(x), col = 2)
points(x, new, col = 4)
legend(3, -1, c("actual y", "log(x)",
"predicted"), col = c(1,2,4), pch=1)
```

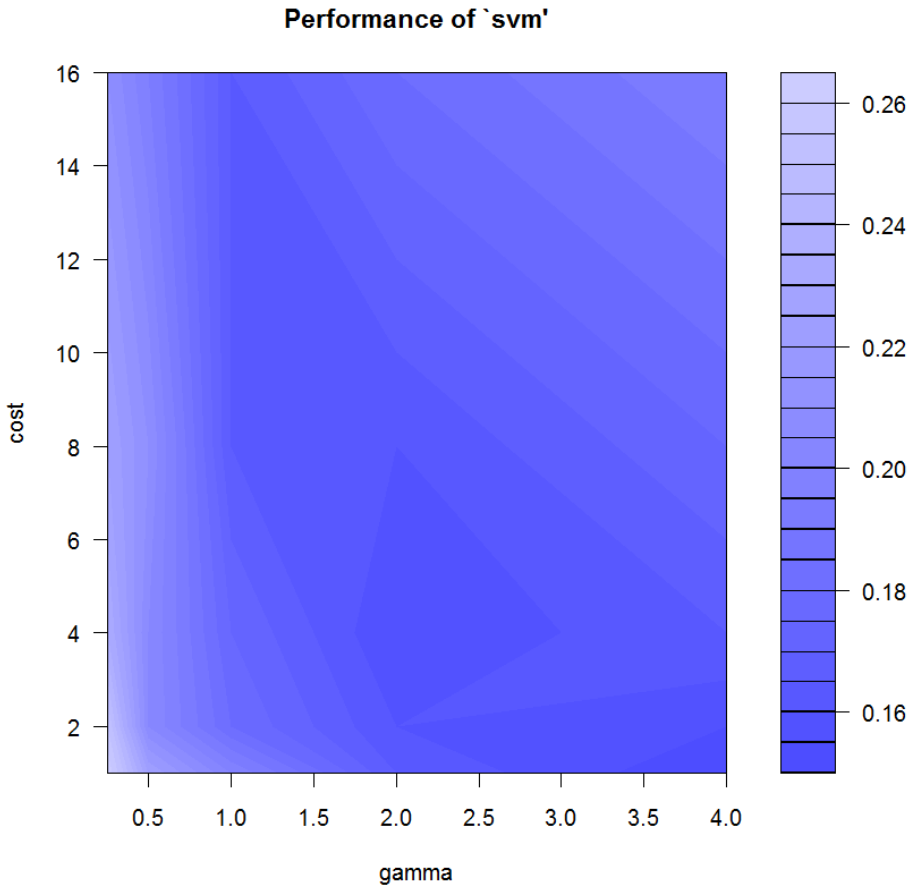


svmExamp.r
KNN vs. SVM (RBF) on MixSim Data Set





Tuning for SVM RBF



SVM Characteristics

- Maximizes Margins between Classifications
- Formulated as Convex Optimization Problem
- Parameter Selection for Kernel Function
  - Use Cross Validation
- Can be extended to multiclass
- Multi-valued categorical attributes can be coded by introducing binary variable for each attribute value (single, married, divorced)