Outline

- Support Vector Machines (SVM)
- Evaluation of binary classifiers: ROC curve
Basic idea of SVM for binary classification tasks

We find a plane that separates the 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
Key concepts needed

- Hyperplane
- Dot product
- Quadratic programming
A **hyperplane** of an $m$-dimensional space is a subspace with dimension $m - 1$.

**Mathematical definition**

$$\Theta_0 + \Theta^T x = 0,$$

where $\Theta = \langle \Theta_1, \ldots, \Theta_m \rangle$

- If $m = 2$, a hyperplane is a line
- $\Theta$ is a normal vector
- If $x$ satisfies the equation, then it lies on the hyperplane
- If $\Theta_0 + \Theta^T x \geq 0$, then $x$ lies to one side of the hyperplane
Hyperplane
Separating hyperplane

\[ \Phi_0 + \Phi^T x = 0 \]
Dot product

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

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

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

\[ \mathbf{x}_1 \cdot \mathbf{x}_2 = \|\mathbf{x}_1\| \|\mathbf{x}_2\| \cos \alpha \]
Point-hyperplane distance

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

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

Quadratic programming is the problem of optimizing a quadratic function of several variables subject to linear constraints on these variables.
SVM for binary classification tasks

\( Y = \{ +1, -1 \} \)

\( h \) has a form of

\[
h(x) = \text{sgn}(\Theta_0 + \Theta^T x)
\]
Outline

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

Data set $Data = \{\langle x_i, y_i \rangle, x_i \in X, y_i \in \{-1, +1\}\}$ is **linearly separable** if there exists a hyperplane so that all instances from $Data$ are classified correctly.
Assume a hyperplane $g$: $\Theta_0 + \Theta^T x = 0$

- **Margin of $x$ w.r.t. $g$** is distance to $g$:

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

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

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

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

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

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

  *i.e. functional margin scaled by $||\Theta||$*
Geometric margin of $x$

$$\Phi_0 + \Phi^T x = 0$$
• **Geometric margin** of \( Data \) w.r.t. \( g \) is

\[
\rho_g(Data) = \arg\min_{(x,y) \in Data} \rho_g(x, y)
\]
We look for $g^*$ so that

$$g^* = \operatorname{argmax}_g \rho_g(Data)$$
• **Functional margin** of \( Data \) w.r.t. \( g \) is

\[
\bar{\rho}_g(Data) = \arg\min_{(x,y) \in Data} \bar{\rho}_g(x, y)
\]

Functional margin of training set is functional margin of closest points
Large Margin Classifier
Training data is linearly separable

\( \Theta_0 + \Theta^T x \) and \( k\Theta_0 + (k\Theta)^T x \) define the same hyperplane. i.e. distances to separator are unchanged

\[
y_i(\Theta_0 + \Theta^T x_i) = y_i(k\Theta_0 + (k\Theta)^T x_i)
\]

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

\[
g^* = \arg\max_g \rho_g(Data) = \arg\max_g \frac{1}{\|\Theta\|}
\]
Large Margin Classifier
Training data is linearly separable

\[ \Phi_0 + \Phi^T x = 0 \]
\[ \Phi_0 + \Phi^T x = -1 \]
\[ \Phi_0 + \Phi^T x = +1 \]
Large Margin Classifier
Training data is linearly separable

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

\[ \Theta^* = \arg\max_{\Theta} \frac{1}{||\Theta||} \]

**Support vectors** are the instances touching the margins.
Large Margin Classifier
Training data is linearly separable
Large Margin Classifier

Training data is linearly separable

\[ \Theta^* = \arg\max_{\Theta} \frac{1}{||\Theta||} \equiv \arg\min_{\Theta} \frac{1}{2} ||\Theta||^2 \]
Large Margin Classifier
Training data is linearly separable

Primal problem

Minimize

\[
\frac{1}{2} \| \Theta \|^2
\]

subject to constraint

\[
y_i (\Theta_0 + \Theta^T x_i) \geq 1, \quad i = 1, \ldots, n
\]

Properties

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

NPFL054  Hladká & Holub  Lecture 7, page 25/51
Large Margin Classifier
Training data is linearly separable

For each training example $\langle x_i, y_i \rangle$, introduce Lagrange multiplier $\alpha_i \geq 0$. Let $\alpha = \langle \alpha_1, ..., \alpha_n \rangle$.

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

$$L(\Theta, \Theta_0, \alpha) = \frac{1}{2} \|\Theta\|^2 - \sum_i \alpha_i (y_i (\Theta_0 + \Theta^T x_i) - 1)$$  \hspace{1cm} (1)

We wish to find the $\Theta$ and $\Theta_0$ which minimizes and the $\alpha$ which maximizes $L$. 
Large Margin Classifier
Training data is linearly separable

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

   $$\Theta = \sum_{i=1}^{n} \alpha_i y_i x_i$$  \hspace{1cm} (2)

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

   $$\sum_{i=1}^{n} \alpha_i y_i = 0$$  \hspace{1cm} (3)
3. Substitute (2) into the primal form (1). Then

\[
L = \sum_{i} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j
\]

subject to \( \alpha_i \geq 0 \) and \( \sum_{i} \alpha_i y_i = 0, i = 1 \ldots n \)
4. Solve the dual problem, i.e. maximize a quadratic function. Do quadratic programming.

\[
L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j
\]

subject to \( \alpha_i \geq 0 \) and \( \sum_i \alpha_i y_i = 0 \), \( i = 1 \ldots n \)

5. Get \( \alpha^* \)

6. Then \( \Theta^* = \sum_{i=1}^n \alpha_i^* y_i x_i \)
Large Margin Classifier
Training data is linearly separable

- $\Theta^*$ is the solution to the primal problem
- $\alpha^*$ is the solution to the dual problem

$\Theta^*$ and $\alpha^*$ satisfy the Karush-Kuhn-Tucker conditions where one of them is so called Karush-Kuhn-Tucker dual complementarity:

$$\alpha_i^* (1 - y_i (\Theta_0 + \Theta^T x_i)) = 0$$

- $y_i (\Theta_0 + \Theta^T x_i) \neq 1$ ($x_i$ is not support vector) $\Rightarrow \alpha_i = 0$
- $\alpha_i \neq 0 \Rightarrow y_i (\Theta_0 + \Theta^T x_i) = 1$ ($x_i$ is support vector)

I.e., finding $\Theta$ is equivalent to finding support vectors and their weights
Large Margin Classifier
Training data is linearly separable

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

\[
h(\mathbf{x}) = \text{sgn}\left(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + \Theta_0\right)
\]

- 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 $\alpha_i$, contributes more to the classification
- if $y_i$ is positive, than the contribution is positive, otherwise negative
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

\[
\text{minimize } ||\Theta||^2 \text{ AND the number of training mistakes}
\]
Soft Margin Classifier

Training data is not linearly separable
Introducing slack variables $\xi_i \geq 0$

- $\xi_i = 0$ if $x_i$ is correctly classified
- $\xi_i$ is distance to "its supporting hyperplane" otherwise
  - $0 < \xi_i \leq 1/||\Theta||$: margin violation
  - $\xi_i > 1/||\Theta||$: misclassification
Soft Margin Classifier
Training data is not linearly separable

Primal problem

Minimize

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

subject to constraint

$$y_i(\Theta_0 + \Theta^T x_i) \geq 1 - \xi_i, \ i = 1, \ldots, n$$

- \( C \geq 0 \) trade-off parameter
  - small \( C \) ⇒ large margin
  - large \( C \) ⇒ narrow margin
Soft Margin Classifier

Training data is not linearly separable

• Do quadratic programming as for Large Margin Classifier
• **Prediction** for a new instance $x$

$$h(x) = \text{sgn} \left( \sum_{i=1}^{n} \alpha_i y_i x_i x + \Theta_0 \right)$$
If the examples are separated by a nonlinear region
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

Idea

Apply Large/Soft margin classifier not to the original features but to the features obtained by the feature mapping $\phi: \phi(x) : \mathbb{R}^m \rightarrow \mathcal{F}$

Large/Soft margin classifier uses dot product $x_i x_j$. Now, replace it with $\phi(x_i) \phi(x_j)$. 
However, finding of $\phi$ could be expensive.

**Kernel trick**

- No need to know what $\phi$ 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 $x_i x_j$ with a Kernel function $K(x_i, x_j) :$

\[
L(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(x_i, x_j)
\]
Support Vector Machines

\[ \phi : (x_1, x_2) \rightarrow (x_1^2, \sqrt{2}x_1x_2, x_2^2) \]

\[ \left( \frac{x_1}{a} \right)^2 + \left( \frac{x_2}{b} \right)^2 = 1 \rightarrow \frac{z_1}{a^2} + \frac{z_3}{b^2} = 1 \]

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

- **Linear:** \( K(x_i, x_j) = x_i^T x_j \)
- **Polynomial:** \( K(x_i, x_j) = (\gamma x_i^T x_j + c)^d \)
- **Radial basis function:** \( K(x_i, x_j) = \exp(-\gamma(\|x_i - x_j\|))^2 \)
- **Sigmoid:** \( K(x_i, x_j) = \tanh(\gamma x_i^T x_j + c) \), where \( \tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \)
SVM with $K$ classes, $K > 2$

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.
SVM with $K$ classes, $K > 2$

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 $\Theta_k, k = 1, \ldots, K$
- The instance $x$ is assigned to the class $k^* = \max_k \Theta_k^T x$
Evaluation of binary classifiers

ROC curve

Confusion matrix

<table>
<thead>
<tr>
<th>True class</th>
<th>Predicted class</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>Positive</td>
<td>TP/(TP+FP)</td>
</tr>
<tr>
<td></td>
<td>Negative</td>
<td>False Negative (FN)</td>
</tr>
<tr>
<td>Negative</td>
<td>Positive</td>
<td>False Positive (FP)</td>
</tr>
<tr>
<td></td>
<td>Negative</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Measure</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>TP/(TP+FP)</td>
</tr>
<tr>
<td>Recall/Sensitivity</td>
<td>TP/(TP+FN)</td>
</tr>
<tr>
<td>Specificity</td>
<td>TN/(TN+FP)</td>
</tr>
<tr>
<td>Accuracy</td>
<td>(TP+TN)/(TP+FP+TN+FN)</td>
</tr>
</tbody>
</table>
Evaluation of binary classifiers

ROC curve

Perfect classifier
Evaluation of binary classifiers

ROC curve

Reality
Evaluation of binary classifiers

ROC curve

100% sensitive classifier

Diagram showing TP (True Positive) and FP (False Positive) classifications.
100% specific classifier
Evaluation of binary classifiers

ROC curve

Sensitivity vs. specificity
Evaluation of binary classifiers

ROC curve