Statistisches Data Mining (StDM)
Woche 10

Zurich University
of Applied Sciences

Oliver Dürr

Institut für Datenanalyse und Prozessdesign

Zürcher Hochschule für Angewandte Wissenschaften

oliver.duerr@zhaw.ch

Winterthur, 22 November 2016

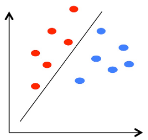
No laptops, no phones, no problems

Multitasking senkt Lerneffizienz:

- **Keine Laptops im Theorie-Unterricht Deckel zu oder fast zu (Sleep modus)**

Overview of classification (until the end to the semester)

Classifiers



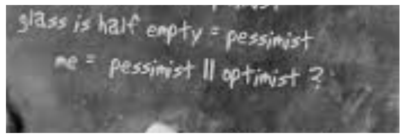
K-Nearest-Neighbors (KNN)
Logistic Regression
Linear discriminant analysis
Support Vector Machine (SVM)
Classification Trees
Neural networks NN
Deep Neural Networks (e.g. CNN, RNN)
…

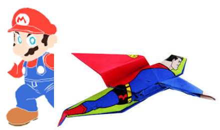
Evaluation

Cross validation
Performance measures
ROC Analysis / Lift Charts

Theoretical Guidance / General Ideas

Bayes Classifier
Bias Variance Trade off (Overfitting)



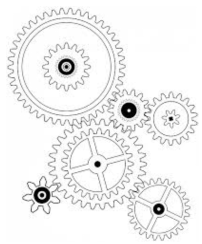


Combining classifiers

Bagging
Boosting
Random Forest

Feature Engineering

Feature Extraction
Feature Selection



SVM
Chapter 9 in ILSR

Note on notation in ISLR

- In ISLR they make an unusual distinction between Support Vector Classifier and Support Vector Machine (SVM).

- Here we call everything a SVM
  - Linear Separable Case
  - SVM with Penalty allowing misclassifications
  - SVM with Kernels

Support Vector Machine (SVM) - Basics

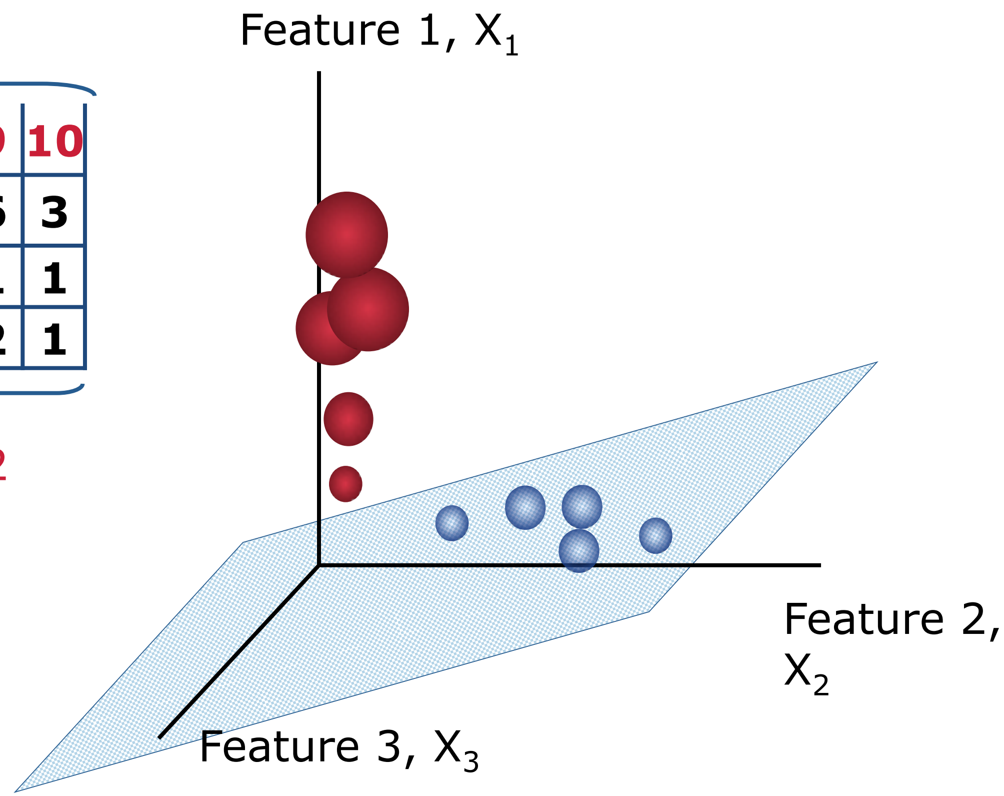
- Each observation ⇔ vector of values (p-Dimensional)
- SVM constructs a hyperplane to separate class members.

Observations N

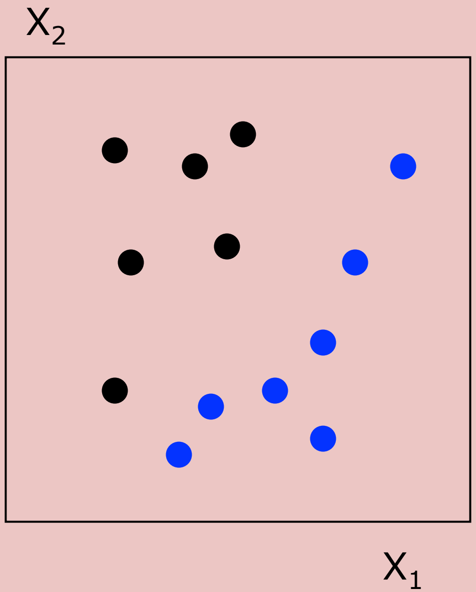
Features p

	1	2	3	4	5	6	7	8	9	10
1	2	1	3	4	4	5	7	6	6	3
2	4	2	6	8	10	2	1	2	1	1
3	3	1	5	6	6	2	1	2	2	1

Class 1 Class 2



Welche Ebene?

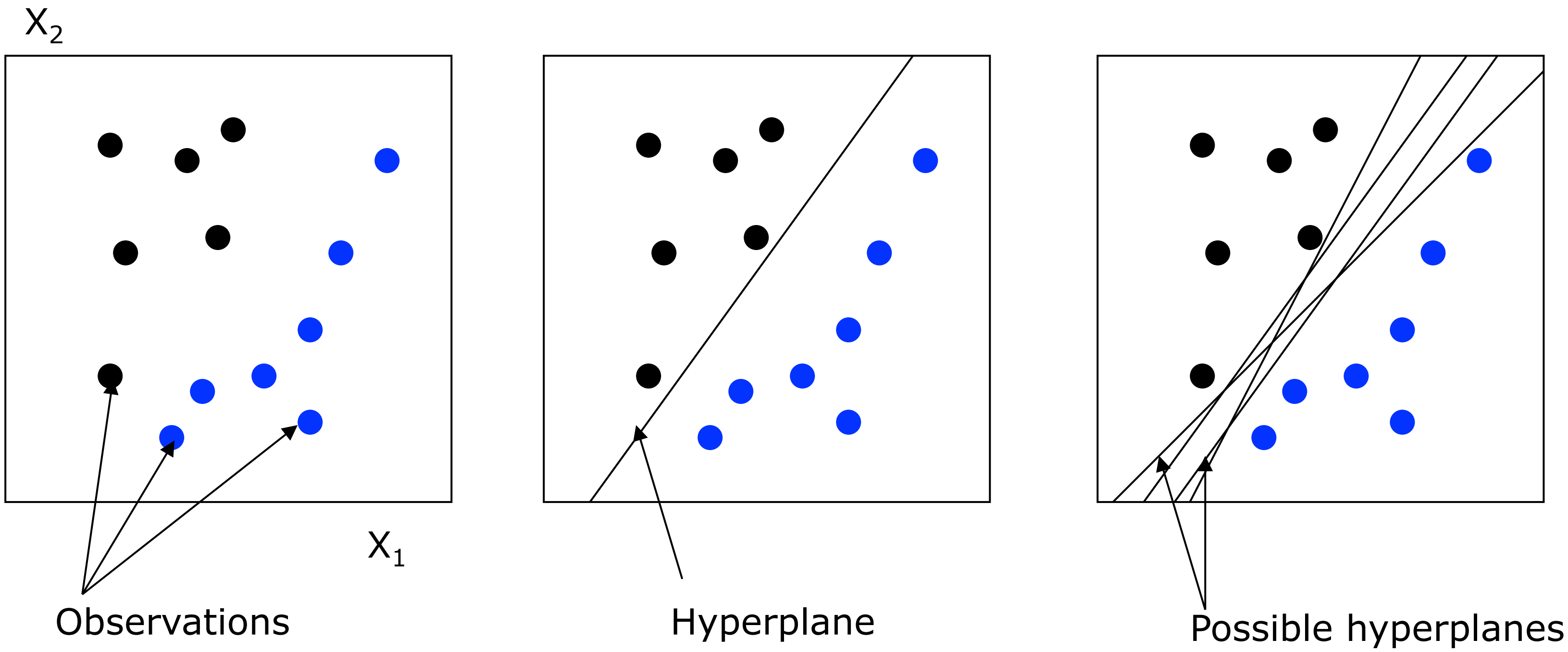


Zeichnen Sie eine Linie, die die beiden Klassen möglichst gut trennt.

Begründen Sie Ihre Wahl

Support Vector Machine - Hyperplanes

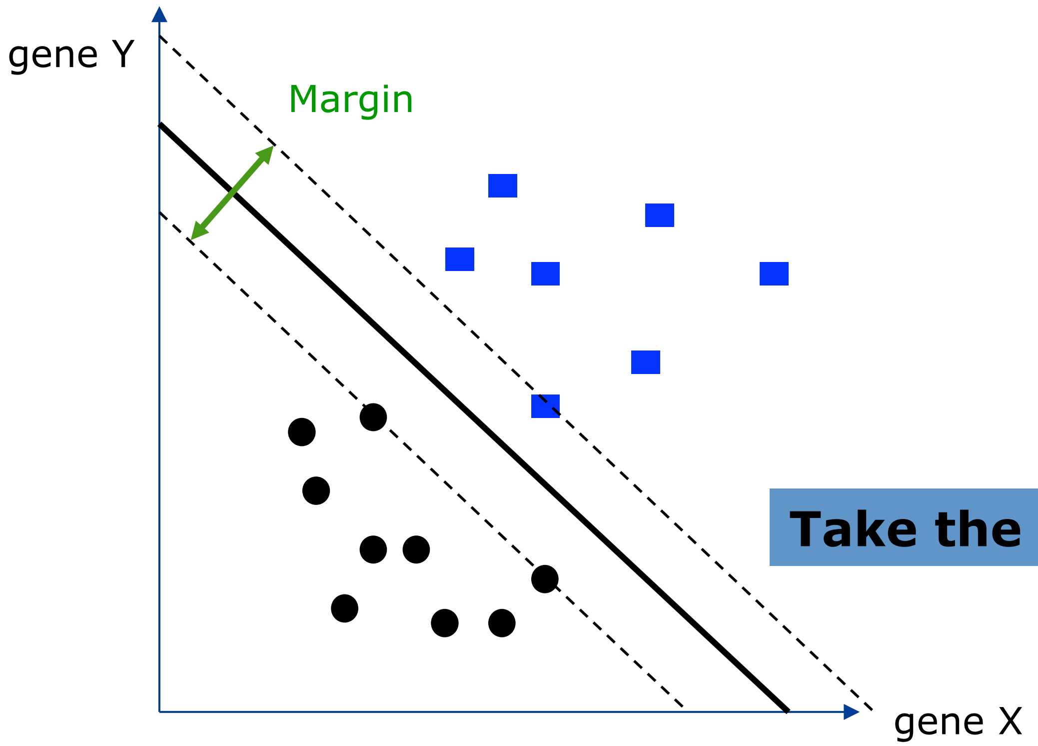
- Each column vector can be viewed as a point in an p-dimensional space (p = number of features).
- A linear binary classifier constructs a hyperplane separating class members from non-members in this space.



...which one?

Support Vector Machine - Maximum Margin Hyperplane

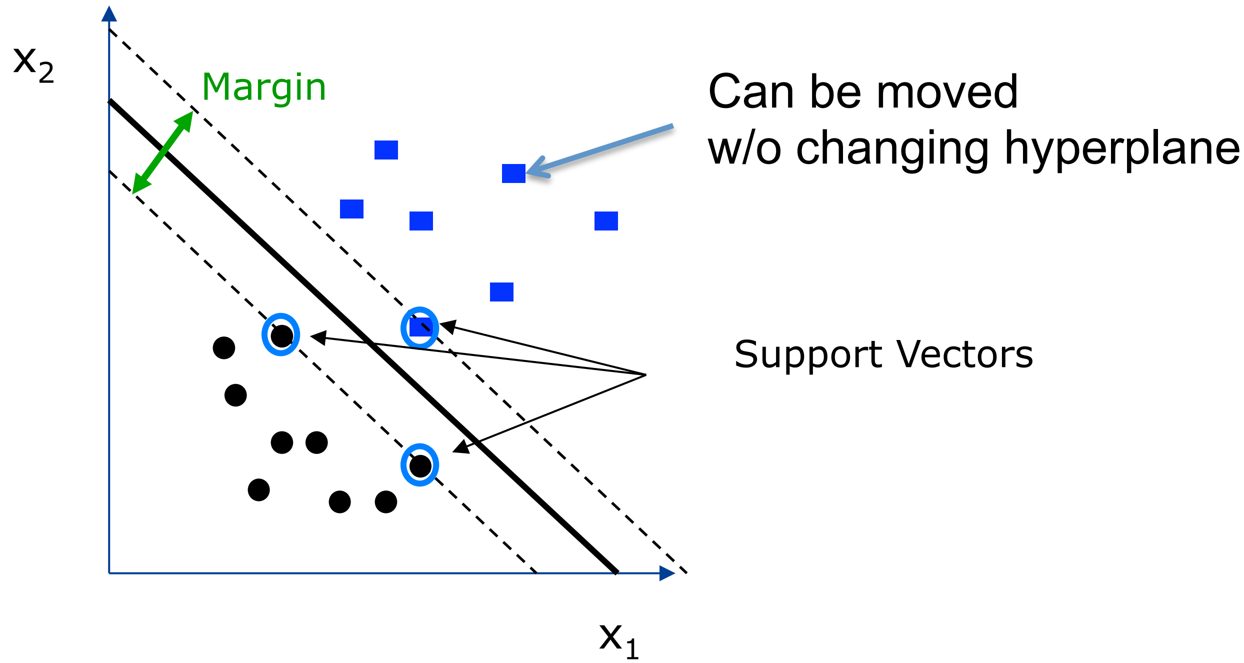
- SVM choose a specific hyperplane among the many that can separate the data, namely the *maximum margin hyperplane*, which maximizes the distance from the hyperplane to the closest training point.
- The maximum margin hyperplane can be represented as a linear combination of (some) training points.



Take the fattest margin!

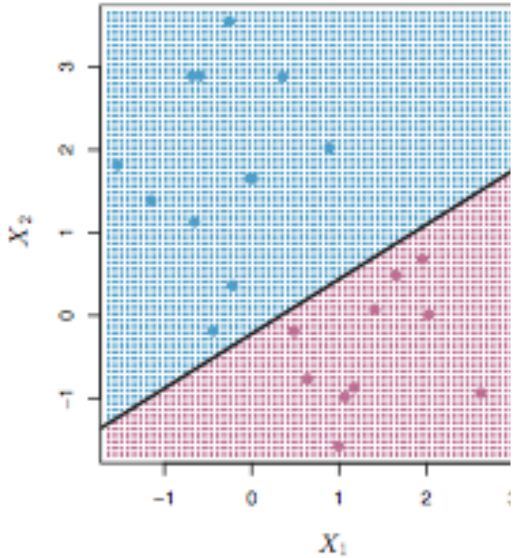
SVM - Support Vectors

- Training examples that lie far away from the hyperplane do not participate in its specification.
- Training examples that lie closest to the decision boundary between the two classes determine the hyperplane.
- These training examples are called the **support vectors**, since removing them would change the location of the separating hyperplane. They determine the classifier.



Mathematical Definition and Optimization (just sketch)

Formal:: Definition of a hyperplane



We assume that classes are separable

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$ Definition of Hyperplane

Separating hyperplane for classes coded as y=±1

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} > 0 \text{ if } y_i = 1,$$

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} < 0 \text{ if } y_i = -1.$$

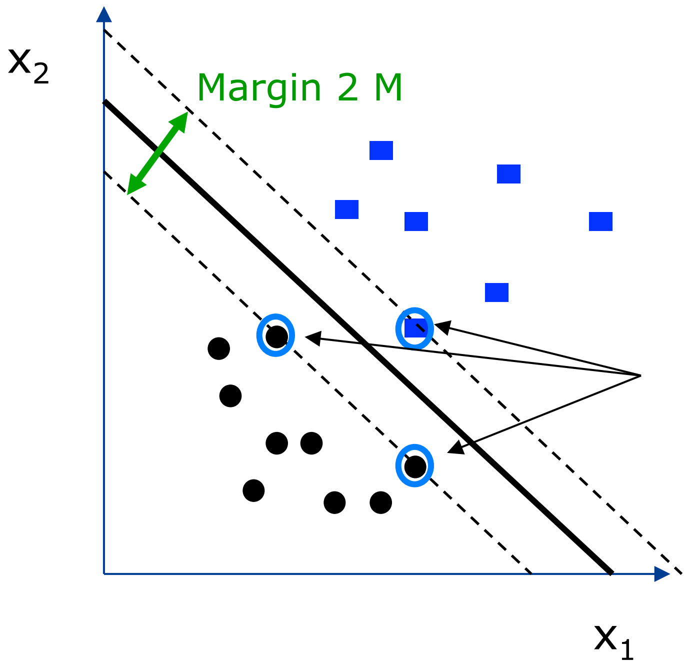
Combining

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) > 0$$

Note that this is only up to a constant (multiplication does not change anything)
➔Fix beta for components 1 to p:

$$\sum_{j=1}^{p} \beta_j^2 = 1,$$ β (for j=1,…,p) is a normal vector

Formal:: Definition of optimization problem



Support Vectors, have distance M to hyperplane

Intuitive Optimization

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{maximize}}\, M$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \;\; \forall\, i = 1, \ldots, n.$$

All vectors have at least distance M.
Support Vectors have = M.

Formal:: Reformulating the optimization problem

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{maximize}}\, M$$

Intuitive Optimization

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \;\; \forall\, i = 1, \ldots, n.$$

Can be reformulated using Lagrange multipliers to

$$L_D \;=\; \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

Technical Optimization

Only the inner product between the vectors of observations enters.

Opens the door to the kernel trick (see below)

$$\text{subject to } \alpha_i \geq 0 \text{ and } \sum_{i=1}^{N} \alpha_i y_i = 0.$$

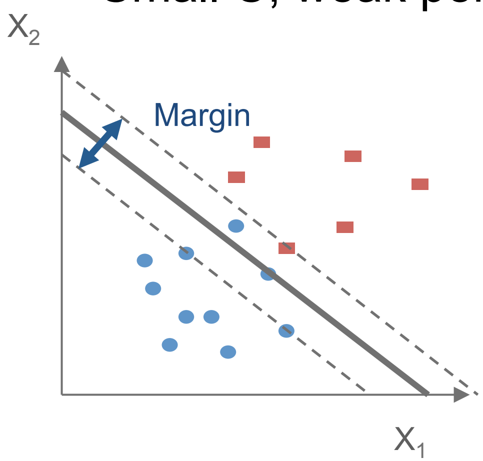
once we have calculated the α's we can calculate β (1,..,p) via

$$\beta \;=\; \sum_{i=1}^{N} \alpha_i y_i x_i$$

SVM – Penalty (general idea)

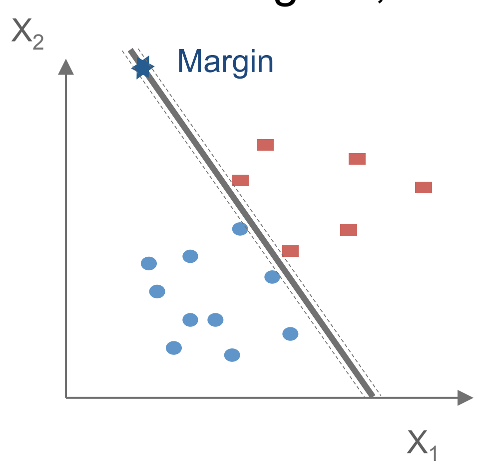
- SVM may not be able to find any separating hyperplane at all, because the data contains untypical or mislabelled experiments.
- The problem can be addressed by using a *soft margin* that accepts some misclassifications of the training examples. The number of misclassifications is triggered by a *penalty factor C*.
- Sometimes a larger margin is worth having some misclassified observations

Small C, weak penalty



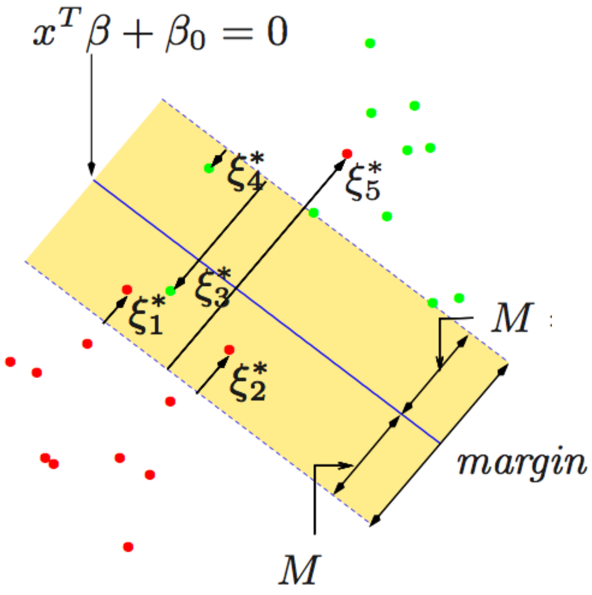
Low penalty:
high # of misclassified experiments

Large C, strong penalty



High penalty:
Low # of misclassified experiments

Formal:: SVM – Penalty



Introduction of slack variables ξ_i, for all observations (measured in units of M).

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \quad M$$

Intuitive Optimization

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i)$$

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

Finally this leads to the following equivalent optimization of L_D with constrains on α.

Technical Optimization (“dual form”)

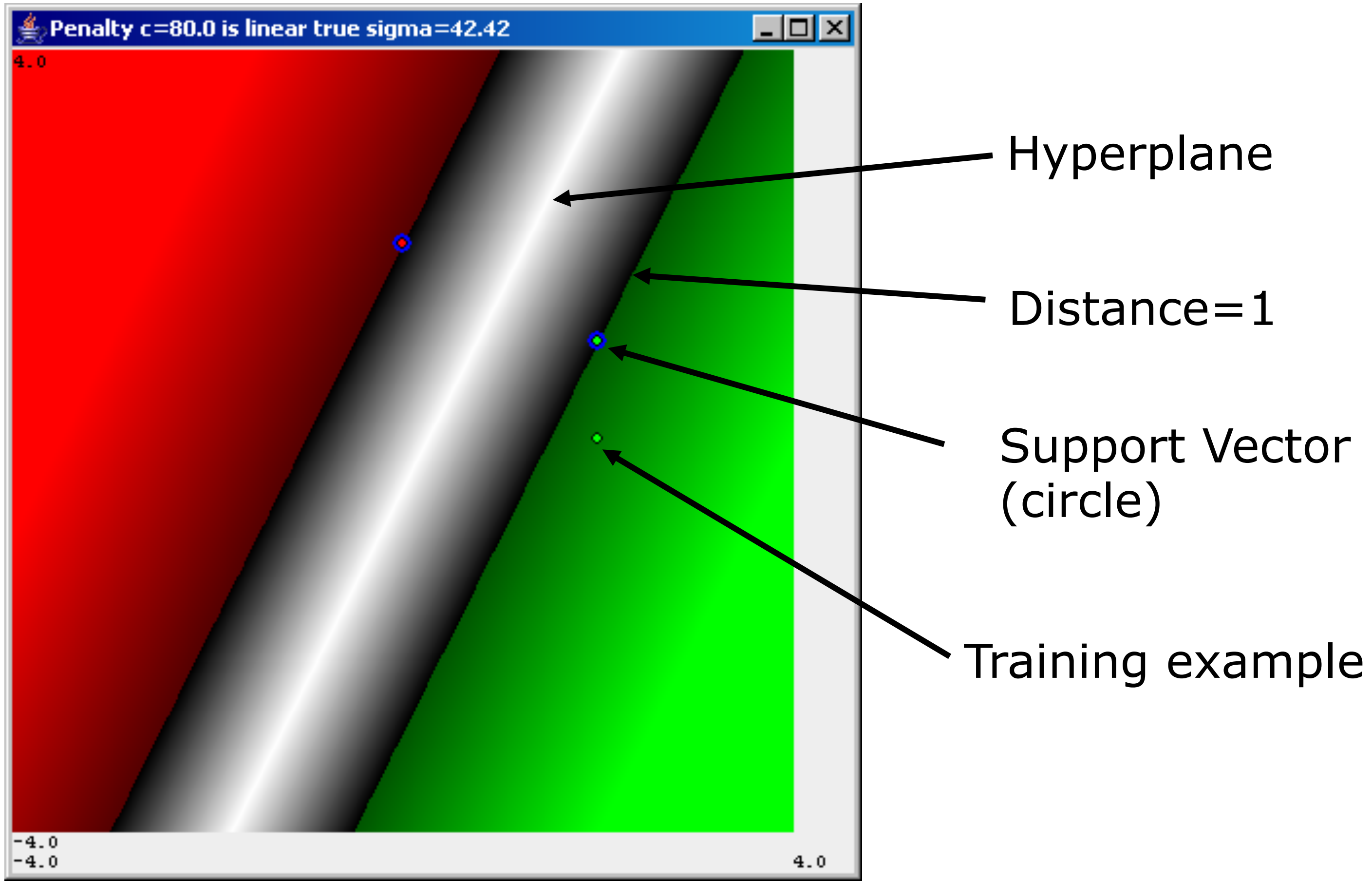
$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'} \qquad 0 \leq \alpha_i \leq C \qquad \sum_{i=1}^{N} \alpha_i y_i = 0$$

From α, β is obtained

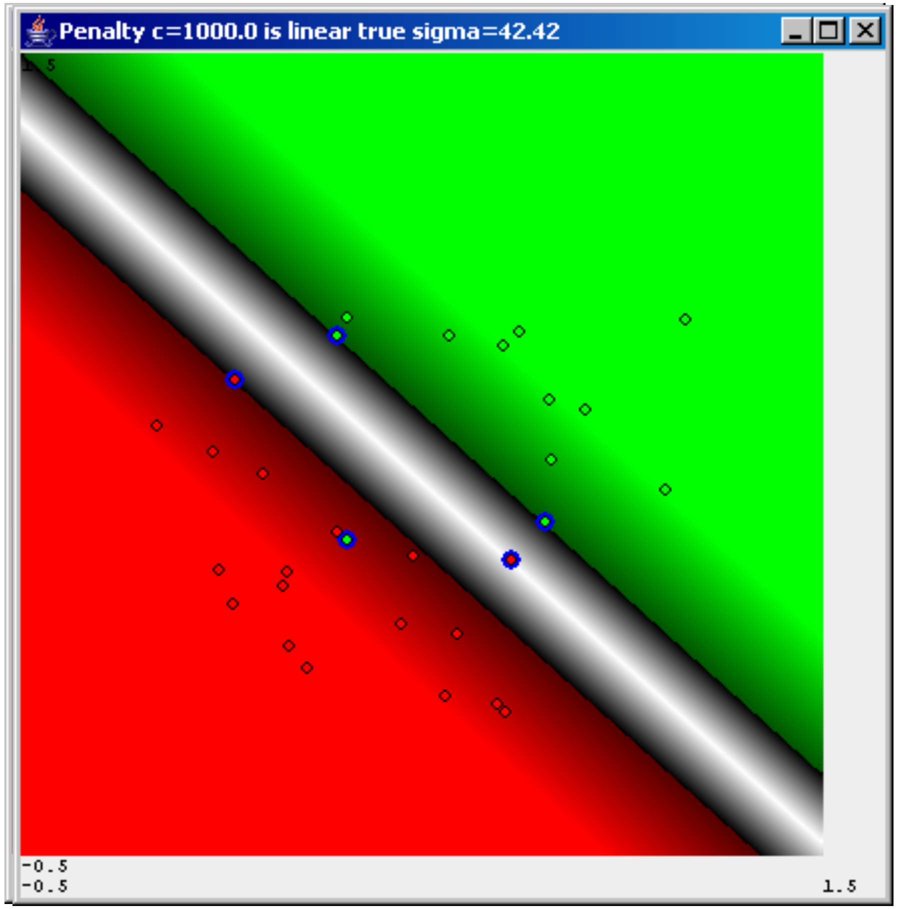
Again only inner product!

Source: Elements of statististical learning

Visualization of the parameter influence
Linear case effect of C



SVM – From low and high penalty



Very low c, nearly
no penalty for
misclassifications.
Big margin.
Let's increase c
and see what happens.

Increase of c leads to convergence to a stable solution.

Why do we want a large margin?

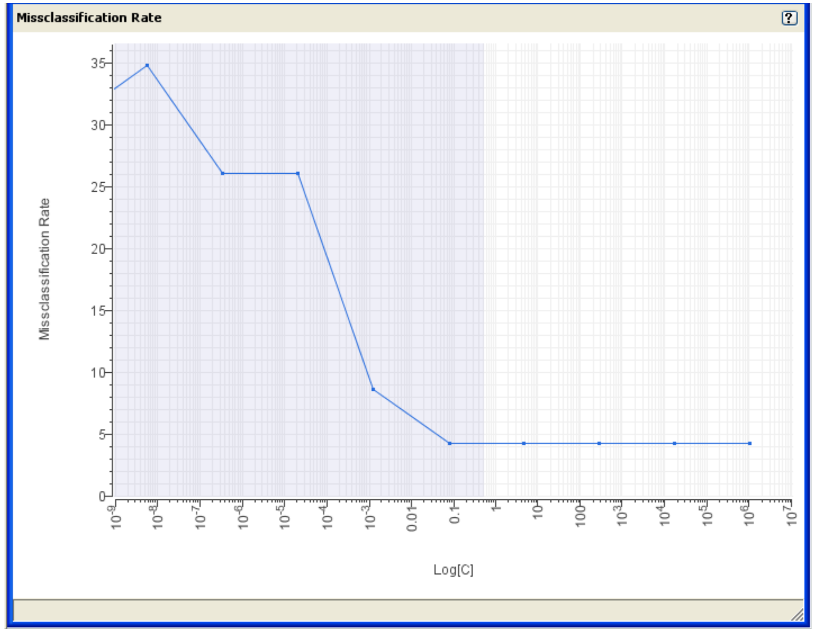
- The margin controls the bias and variance

- Small margin (large C)
  - We expect that the margin depends more on the details of the concrete realization of the data. Hence: **large variance, small bias**

- Large margin (small C)
  - The margin depends less on the details of the concrete realization. Hence **small variance, large bias**

“Experimental” Observations (SVM) for gene expression

- Geneexpression: p>>N
- C too low nearly no penalty for misclassification:
  - Overgeneralization (“don’t care”)
- C larger :
  - Converting to a stable solution.



Typical curve for gene expression Misclassification rate as a function of Log(C)

In general C is a hyper-parameter which can be optimized (beware of overfitting)

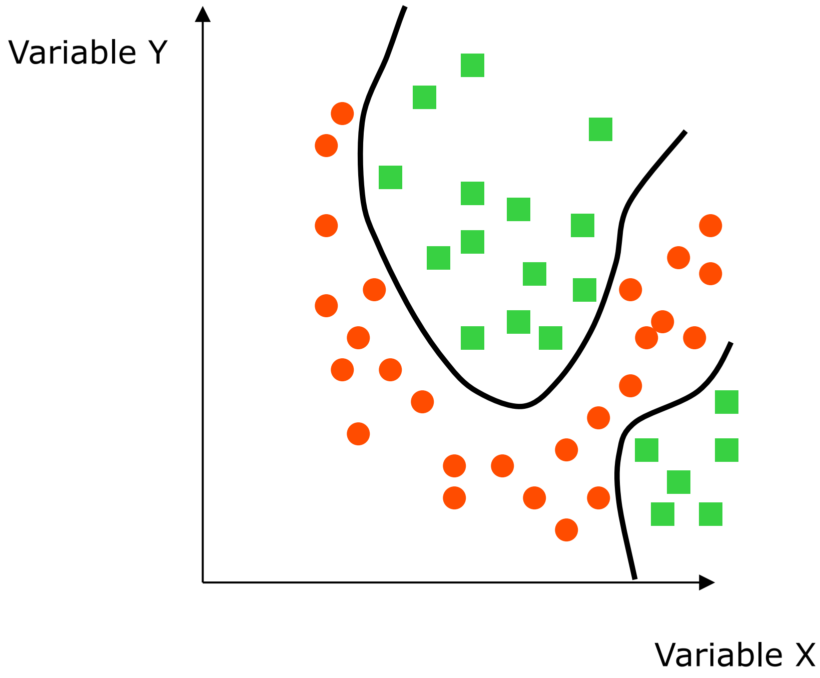
SVM in R (two classes)

```
library(e1071)
iris1 = iris[51:150,]
table(iris1$Species)
fit = svm(Species ~ ., data=iris1, kernel="linear", cost=10)
res = predict(fit, iris1)
sum(res == iris1$Species)

res_tune = tune(svm, Species ~ ., data=iris1,
kernel="linear", ranges = list(cost = c(0.1,1,10)))
summary(res_tune)
…

- Detailed performance results:
  cost error dispersion
1  0.1  0.04 0.05621827
2  1.0  0.04 0.03442652
3 10.0  0.04 0.03442652
```

Kernels

SVM - Non-separable data in the input space

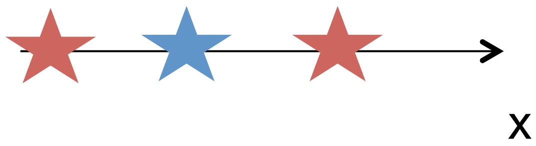
- Some problems involve non-separable data for which there does not exist a hyperplane.
- The solution is to map the data into a higher-dimensional space and define a separating hyperplane there.



Note that this is often not the typical case.

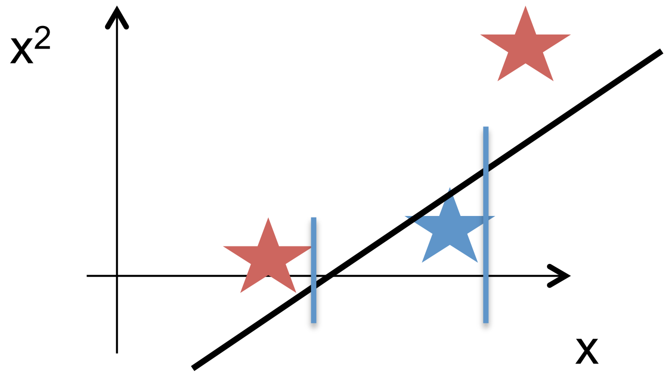
Variable Transformation, make non-separable case separable

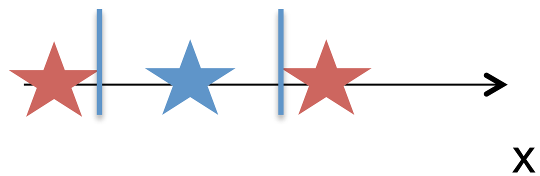
- Only a single variable x.
- Not separable by a point (hyperplane in 1D)



Take single variable x and x^2

Separable by a line (hyperplane in 2D)





View again in 1D

SVM - Feature space

- This higher-dimensional space is called the ***feature space*** as opposed to the input space.
- With an appropriately chosen feature space of sufficient dimensionality any consistent training set can be made separable.
- Example (last slide) x→(x,x^2)
- In the program, one has to calculate

Optimization:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'} \quad 0 \leq \alpha_i \leq C \quad \sum_{i=1}^{N} \alpha_i y_i = 0$$

The only place where x enters

Kernel Trick

Optimization:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'} \qquad 0 \leq \alpha_i \leq C \qquad \sum_{i=1}^{N} \alpha_i y_i = 0$$

The only place where x enters

$$x_i^T x_{i'} = (x_{i1}, x_{i2}, \ldots, x_{ip}) \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{pmatrix} = \sum_{j=1}^{p} x_{ij} x_{i'j} =: K(x_i, x_{i'})$$

- **Kernel Trick:**

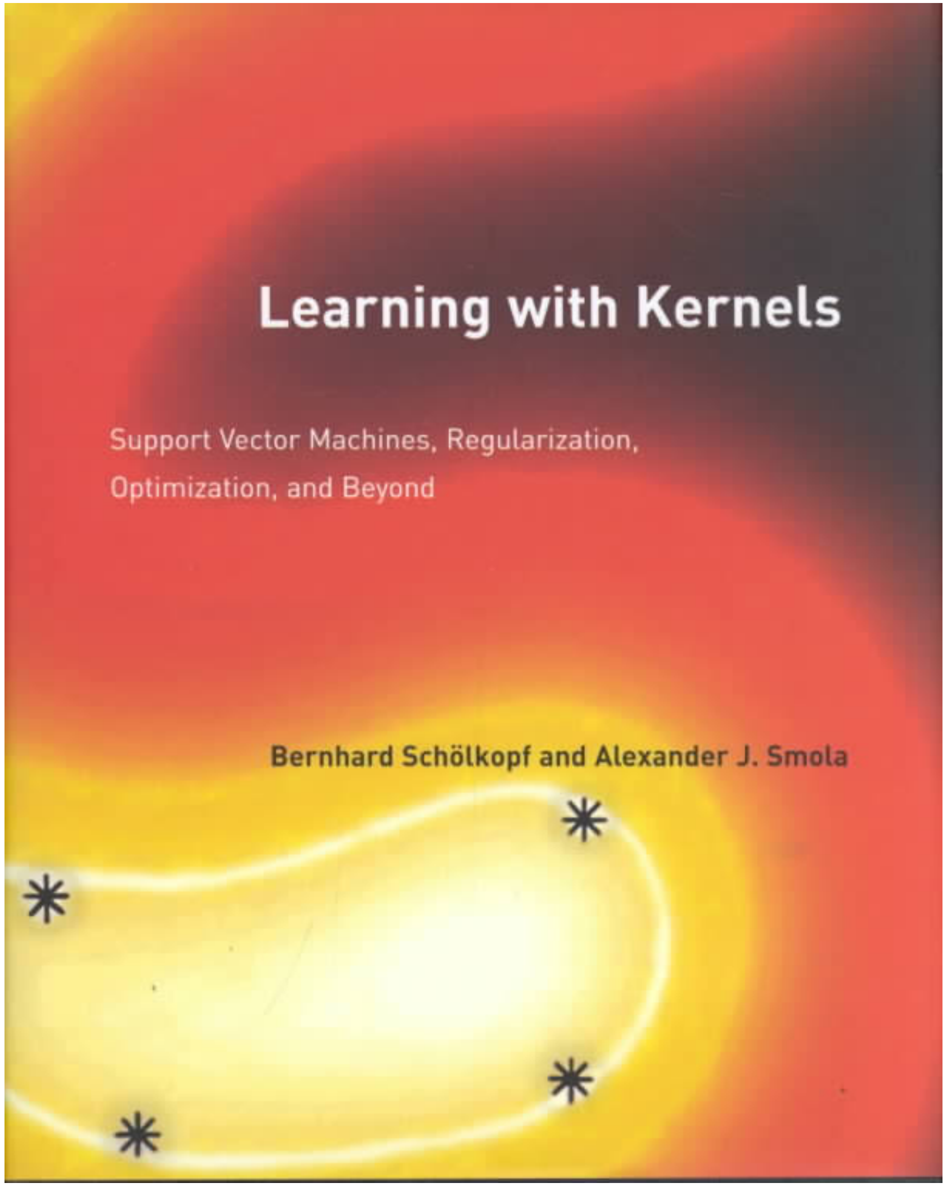
  Replace: $K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$

  With: $K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j} + \sum_{j=1}^{p} x^2_{ij} x^2_{i'j}$

  Is the same as explicitly making new features.

  “Computed on the fly”

Hot topic in 1990's and early 2000s and still used



Kernel functions

- Instead of calculating the inner product, we calculate the kernel. The following Kernels are commonly used:

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

Identity (just the inner product)
In R 'linear kernel'

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

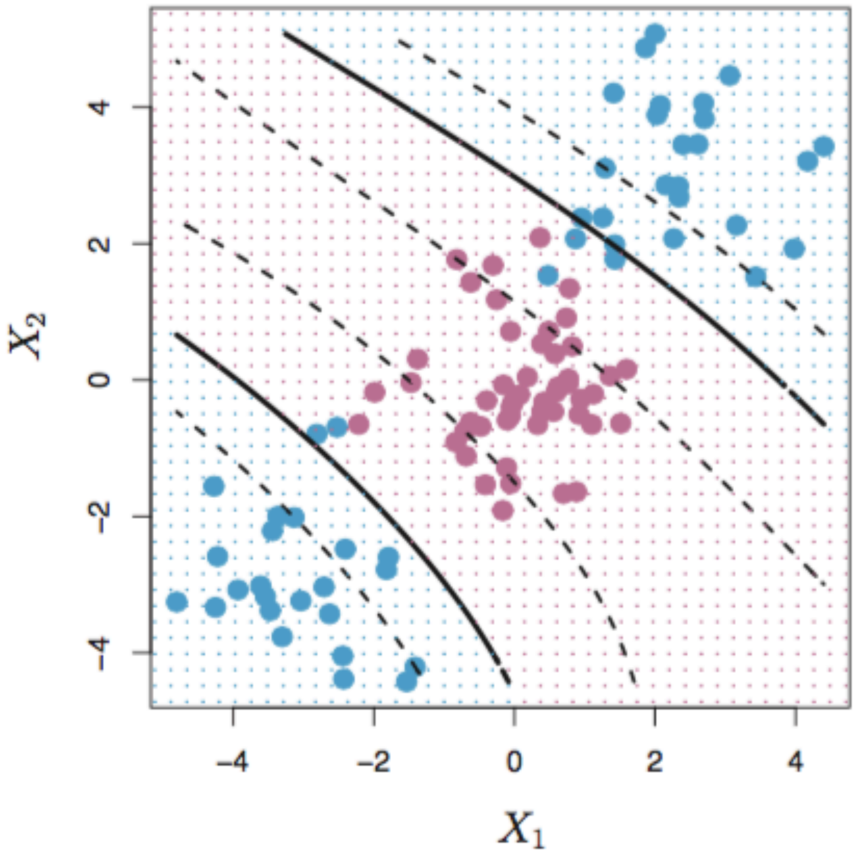
Polynomial of degree d

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$

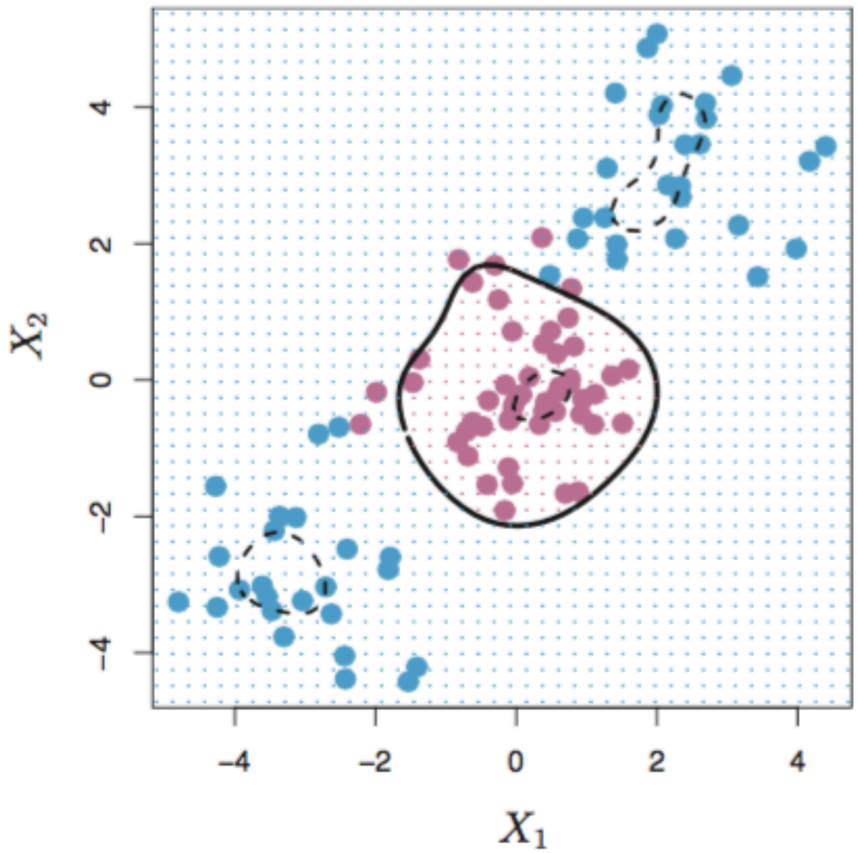
Gaussian, aka radial basis RBF.
Sometime $\gamma = 1/\sigma^2$

Kernels can also be used when data is not in the vector format. E.g. string kernels on text.

Example non-separable



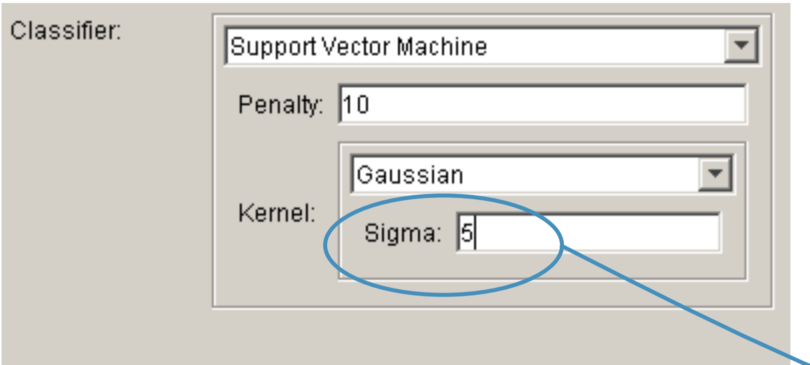
Polynomial



RBF

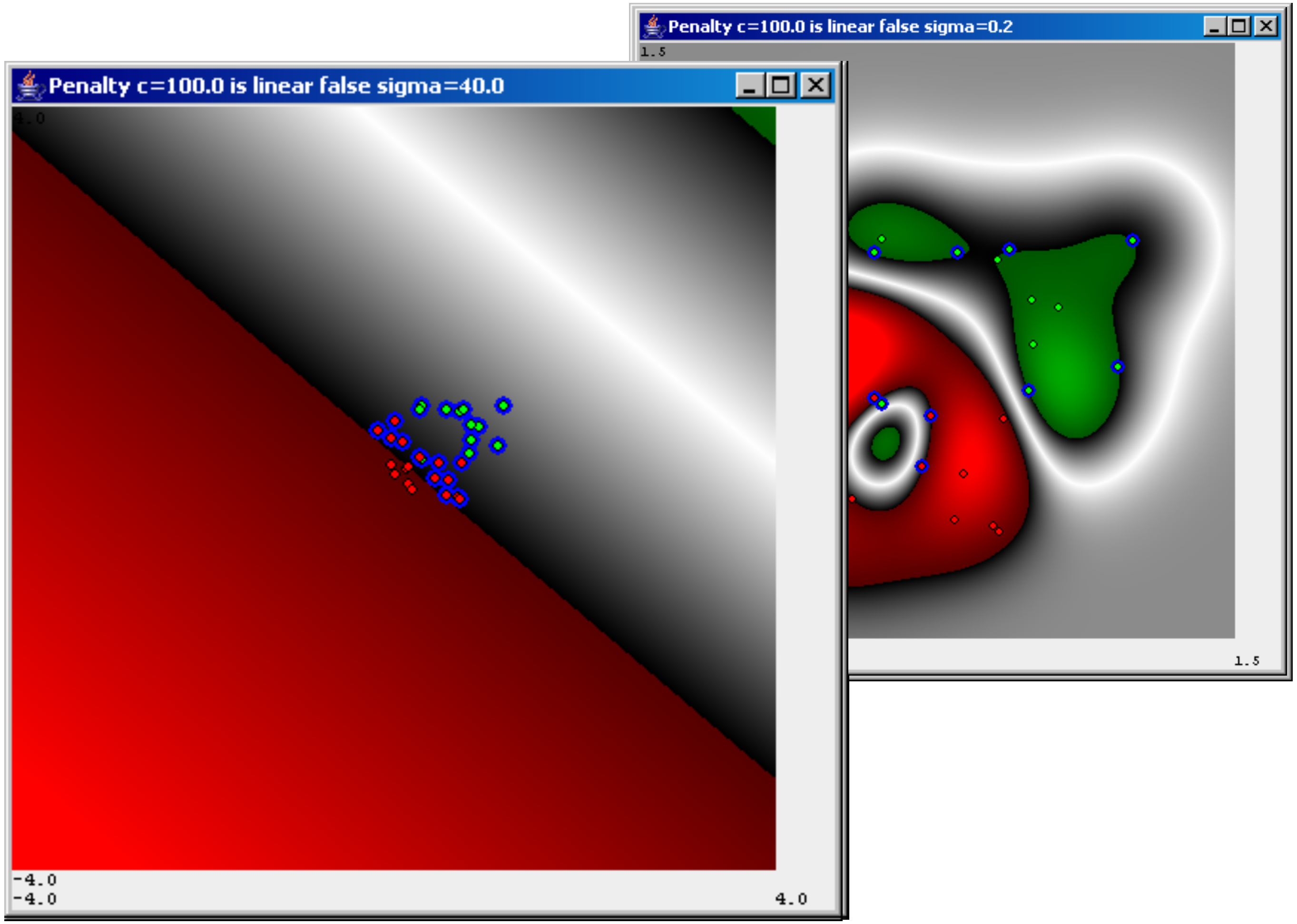
SVM - Gaussian

- In a space in which the members of a class form one or more clusters, an accurate classifier might place a Gaussian around each cluster, thereby separating the clusters from the remaining space of non-class members.
- This effect can be accomplished by placing a Gaussian with a width (sigma) over each support vector in the training set.

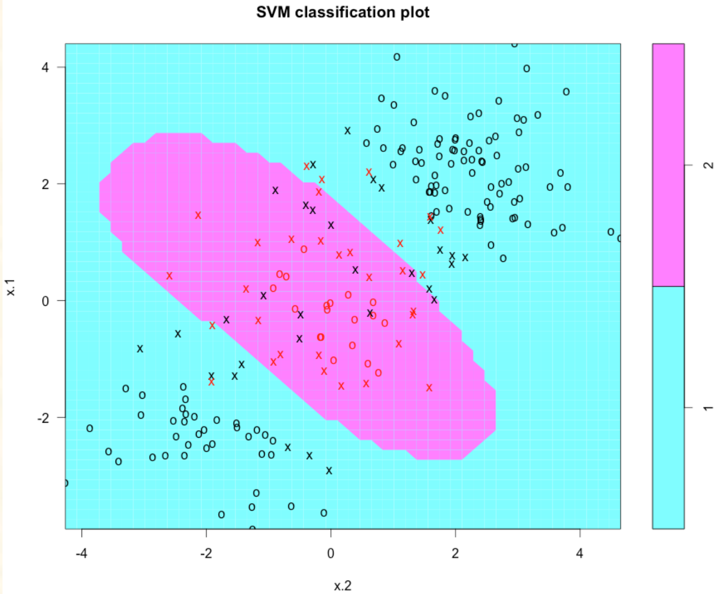


$$K(\mathbf{X},\mathbf{Y}) = exp\left(\frac{-||\bar{\mathbf{X}}-\bar{\mathbf{Y}}||^2}{2\sigma^2}\right)$$

Visualization of the parameter influence
Gaussian Kernel effect of sigma



Gaussian Kernel in R




```
############################
# Non-Linear Decission Boundary
set.seed(1)
x=matrix(rnorm(200*2), ncol=2)
x[1:100,]=x[1:100,]+2
x[101:150,]=x[101:150,]-2
y=c(rep(1,150),rep(2,50))
dat=data.frame(x=x,y=as.factor(y))

require(manipulate)
manipulate({
  svmfit=svm(y ~ .,data=dat, kernel="radial",
gamma=gamma, cost = cost)
  plot(svmfit , dat) #Plotting
}, gamma = slider(0.1,10), cost=slider(0.1,10))
```

Separation and dimensionality

Consider examples of 2 classes

Draw 2 points on a line. Can you always separate them?

Draw 3 points in a plane (not in a line!). Can you always separate them?

Imaging 4 points 3D, can you always separate them?

...

A word of warning

- It’s quite fancy to write “I have used Gaussian Kernels”. But always consider if you really need them!

- If number of features > number of examples called (p>n) you probably don’t need them.

- Overfitting is then the problem!

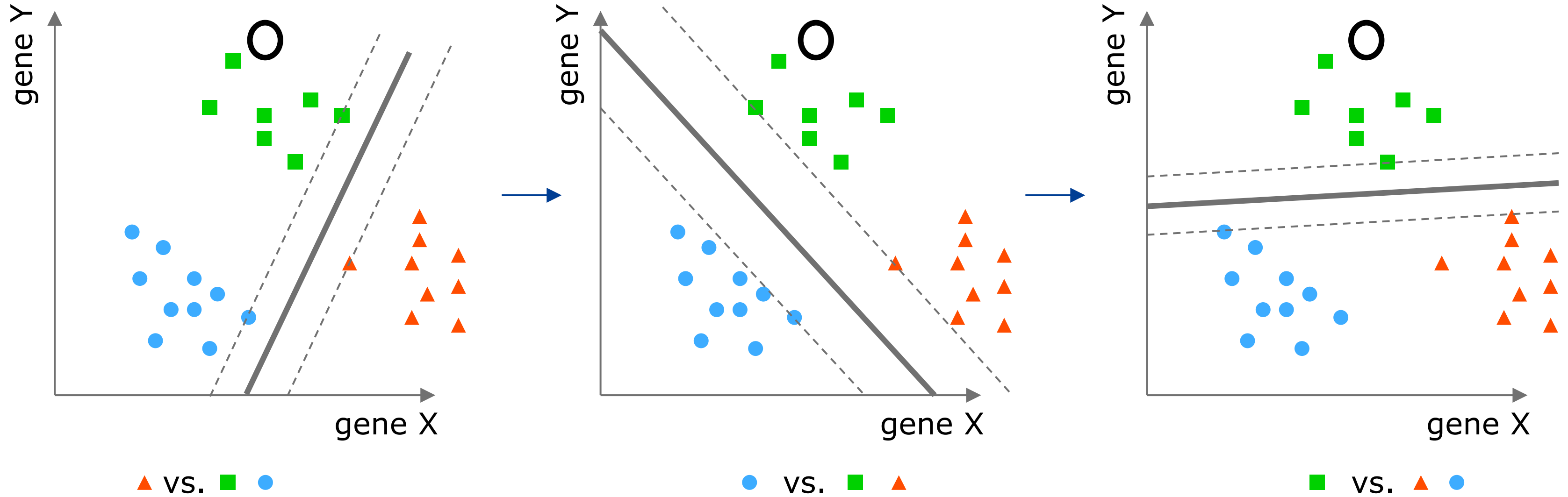
- If not it is still a good idea to try a linear kernel first!

More than 2 classes

SVM - More than 2 classes (one vs rest)

- SVM is a *binary* classifier. It can only separate two classes
- What if there are more than 2 classes?
- N>2 classes N times **'one vs. rest'**

Unkown O



Distance of O
to single class ~-3

Distance of O
to single class ~-2

Distance of O
to single class ~+2

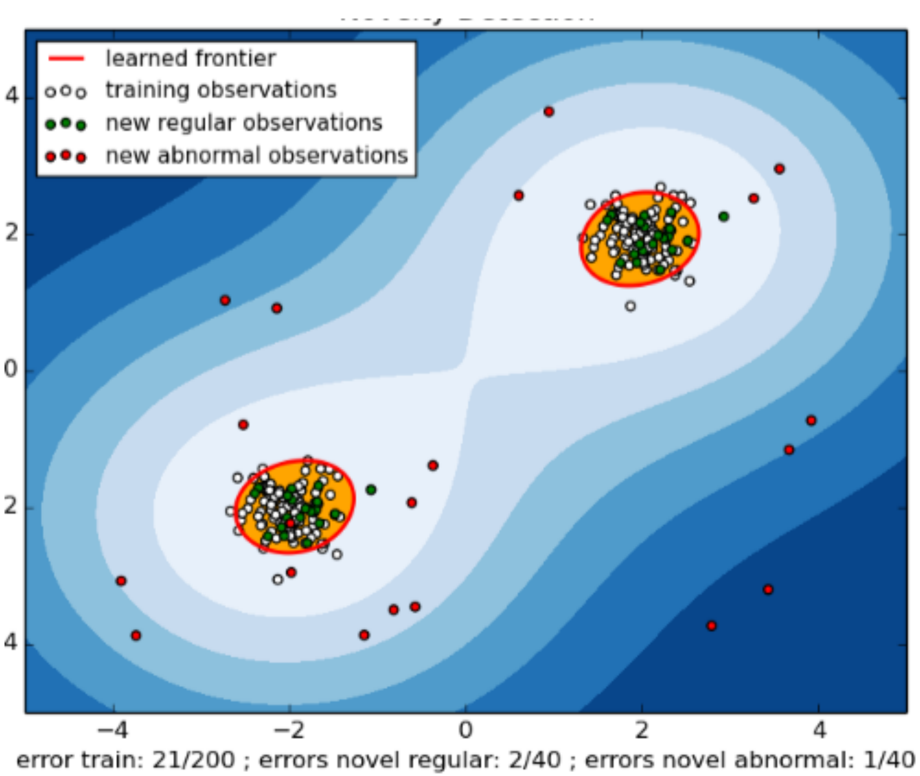
O has the highest distance in the green case.
It will be classified as green.

One vs. all classification

```
############################
# More than 2 classes
# (Preforms one vs all classification)
fit = svm(Species ~ ., data=iris,
kernel="linear", cost=10)
res = predict(fit, iris)
sum(res == iris$Species)
```

SVM Advanced Topics

- Custom Kernels e.g. for text
- SVM Regression
- Outlier Detection with one-class SVM (see below)



Praktikum

Bewertete Hausaufgabe

- Mitmachen an einer Data Science Challenge
- Erste Möglichkeit Otto Produkt Klassifikation (https://www.kaggle.com/c/otto-group-product-classification-challenge)

otto group

Completed • $10,000 • 3,514 teams

Otto Group Product Classification Challenge

Tue 17 Mar 2015 – Mon 18 May 2015 (6 months ago)

- Einreichen unter:
  - http://srv-lab-t-864/submission/Otto_2016/
- Leaderboard:
  - http://srv-lab-t-864/leaderboard/Otto_2016/
- Andere Challenges von Kaggle
  - Nach Rücksprache können Sie auch an einer anderen Kaggle Challenge teilnehmen (nicht Titanic)
  - Zum Beispiel: MNIST
  - Beachten Sie, es muss ein Klassifizierungsproblem sein.
  - Username muss dann mitgeteilt werden

Bewertete Hausaufgabe

- 2er Teams OK
- Teams melden bis 9 Dezember
- Vorstellung im letzten Praktikum (20.12.2016)
  - Etwa 10-20 Minuten
- Einreichen der Lösung
- Bewertung in halben Noten
  - Performance
  - Vortrag
  - Folien
- Note zählt nur zur Verbesserung!

Code für LSG

```
X_Train = read.table("train_otto.csv", sep=';', header = TRUE, stringsAsFactors = FALSE)
X_Test = read.table("test_otto.csv", sep=';', header = TRUE, stringsAsFactors = FALSE)

# LDA
library(MASS)
fit = lda(target ~ ., data = X_Train)
res = predict(fit, X_Test)
df = data.frame(key=X_Test$id, value=res$class)
write.table(x=df, file = 'predictions_lda.csv', sep=';', row.names = FALSE)
```