Introduction to Machine Learning
NPFL 054

http://ufal.mff.cuni.cz/course/npfl054

Barbora Hladká
hladka@ufal.mff.cuni.cz

Martin Holub
holub@ufal.mff.cuni.cz

Charles University,
Faculty of Mathematics and Physics,
Institute of Formal and Applied Linguistics

Outline

- **Support Vector Machines (SVM)**
- **Evaluation of binary classifiers (cntnd): ROC curve**

Support Vector Machines

Basic idea of SVM for binary classification tasks

We find a plane that separates the two classes in the feature space.

If it is not possible

- allow some training errors, or
- enrich the feature space so that finding a separating plane is possible

Support Vector Machines

Three key ideas

- Maximizing the margin
- Duality optimization task
- Kernels

Key concepts needed

- Hyperplane
- Dot product
- Quadratic programming

Hyperplane

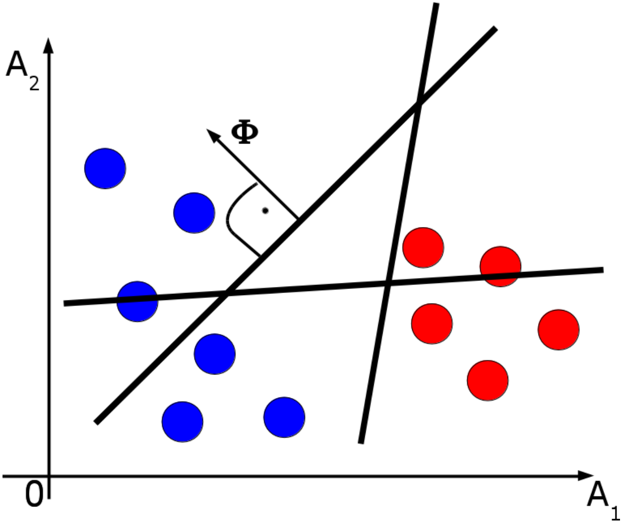
A **hyperplane** of an m-dimensional space is a subspace with dimension $m - 1$.

Mathematical definition

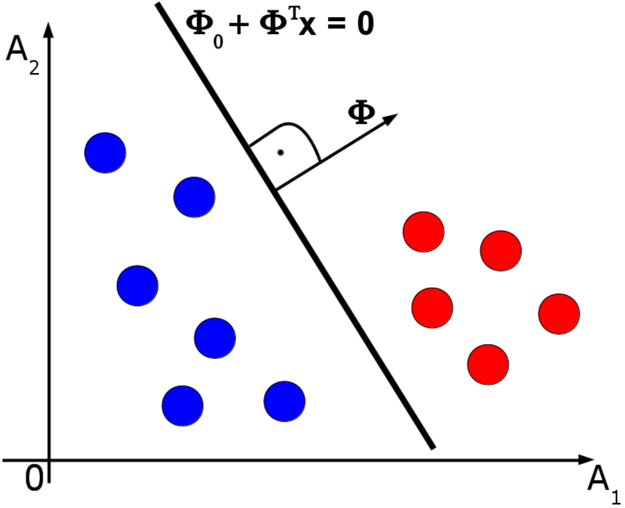
$$\Theta_0 + \mathbf{\Theta}^T \mathbf{x} = 0, \text{where } \mathbf{\Theta} = \langle \Theta_1, \ldots, \Theta_m \rangle$$

- If $m = 2$, a hyperplane is a line
- If $m = 3$, a hyperplane is a plane
- $\mathbf{\Theta}$ is a normal vector
- If $\overline{\mathbf{x}}$ satisfies the equation, then it lies on the hyperplane
- If $\Theta_0 + \mathbf{\Theta}^T \overline{\mathbf{x}} \neq 0$, then $\overline{\mathbf{x}}$ lies to one side of the hyperplane

Hyperplane



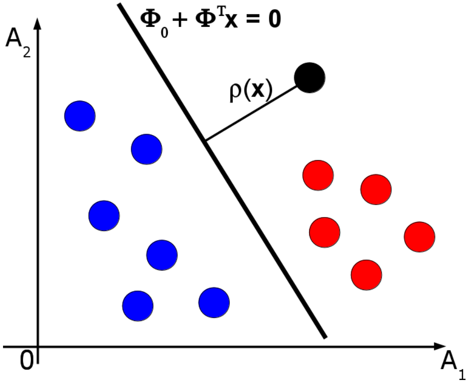
Hyperplane
Separating hyperplane



Hyperplane
Point-hyperplane distance

Distance of x to the hyperplane $\Theta_0 + \mathbf{\Theta}^T\mathbf{x} = 0$

$$\rho(\mathbf{x}) = \frac{|\Theta_0 + \mathbf{\Theta}^T\mathbf{x}|}{||\mathbf{\Theta}||}$$



Dot product

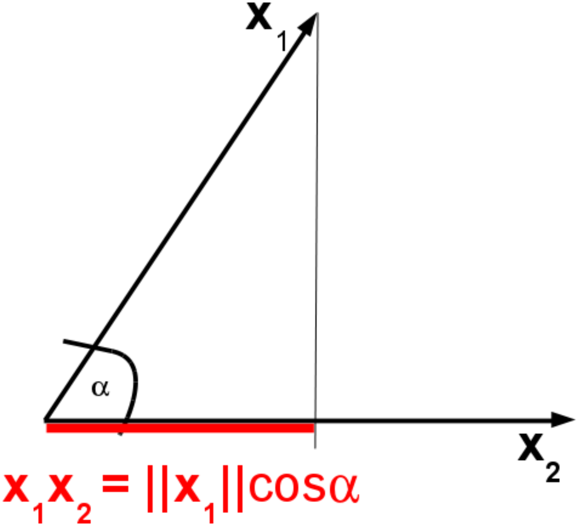
- $\mathbf{x} \in \mathcal{R}^m$
- length of $\mathbf{x}$ $||\mathbf{x}|| = \sqrt{\sum_{i=1}^{m} x_i^2}$
- dot product of two vectors $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{R}^m$

$$\mathbf{x}_1\mathbf{x}_2 = \sum_{i=1}^{m} x_{1_i} x_{2_i}$$

  - $\mathbf{x}_1\mathbf{x}_2 = ||\mathbf{x}_1||.||\mathbf{x}_2||.\cos\alpha$
  - geometric interpretation of $\mathbf{x}_1\mathbf{x}_2$:
    the length of the projection of $\mathbf{x}_1$ onto the unit vector $\mathbf{x}_2$ ($||\mathbf{x}_2|| = 1$)
  - $\mathbf{x}\mathbf{x} = ||\mathbf{x}||^2$

Dot product

$||\mathbf{x}_2|| = 1$



Quadratic programming

Quadratic programming is the problem of optimizing a quadratic function of several variables subject to linear constraints on these variables.

Support Vector Machines
Binary classification task $Y = \{+1, -1\}$

h has a form of

$$h(\mathbf{x}) = \mathrm{sgn}(\Theta_0 + \mathbf{\Theta}^T\mathbf{x})$$

Support Vector Machines
Binary classification task $Y = \{+1, -1\}$

Outline

1. Large margin classifier (linear separability)
2. Soft margin classifier (not linear separability)
3. Kernels (non-linear class boundaries)

Support Vector Machines
Binary classification task $Y = \{+1, -1\}$

Data set $Data = \{\langle \mathbf{x}_i, y_i \rangle, \mathbf{x}_i \in X, y_i \in \{-1, +1\}\}$ is **linearly separable**

if there exists a hyperplane so that all instances from *Data* are classified correctly.

Support Vector Machines
Binary classification task $Y = \{+1, -1\}$

Assume a hyperplane g: $\Theta_0 + \mathbf{\Theta}^T\mathbf{x} = 0$

- **Margin of x** w.r.t. g is distance of $\mathbf{x}$ to g:

$$\rho_g(\mathbf{x}) = \frac{|\Theta_0 + \mathbf{\Theta}^T\mathbf{x}|}{||\mathbf{\Theta}||}$$

- **Functional margin of x**, $\langle \mathbf{x}, y \rangle \in Data$ w.r.t. g is

$$\overline{\rho}_g(\mathbf{x}, y) = y(\Theta_0 + \mathbf{\Theta}^T\mathbf{x})$$

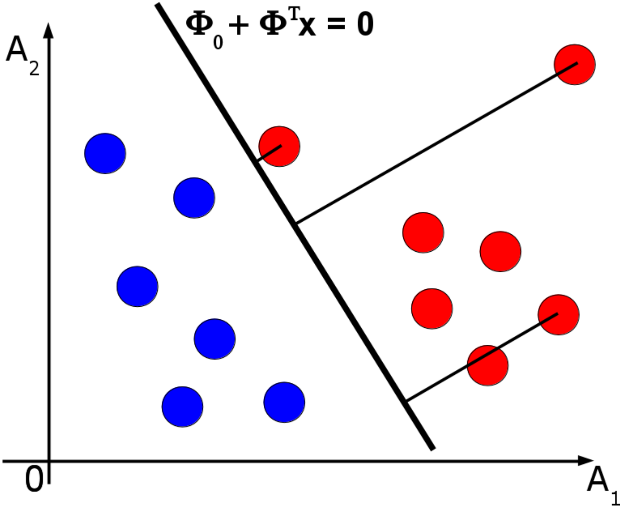
  Is $\mathbf{x}$ *classified correctly or not?*
  Large functional margin represents correct and confident classification.

- **Geometric margin of x**, $\langle \mathbf{x}, y \rangle \in Data$ w.r.t. g is

$$\rho_g(\mathbf{x}, y) = \overline{\rho}_g(\mathbf{x}, y)/||\mathbf{\Theta}||$$

  I.e. functional margin scaled by $||\mathbf{\Theta}||$

Geometric margin of x



Functional margin of *Data* w.r.t. g

$$\overline{\rho}_g(Data) = \min_{\langle \mathbf{x}, y\rangle \in Data} \overline{\rho}_g(\mathbf{x}, y)$$

Geometric margin of *Data* w.r.t. g

$$\rho_g(Data) = \min_{\langle \mathbf{x}, y \rangle \in Data} \rho_g(\mathbf{x}, y)$$

Large Margin Classifier
Training data is linearly separable

We look for $g^\star$ so that

$$g^\star = \mathrm{argmax}_g \rho_g(Data)$$

Large Margin Classifier
Training data is linearly separable

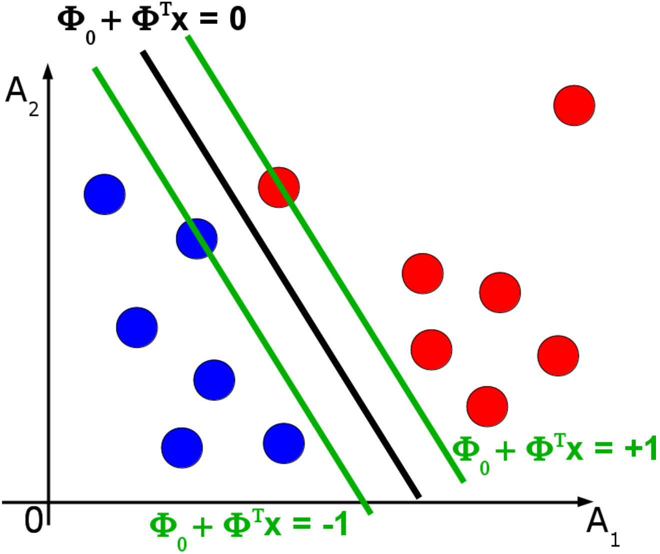
$\Theta_0 + \mathbf{\Theta}^T\mathbf{x}$ and $k\Theta_0 + (k\mathbf{\Theta})^T\mathbf{x}$ define the same hyperplane.

$$\frac{y_i(\Theta_0 + \mathbf{\Theta}^T\mathbf{x}_i)}{||\mathbf{\Theta}||} = \frac{y_i(k\Theta_0 + (k\mathbf{\Theta})^T\mathbf{x}_i)}{||k\mathbf{\Theta}||}$$

Thus, we can choose $\mathbf{\Theta}$ so that $\overline{\rho}_g(Data) = 1$. Then

$$g^\star = \mathrm{argmax}_g \rho_g(Data) = \mathrm{argmax}_g \frac{1}{||\mathbf{\Theta}||}$$

Large Margin Classifier
Training data is linearly separable



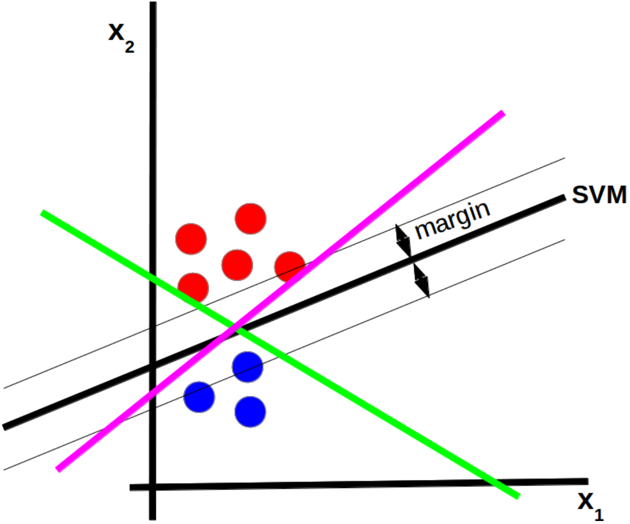
Large Margin Classifier
Training data is linearly separable

Goal: Orientate the separatig hyperplane to be as far as possible from the closest instances of both classes.

$$\boldsymbol{\Theta}^{\star} = \mathrm{argmax}_{\boldsymbol{\Theta}} \frac{1}{||\boldsymbol{\Theta}||}$$

Support vectors are the instances touching the margins.

Large Margin Classifier
Training data is linearly separable



Large Margin Classifier
Training data is linearly separable

$$\mathbf{\Theta}^\star = \text{argmax}_{\mathbf{\Theta}} \frac{1}{||\mathbf{\Theta}||} \equiv \text{argmin}_{\mathbf{\Theta}} \frac{1}{2} ||\mathbf{\Theta}||^2$$

Large Margin Classifier
Training data is linearly separable

Primal problem
Optimization problem in $m + 1$ parameters with n linear inequality constrainst

Minimize

$$\frac{1}{2}||\mathbf{\Theta}||^2$$

subject to

$$y_i(\Theta_0 + \mathbf{\Theta}^T\mathbf{x}_i) \geq 1, i = 1, \ldots\ n$$

Properties

1. Convex optimization
2. Unique solution for linearly separable training data

Large Margin Classifier
Training data is linearly separable

For each training example $\langle \mathbf{x}_i, y_i \rangle$
introduce Lagrange multiplier $\alpha_i \geq 0$. Let $\boldsymbol{\alpha} = \langle \alpha_1, ..., \alpha_n \rangle$.

Primal Lagrangian $L(\boldsymbol{\Theta}, \Theta_0, \boldsymbol{\alpha})$ is given by

$$L(\boldsymbol{\Theta}, \Theta_0, \boldsymbol{\alpha}) = \frac{1}{2}||\boldsymbol{\Theta}||^2 - \sum_i \alpha_i (y_i (\Theta_0 + \boldsymbol{\Theta}^T \mathbf{x}_i) - 1) \tag{1}$$

subject to

$$\alpha_i [y_i (\Theta_0 + \boldsymbol{\Theta}^T \mathbf{x}_i) - 1] = 0, i = 1, \ldots n$$

Large Margin Classifier
Training data is linearly separable

1. Minimize L w.r.t. $\mathbf{\Theta}$
   Thus differentiate L w.r.t. $\mathbf{\Theta}$ and $\frac{\partial L}{\partial \mathbf{\Theta}} = 0$
   It gives

$$\mathbf{\Theta} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \tag{2}$$

2. Minimize L w.r.t. Θ_0
   Thus differentiate L w.r.t. Θ_0 and $\frac{\partial L}{\partial \Theta_0} = 0$
   It gives

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{3}$$

Large Margin Classifier
Training data is linearly separable

3. Substitute (2) into the primal form (1).
   Then

$$L(\boldsymbol{\Theta}, \Theta_0, \boldsymbol{\alpha}) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

   subject to

$$\alpha_i \geq 0, \sum_i \alpha_i y_i = 0, i = 1 \ldots n$$

4. Solve the dual problem, i.e. maximize a quadratic function.
5. Get $\boldsymbol{\alpha}^\star$
6. Then $\boldsymbol{\Theta}^\star = \sum_{i=1}^n \alpha_i^\star y_i \mathbf{x}_i$, $\Theta_0 = -\frac{1}{2}(\min_{y_i=+1}({\boldsymbol{\Theta}^\star}^T \mathbf{x}_i) + \max_{y_i=-1}({\boldsymbol{\Theta}^\star}^T \mathbf{x}_i))$

Large Margin Classifier
Training data is linearly separable

- $\mathbf{\Theta}^\star$ is the solution to the primal problem
- $\boldsymbol{\alpha}^\star$ is the solution to the dual problem
- due to certain properties of $\mathbf{\Theta}^\star$ and $\boldsymbol{\alpha}^\star$, the solutions must satisfy the Karush-Kuhn-Tucker conditions where one of them is so called *KKT dual complementarity*:

$$\alpha_i * (1 - y_i(\Theta_0 + \mathbf{\Theta}^T \mathbf{x}_i) = 0$$

  - $y_i(\Theta_0 + \mathbf{\Theta}^T \mathbf{x}_i) \neq 1$ ($\mathbf{x}_i$ is not support vector) $\Rightarrow \alpha_i = 0$
  - $\alpha_i \neq 0 \Rightarrow y_i(\Theta_0 + \mathbf{\Theta}^T \mathbf{x}_i) = 1$ ($\mathbf{x}_i$ is support vector)

I.e., finding $\mathbf{\Theta}$ is equivalent to finding support vectors and their weights

Large Margin Classifier
Training data is linearly separable

Prediction for a new instance $\mathbf{x}$

$$h(\mathbf{x}) = \mathrm{sgn}(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \mathbf{x} + \Theta_0)$$

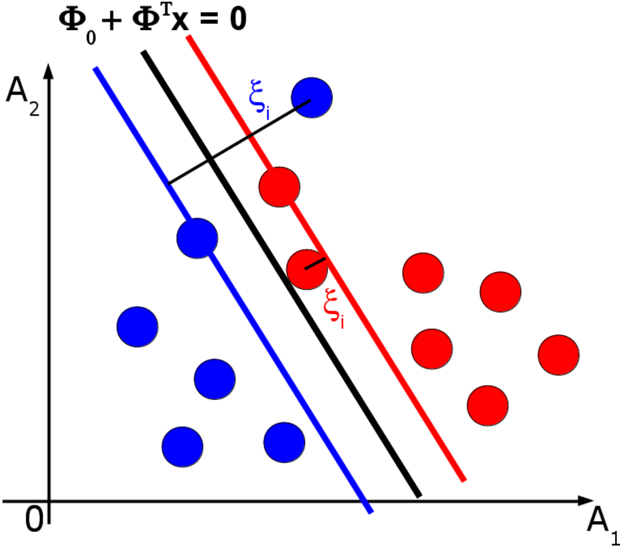
- similarity between $\mathbf{x}$ and support vector $\mathbf{x}_i$: a support vector that is more similar contributes more to the classification
- support vector that is more important, i.e. has larger α_i, contributes more to the classification
- if y_i is positive, than the contribution is positive, otherwise negative

Soft Margin Classifier
Training data is not linearly separable

In a real problem it is unlikely that a line will exactly separate the data – even if a curved decision boundary is possible. So exactly separating the data is probably not desirable – if the data has noise and outliers, a smooth decision boundary that ignores a few data points is better than one that loops around the outliers.
Thus

$$\text{minimize } ||\boldsymbol{\Theta}||^2 \textbf{ AND } \text{the number of training mistakes}$$

Soft Margin Classifier
Training data is not linearly separable



Soft Margin Classifier
Training data is not linearly separable

Introducing slack variables $\xi_i \geq 0$

- $\xi_i = 0$ if $\mathbf{x}_i$ is correctly classified
- ξ_i is distance to "its supporting hyperplane" otherwise
  - $0 < \xi_i \leq 1/||\boldsymbol{\Theta}||$: margin violation
  - $\xi_i > 1/||\boldsymbol{\Theta}||$: misclassification

Soft Margin Classifier
Training data is not linearly separable

Primal problem

Minimize

$$\frac{1}{2}||\mathbf{\Theta}||^2 + C\sum_{i=1}^{n}\xi_i$$

subject to constraint

$$y_i(\Theta_0 + \mathbf{\Theta}^T\mathbf{x}_i) \geq 1 - \xi_i, i = 1, \ldots n$$

- $C \geq 0$ trade-off parameter
  - small $C \Rightarrow$ large margin
    relaxed model; misclassifications are not penalized
  - large $C \Rightarrow$ narrow margin
    misclassifications are penalized strongly
    the model will not generalize much

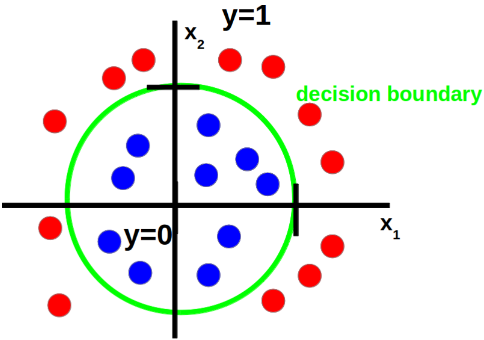
Soft Margin Classifier
Training data is not linearly separable

- Do quadratic programming as for Large Margin Classifier
- **Prediction** for a new instance $\mathbf{x}$

$$h(\mathbf{x}) = \mathrm{sgn}(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \mathbf{x} + \Theta_0)$$

Support Vector Machines
Non-linear boundary

If the examples are separated by a nonlinear region



Support Vector Machines
Non-linear boundary

Recall polynomial regression

Polynomial regression is an extension of linear regression where the relationship between features and target value is modelled as a d-th order polynomial.

Simple regression
$y = \Theta_0 + \Theta_1 x_1$

Polynomial regression
$y = \Theta_0 + \Theta_1 x_1 + \Theta_2 x_1^2 + \ldots \Theta_d x_1^d$

It is still a linear model with features $A_1, A_1^2, \ldots, A_1^d$.

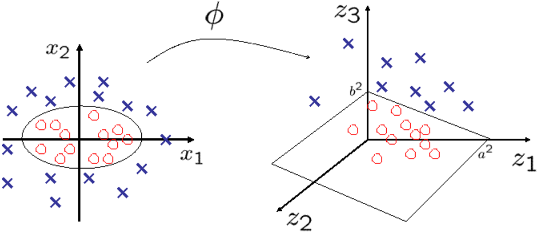
This defines a feature mapping $\phi(x_1) = [x_1, x_1^2, \ldots, x_1^d]$

Support Vector Machines
Kernels

Idea

- Apply Large/Soft margin classifier not to the orginal features but to the features obtained by the feature mapping ϕ
  – $\phi(\mathbf{x}) : \mathcal{R}^m \rightarrow \mathcal{F}$
- Large/Soft margin classifier uses dot product $\mathbf{x}_i \mathbf{x}_j$. Now, replace it with $\phi(\mathbf{x}_i)\phi(\mathbf{x}_j)$.

Support Vector Machines
Kernels



$$\phi : (x_1, x_2) \longrightarrow (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

$$\left(\frac{x_1}{a}\right)^2 + \left(\frac{x_2}{b}\right)^2 = 1 \longrightarrow \frac{z_1}{a^2} + \frac{z_3}{b^2} = 1$$

Source: `http://omega.albany.edu:8008/machine-learning-dir/notes-dir/ker1/ker1-l.html`

However, finding ϕ could be expensive.

Kernel trick

- No need to know what ϕ is and what the feature space is, i.e. no need to explicitly map the data to the feature space
- Define a kernel function $K : \mathcal{R}^m \times \mathcal{R}^m \rightarrow \mathcal{R}$
- Replace the dot product $\mathbf{x}_i \mathbf{x}_j$ with a Kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$:

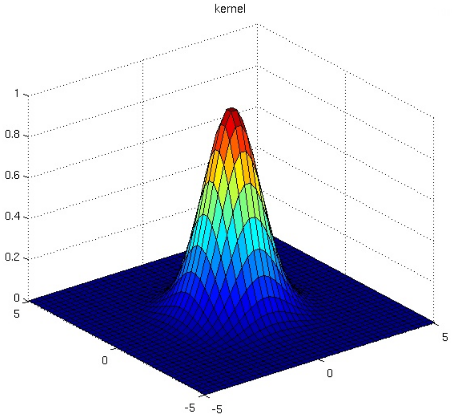
$$L(\boldsymbol{\alpha}) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

Support Vector Machines
Common kernel functions

- **Linear**
  $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
- **Polynomial**
  $K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i^T \mathbf{x}_j + c)^d$
  – smaller degree can generalize better
  – higher degree can fit (only) training data better
- **Radial basis function**
  $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma(||\mathbf{x}_i - \mathbf{x}_j||^2))$
  – very robust
  – use it when polynomial kernel is weak to fit data
- **Sigmoid**
  $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma \mathbf{x}_i^T \mathbf{x}_j + c)$, where $\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$

Radial Basis Function Kernel

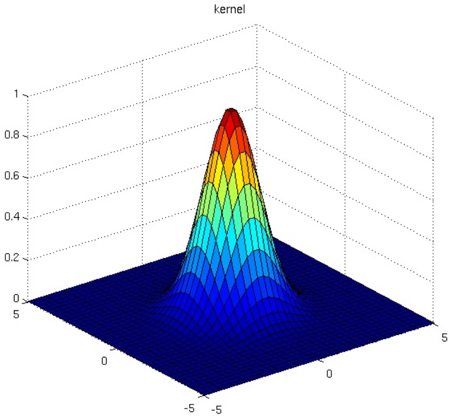
$K(\mathbf{x}, \mathbf{l}_j) = e^{-\gamma ||\mathbf{x} - \mathbf{l}_j||^2}$

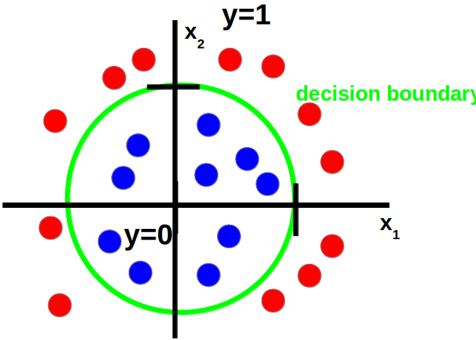


Source: http://www.cs.toronto.edu/ duvenaud/cookbook/index.html

Radial Basis Function Kernel

$$K(\mathbf{x}, \mathbf{l}_j) = e^{-\gamma||\mathbf{x}-\mathbf{l}_j||^2}$$





Support Vector Machines
Multiclass classification tasks

One-to-one

- Train $\binom{K}{2}$ SVM binary classifiers
- Classify $\mathbf{x}$ using each of the $\binom{K}{2}$ classifiers. The instance is assigned to the class which is the most frequent class assigned in the pairwise classification.

Support Vector Machines
Multiclass classification tasks

One-to-all

- Train K SVM binary classifiers. Each of them, doing classification of k-th class (+1) to the others (-1), is characterized by the hypothesis parameters $\mathbf{\Theta}_k = \langle \Theta_{0_k}, \dots, \Theta_{m_k} \rangle$, $k = 1, \dots, K$
- The instance $\mathbf{x}$ is assigned to the class $k^\star = \max_k \mathbf{\Theta}_k^T \mathbf{x}$

Evaluation of binary classifiers
Sensitivity vs. specificity

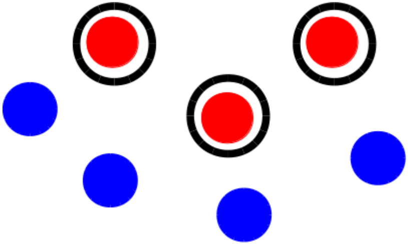
Confusion matrix

		Predicted class	
		Positive	**Negative**
True class	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Measure	**Formula**
Precision	TP/(TP+FP)
Recall/Sensitivity	TP/(TP+FN)
Specificity	TN/(TN+FP)
1-Specificity	FP/(TN+FP)
Accuracy	(TP+TN)/(TP+FP+TN+FN)

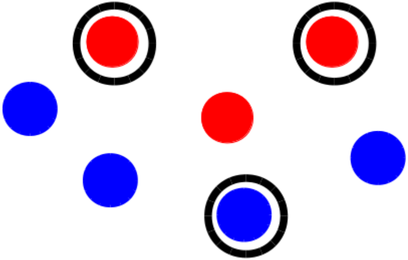
Evaluation of binary classifiers
Sensitivity vs. specificity

Perfect classifier



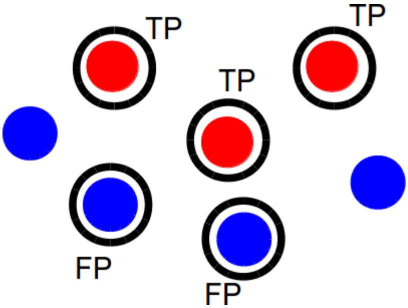
Evaluation of binary classifiers
Sensitivity vs. specificity

Reality



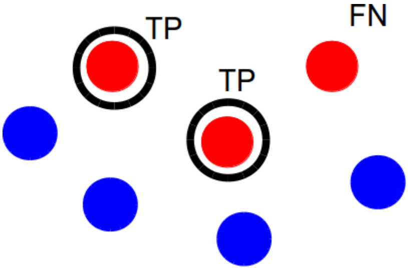
Evaluation of binary classifiers
Sensitivity vs. specificity

100% sensitive classifier



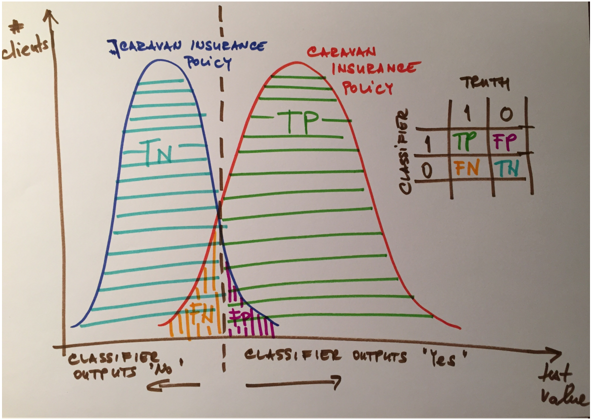
Evaluation of binary classifiers
Sensitivity vs. specificity

100% specific classifier

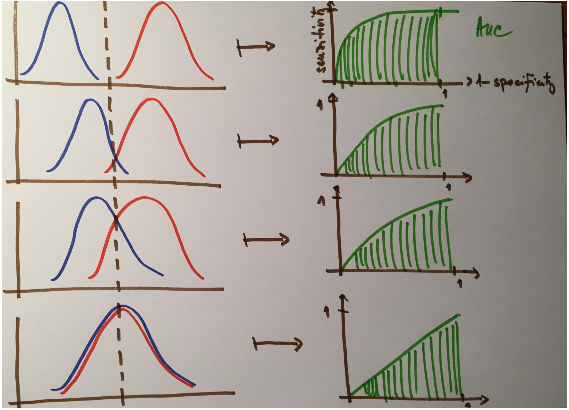


Evaluation of binary classifiers
Sensitivity vs. specificity

Sensitivity vs. specificity



ROC curve



Area Under the ROC (AUC) is a measure of how good is a distinguishing property of classifier

Summary of Examination Requirements

- Key ideas of SVM
  maximizing the margin, duality optimization task, Kernels
- Geometric/Functional margin of example/dataset
- Linearly saparable data
- Large Marging Classifier
- Soft Margin Classifiers
- Kernel trick
- Binary classifier evaluation
  sensitivity, specificity, ROC curve, AUC