# A Tutorial on Support Vector Machine

Tan Yee Fan

School of Computing
National University of Singapore

# Contents

# Contents

# Classifier

What is a classifier?

- A function that maps instances to classes.
- Instance usually expressed as a vector of $n$ attributes.

Example: Fit-And-Trim club

- Attributes: Gender, Weight, Height
- Class (Member): Yes, No
- Is Garvin a member of the Fit-And-Trim club?

| Person | Gender | Weight | Height | Member |
|---|---|---|---|---|
| Garvin | Male | 78 | 179 | ? |

# Classifier

Training a classifier:

- Given: Training data (a set of instances whose classes are known).
- Output: A function, selected from a predefined set of functions.

Example: Fit-And-Trim club

- Training instances:

| Person | Gender | Weight | Height | Member |
|---|---|---|---|---|
| Alex | Male | 79 | 189 | Yes |
| Betty | Female | 76 | 170 | Yes |
| Charlie | Male | 77 | 155 | Yes |
| Daisy | Female | 72 | 163 | No |
| Eric | Male | 73 | 195 | No |
| Fiona | Female | 70 | 182 | No |

- Possible classification rule: Weight more than 74.5?

# Classifier

Training a classifier:

- Aim to minimize both training error (errors on seen instances) and generalization error (errors on unseen instances).
- A classifier that has low training error but high generalization error is said to overfit the training data.

Example: Fit-And-Trim club

- Suppose we use person name as an attribute, and the trained classifier uses the following classification rules:
  - Alex $\rightarrow$ Yes
  - Betty $\rightarrow$ Yes
  - Charlie $\rightarrow$ Yes
  - Daisy $\rightarrow$ No
  - Eric $\rightarrow$ No
  - Fiona $\rightarrow$ No
- This classifier severely overfits: it achieves 100% accuracy on the training data, but is unable to classify any unseen test instance.

# Linear Classifier

- Attributes are real numbers, so each instance $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T \in \mathbb{R}^n$ is a $n$-dimensional vector.
- Classes are $+1$ (positive class) and $-1$ (negative class).
- Classification rule:

$$y = \text{sign}[f(\mathbf{x})] = \begin{cases} +1 & \text{if } f(\mathbf{x}) \geq 0 \\ -1 & \text{if } f(\mathbf{x}) < 0 \end{cases}$$

where the decision function $f(\cdot)$ is

$$f(\mathbf{x}) = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b = \mathbf{w}^T \mathbf{x} + b$$

for some weight vector $\mathbf{w} = (w_1, w_2, \ldots, w_n)^T \in \mathbb{R}^n$ and bias $b \in \mathbb{R}$.

- Training a linear classifier means tuning $\mathbf{w}$ and $b$.

# Linear Classifier

Example:

- A course is graded based on two written tests.
- Weightage: Test 1 – $30\%$, Test 2 – $70\%$.
- Students pass if the total weighted score is at least $50\%$.

Formulation:

- $x_1 =$ Test 1 score, $x_2 =$ Test 2 score.
- To pass, we need:

$$0.3x_1 + 0.7x_2 \geq 50$$

- Decision function of linear classifier:

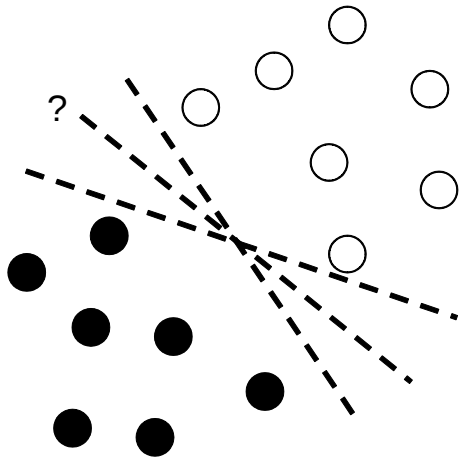
$$f(\mathbf{x}) = 0.3x_1 + 0.7x_2 - 50$$

- Positive class = pass, negative class = fail.

# Linear Classifier

$$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b = w_1x_1 + w_2x_2 + \ldots + w_nx_n + b$$

- $f(\mathbf{x}) = 0$ is a hyperplane in the $n$-dimensional real space $\mathbb{R}^n$.
- Training: Find a hyperplane that separates positive and negative instances.




## Exercise 1

*Show that* $\mathbf{w}$ *is orthogonal to the hyperplane.*

# Properties

Given a linear classifier with:

$$f(\mathbf{x}) = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b$$

The following are equivalent classifiers:

- $f(\mathbf{x}) = (aw_1)x_1 + (aw_2)x_2 + \ldots + (aw_n)x_n + (ab)$ for any constant $a > 0$.
- $f(\mathbf{x}) = w_1 x_1 + \ldots + w'_i(ax_i) + \ldots + w_n x_n + b$ for any constant $a > 0$ and where $w'_i = \frac{w_i}{a}$.
- $f(\mathbf{x}) = w_1 x_1 + \ldots + w_i(x_i + a) + \ldots + w_n x_n + b'$ for any constant $a$ and where $b' = b - aw_i$.

In other words, possible to scale whole problem, and scale and shift individual attributes.

# Properties

Using training data, we can normalize each attribute $x_i$:

- $0 \leq x_i \leq 1$.
- $x_i$ has mean $0$ and standard deviation $1$.

Normalization makes training easier by avoiding numerical difficulties.
For linear classifier with normalized attributes:

$$f(\mathbf{x}) = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + b$$

Larger $|w_i|$ or $w_i^2$ means $x_i$ more important or relevant.

- Can be used for attribute ranking or selection.
- If $x_i$ is missing, can be used to decide whether to acquire it.

# Transforming Non-numeric Attributes

If attribute values are ordered, use the ordering.

- Consider attribute values $\{\text{Small}, \text{Medium}, \text{Large}\}$.
- Map Small to $1$.
- Map Medium to $2$.
- Map Large to $3$.

If attribute values have no ordering information, create one numeric attribute whose values are $\{0, 1\}$ for each discrete attribute value.

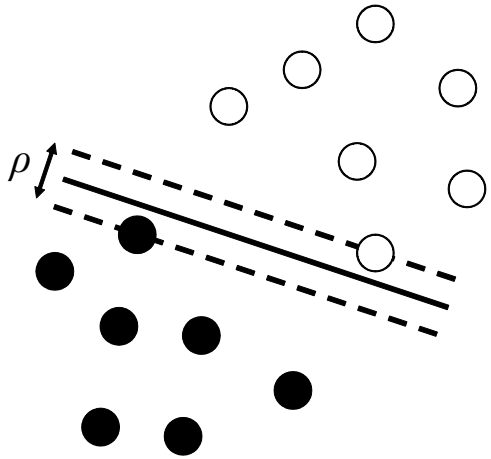
- Consider attribute values $\{\text{Red}, \text{Green}, \text{Blue}\}$.
- Create three attributes $x_{\text{Red}}$, $x_{\text{Green}}$, $x_{\text{Blue}}$.
- Map Red to $x_{\text{Red}} = 1$, $x_{\text{Green}} = 0$, $x_{\text{Blue}} = 0$.
- Map Green to $x_{\text{Red}} = 0$, $x_{\text{Green}} = 1$, $x_{\text{Blue}} = 0$.
- Map Blue to $x_{\text{Red}} = 0$, $x_{\text{Green}} = 0$, $x_{\text{Blue}} = 1$.

If attribute values are finite sets, use the above method.

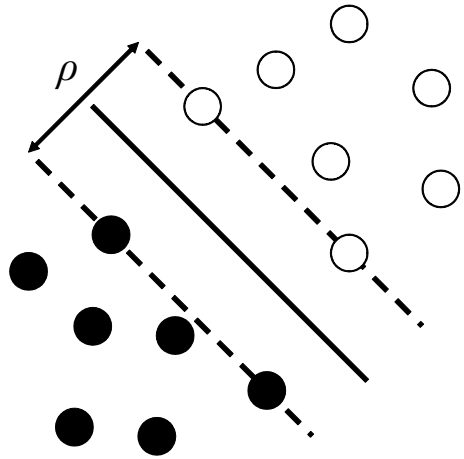
# Contents

# Support Vector Machine

Narrow margin




Wide margin

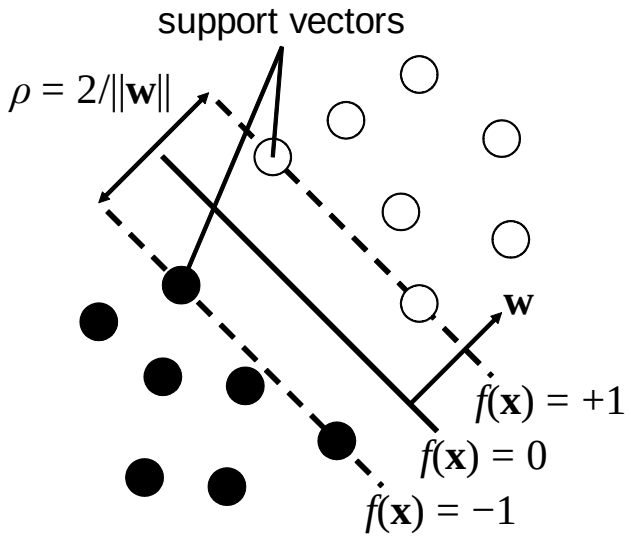



Wide margin gives better generalization performance.

# Support Vector Machine

- Support vector machine (SVM) is a maximum margin linear classifier.
- Training data: $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)\}$.
- Assume $D$ is linearly separable, i.e., can separate positive and negative instances exactly using a hyperplane.
- SVM requires:
  - Training instance $\mathbf{x}_i$ with class $y_i = +1$: $f(\mathbf{x}_i) = \mathbf{w}^T\mathbf{x}_i + b \geq +1$.
  - Training instance $\mathbf{x}_i$ with class $y_i = -1$: $f(\mathbf{x}_i) = \mathbf{w}^T\mathbf{x}_i + b \leq -1$.
- Combined, we have $y_i f(\mathbf{x}_i) = y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$.
- Support vector: A training instance $\mathbf{x}_i$ with $f(\mathbf{x}_i) = \mathbf{w}^T\mathbf{x}_i + b = \pm 1$.

# Support Vector Machine




Test instance $\mathbf{x}$: Greater $|f(\mathbf{x})| \Rightarrow$ Greater classification confidence.

Maximize $\rho \Rightarrow$ Minimize $\|\mathbf{w}\|_2 \Rightarrow$ Minimize $\mathbf{w}^T\mathbf{w}$.

($\|\mathbf{w}\|_2 = \sqrt{w_1^2 + w_2^2 + \ldots + w_n^2}$ is the length or 2-norm of the vector $\mathbf{w}$)

## Exercise 2

*Show that* $\rho = \frac{2}{\|\mathbf{w}\|_2}$.

# Support Vector Machine

Primal problem:

$$\begin{aligned} \text{Minimize} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{Subject to} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \end{aligned}$$

Primal problem is convex optimization problem, i.e., any local minimum is also global minimum.

But more convenient to solve the dual problem instead.

# Lagrangian Theory

Let $f(u_1, u_2, \ldots, u_k)$ be function of variables $u_1$, $u_2$, $\ldots$, $u_k$.

Partial derivatives:

- Partial derivative $\frac{\partial f}{\partial u_i}$ means differentiate w.r.t. $u_i$ while holding all other $u_j$'s as constants.
- Example: $f(x, y) = \sin x + xy^2$, $\frac{\partial f}{\partial x} = \cos x + y^2$, $\frac{\partial f}{\partial y} = 2xy$.
- Stationary point when $\frac{\partial f}{\partial u_i} = 0$ for all $i$.

Partial derivatives w.r.t. vectors:

- Let $\mathbf{u} = [u_1, u_2, \ldots, u_n]^T$.
- $\nabla f = \frac{\partial f}{\partial \mathbf{u}} = \left[\frac{\partial f}{\partial u_1}, \frac{\partial f}{\partial u_2}, \ldots, \frac{\partial f}{\partial u_n}\right]^T$.
- Stationary point when $\nabla f = \frac{\partial f}{\partial \mathbf{u}} = \mathbf{0}$.

## Exercise 3

*Verify that* $\frac{\partial}{\partial \mathbf{u}} \mathbf{a}^T \mathbf{u} = \frac{\partial}{\partial \mathbf{u}} \mathbf{u}^T \mathbf{a} = \mathbf{a}$ *and* $\frac{\partial}{\partial \mathbf{u}} \mathbf{u}^T \mathbf{u} = 2\mathbf{u}$.

# Lagrangian Theory

## Primal problem:

$$\begin{aligned} \text{Minimize} \quad & f(\mathbf{u}) \\ \text{Subject to} \quad & g_1(\mathbf{u}) \leq 0, g_2(\mathbf{u}) \leq 0, \ldots, g_m(\mathbf{u}) \leq 0 \\ & h_1(\mathbf{u}) = 0, h_2(\mathbf{u}) = 0, \ldots, h_n(\mathbf{u}) = 0 \end{aligned}$$

## Lagrangian function:

$$L(\mathbf{u}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = f(\mathbf{u}) + \sum_{i=1}^{m} \alpha_i g_i(\mathbf{u}) + \sum_{i=1}^{n} \beta_i h_i(\mathbf{u})$$

The $\alpha_i$'s and $\beta_i$'s are Lagrange multipliers.

Optimal solution must satisfy Karush-Kuhn-Tucker (KKT) conditions:

$$\frac{\partial L}{\partial \mathbf{u}} = \mathbf{0} \qquad g_i(\mathbf{u}) \leq 0 \qquad h_i(\mathbf{u}) = 0$$

$$\alpha_i g_i(\mathbf{u}) = 0 \qquad \alpha_i \geq 0$$

# Lagrangian Theory

KKT conditions:

$$\frac{\partial L}{\partial \mathbf{u}} = \mathbf{0} \qquad g_i(\mathbf{u}) \leq 0 \qquad h_i(\mathbf{u}) = 0$$

$$\alpha_i g_i(\mathbf{u}) = 0 \qquad \alpha_i \geq 0$$

Dual problem:

$$\begin{array}{ll} \text{Maximize} & f(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \inf_{\mathbf{u}} L(\mathbf{u}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\ \text{Subject to} & \text{KKT conditions} \end{array}$$

($\inf$ means infimum or greatest lower bound)

Relation between primal and dual objective functions:

$$f(\mathbf{u}) \geq f(\boldsymbol{\alpha}, \boldsymbol{\beta})$$

Under certain conditions (satisfied by SVM), equality occurs at optimal solution.

# Formulation

Primal problem:

$$\begin{aligned} \text{Minimize} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{Subject to} \quad & 1 - y_i(\mathbf{w}^T\mathbf{x}_i + b) \leq 0 \end{aligned}$$

Lagrangian:

$$\begin{aligned} L(\mathbf{w}, b, \boldsymbol{\alpha}) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{i=1}^{N} \alpha_i (1 - y_i(\mathbf{w}^T\mathbf{x}_i + b)) \\ &= \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{i=1}^{N} \alpha_i - \sum_{i=1}^{N} \alpha_i y_i \mathbf{w}^T\mathbf{x}_i - b\sum_{i=1}^{N} \alpha_i y_i \end{aligned}$$

KKT conditions:

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i = \mathbf{0} \quad \frac{\partial L}{\partial b} = \sum_{i=1}^{N} \alpha_i y_i = 0$$

$$1 - y_i(\mathbf{w}^T\mathbf{x}_i + b) \leq 0 \quad \alpha_i(1 - y_i(\mathbf{w}^T\mathbf{x}_i + b)) = 0 \quad \alpha_i \geq 0$$

# Formulation

Since $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$, we have:

$$\mathbf{w}^T\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{w}^T \mathbf{x}_i = \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

Note also $\sum_{i=1}^{N} \alpha_i y_i = 0$.

$$\begin{aligned} L(\mathbf{w}, b, \boldsymbol{\alpha}) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{i=1}^{N} \alpha_i - \sum_{i=1}^{N} \alpha_i y_i \mathbf{w}^T \mathbf{x}_i - b\sum_{i=1}^{N} \alpha_i y_i \\ &= \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \end{aligned}$$

# Formulation

Dual problem:

$$\text{Maximize} \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{Subject to} \quad \alpha_i \geq 0$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

Dual problem is also convex optimization problem.

Many software packages available and tailored to solve this form of optimization problem efficiently.

# Formulation

Solving $\mathbf{w}$ and $b$:

- $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$.
- Choose support vector $\mathbf{x}_i$, then $b = y_i - \mathbf{w}^T \mathbf{x}_i$.

Decision function:

- $f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$.

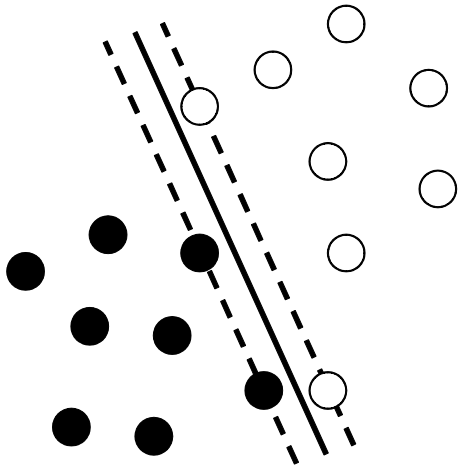
Support vectors:

- KKT condition: $\alpha_i(1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)) = 0$.
- $\alpha_i > 0 \Leftrightarrow y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1 \Leftrightarrow \mathbf{x}_i$ is support vector.

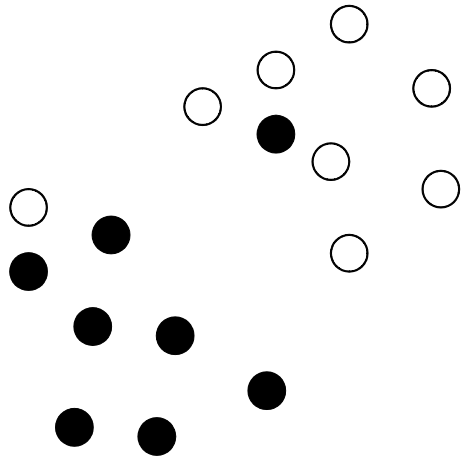
Usually relatively few support vectors, many $\alpha_i$'s will vanish.

# Soft Margin

Narrow margin



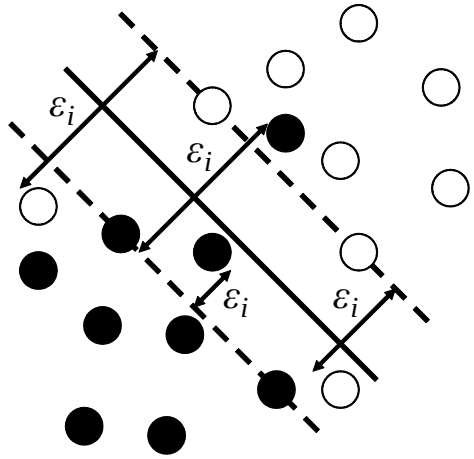
Not linearly separable



Solution: Allow training instances inside the margin or on the other side of the separating hyperplane (misclassified).

# Soft Margin

Soft margin SVM




Allow training instance $(\mathbf{x}_i, y_i)$ to have $y_i(\mathbf{w}^T\mathbf{x}_i + b) < 1$, or $y_i(\mathbf{w}^T\mathbf{x}_i + b) = 1 - \varepsilon_i$ for $\varepsilon_i > 0$, but penalize it with a constant factor $C > 0$.

# Soft Margin

Primal problem:

$$\begin{aligned} \text{Minimize} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\varepsilon_i \\ \text{Subject to} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \varepsilon_i \\ & \varepsilon_i \geq 0 \end{aligned}$$

Dual problem:

$$\begin{aligned} \text{Maximize} \quad & \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j \\ \text{Subject to} \quad & 0 \leq \alpha_i \leq C \\ & \sum_{i=1}^{N}\alpha_i y_i = 0 \end{aligned}$$

## Exercise 4

*Derive the dual problem for the soft margin SVM.*

# Soft Margin

Recall:

- $0 \leq \alpha_i \leq C$.
- $\mathbf{x}_i$ is support vector if $\alpha_i > 0$.

Two types of support vectors:

- $\mathbf{x}_i$ is free support vector if $\alpha_i < C$.
- $\mathbf{x}_i$ is bounded support vector if $\alpha_i = C$.

## Exercise 5

*What are the characteristics of free support vectors and bounded support vectors? (Hint: Consider the KKT conditions.)*

# Soft Margin

Possible to weigh each training instance $\mathbf{x}_i$ with $\lambda_i > 0$ to indicate its importance.

Primal problem:

$$\begin{aligned} \text{Minimize} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\lambda_i\varepsilon_i \\ \text{Subject to} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \varepsilon_i \\ & \varepsilon_i \geq 0 \end{aligned}$$

Cost-sensitive SVM:

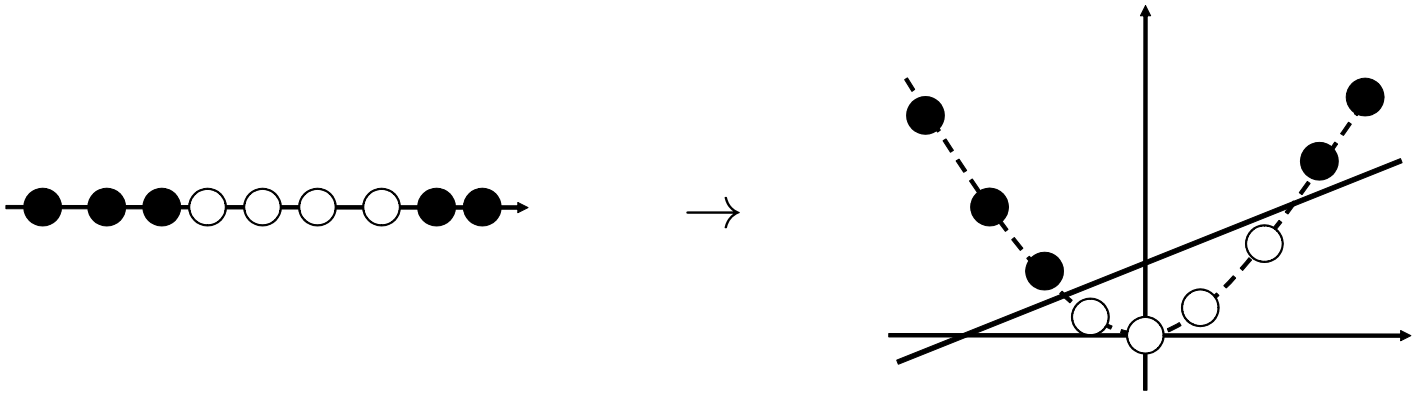
- Different costs for misclassifying positive and negative instances.
- Set $\lambda_i$ proportional to misclassification cost of $\mathbf{x}_i$.

## Exercise 6

*Derive the dual problem for this SVM.*

# Kernel

- Nonlinear feature map $\mathbf{\Phi} : \mathbb{R}^n \to \mathbb{R}^m$ from attribute space to feature space.
- Training instances linearly separable in feature space.



$$\mathbf{\Phi}(x) = (x, x^2)^T$$

Cover's theorem: Instances more likely to be linearly separable in high dimension space.

# Kernel

Dual problem:

$$\text{Maximize} \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \boldsymbol{\Phi}(\mathbf{x}_i)^T \boldsymbol{\Phi}(\mathbf{x}_j)$$

$$\text{Subject to} \quad 0 \leq \alpha_i \leq C$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

Decision function:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{\Phi}(\mathbf{x}_i)^T \boldsymbol{\Phi}(\mathbf{x}) + b$$

Observations:

- Computationally expensive or difficult to compute $\boldsymbol{\Phi}(\mathbf{x})$.
- But often $\boldsymbol{\Phi}(\mathbf{x})^T \boldsymbol{\Phi}(\mathbf{x}')$ has simple expression.

# Kernel

Dual problem:

$$\text{Maximize} \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{Subject to} \quad 0 \leq \alpha_i \leq C$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

Decision function:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b$$

The kernel is the function $K(\mathbf{x}, \mathbf{x}') = \mathbf{\Phi}(\mathbf{x})^T \mathbf{\Phi}(\mathbf{x}')$.

## Exercise 7

*Express $\mathbf{w}^T \mathbf{w}$ and $b$ in terms of the kernel.*
*What is the margin of the kernelized SVM?*

# Kernel

- Kernel $K(\cdot, \cdot)$ can take any expression that satisfies Mercer's condition to ensure $K(\mathbf{x}, \mathbf{x}') = \mathbf{\Phi}(\mathbf{x})^T \mathbf{\Phi}(\mathbf{x}')$ for some feature map $\mathbf{\Phi}(\cdot)$:
  - Mercer's condition: Kernel matrix $\mathbf{K}$ formed by $K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$ is positive semidefinite, i.e., eigenvalues of $\mathbf{K}$ are all nonnegative, or $\mathbf{v}^T \mathbf{K} \mathbf{v} \geq 0$ for all vectors $\mathbf{v}$.
  - No need to compute or even know explicit form of $\mathbf{\Phi}(\cdot)$.
- Examples:
  - Linear kernel: $K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$.
  - Polynomial kernel: $K(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T \mathbf{x}' + k)^d$.
  - Radial basis function (RBF) kernel: $K(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{\|\mathbf{x} - \mathbf{x}'\|_2}{2\sigma^2}\right)$.
- Suggestion: Try linear kernel first, then RBF kernel.

# Contents

# Parameter Tuning

Dual problem:

$$\text{Maximize} \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{Subject to} \quad 0 \leq \alpha_i \leq C$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

RBF kernel:

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{\|\mathbf{x} - \mathbf{x}'\|_2}{2\sigma^2}\right)$$

- Hyperparameters include $C$ of SVM formulation and kernel parameters, e.g., $\sigma^2$ of RBF kernel.
- **Important to select good hyperparameters, has large impact on SVM performance.**

# Parameter Tuning

Traditional way of parameter tuning:

- Use a training set and a validation set.
- Do a grid search on the parameters, e.g., when using RBF kernel – for each $C \in \{2^{-10}, 2^{-8}, \ldots, 2^{0}, \ldots, 2^{8}, 2^{10}\}$ and $\sigma^2 \in \{2^{-10}, 2^{-8}, \ldots, 2^{0}, \ldots, 2^{8}, 2^{10}\}$:
  - Train on training set.
  - Test on validation set.

  Choose $C$ and $\sigma^2$ that gives best performance on validation set.
- Extensions of this idea: $k$-fold cross-validation and leave-one-out cross-validation.

Some SVM implementations have integrated (and more optimized) parameter tuning, others require user to perform own tuning before training.

# Estimating Posterior Probability

Probability of test instance $\mathbf{x}$ belonging to a class can be estimated by:

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(Af(\mathbf{x}) + B)}$$

$A$ and $B$ are parameters to be determined from training data.

# Handling Multiple Classes

- For $k > 2$ classes, decompose into multiple two-class SVMs.
- Pairwise SVMs:
  - Train $\binom{k}{2}$ SVMs, one for each pair of classes.
  - For a test instance, each SVM casts a vote on a class.
  - Classify test instance as the class that gets the most votes.
- One-against-all SVMs:
  - Train $k$ SVMs, the $i$th SVM considers class $i$ as positive and all other classes as negative.
  - Classify test instance to class $i$ if the $i$th SVM gives the greatest $f(\cdot)$ value.

# Applications

Applications:

- Text processing:
  - Each text document is an instance.
  - Each word is an attribute.
  - Attribute values are word counts.
- Image processing:
  - Each image is an instance.
  - Each pixel is an attribute.
  - Attribute values are pixel values.

Cover's theorem: Instances more likely to be linearly separable in high dimension space.

These applications already have many attributes, and each instance is a sparse vector, so linear SVM often performs well.

# Applications

SVM decision function:

$$f(\mathbf{x}) = w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4 + b$$

2-norm of weight vector is $\|\mathbf{w}\|_2 = \sqrt{w_1^2 + w_2^2 + w_3^2 + w_4^2}$.

Test instance with missing attribute values: $\mathbf{x} = (x_1, ?, x_3, ?)$

Substitute with constants attributewise: $\mathbf{x} = (x_1, x_2', x_3, x_4')$

Decision function becomes:

$$f(\mathbf{x}) = w_1 x_1 + w_2 x_2' + w_3 x_3 + w_4 x_4' + b = w_1 x_1 + w_3 x_3 + b'$$

New weight vector is $\mathbf{w}'(\mathbf{x}) = (w_1, w_3)^T$ and its 2-norm decreased to $\|\mathbf{w}'(\mathbf{x})\|_2 = \sqrt{w_1^2 + w_3^2}$.

Can use $f(\mathbf{x})$, $\|\mathbf{w}\|_2$ and $\|\mathbf{w}'(\mathbf{x})\|_2$ to estimate probability of misclassification, and hence whether to acquire missing attribute values.

# Contents

# Decision Tree

Advantages of SVM:

- SVM can handle linear relationships between attributes, but typical decision tree split only on one attribute at a time.
- Typically better on datasets with many attributes.
- Handles continuous values well.

Advantages of decision tree:

- Decision tree is better at handling nested if-then-else type of rules, which SVM is not good at.
- Typically better on datasets with fewer attributes.
- Handles discrete values well.

# $K$-nearest Neighbours

Advantages of SVM:

- Faster classification compared to $K$-nearest neighbours.
- Smooth separating hyperplane.
- Less suspectible to noise.

Advantages of $K$-nearest neighbours:

- Training is instantaneous – nothing to do.
- Local variations are considered.

# Naïve Bayes

- Naïve Bayes often has the dubious honour of being placed among the last in evaluations.
- Naïve Bayes is fast and easy to implement, hence often treated as a baseline.
- One key handicap of naïve Bayes is its independence assumption.

# Neural Network

- A linear SVM can be seen as a neural network with no hidden layers.
- But training algorithm is different.
  - SVM design is driven by sound theory, neural network design is driven by applications.
  - Backpropagation algorithm for training multi-layer neural network does not find the maximal margin separating hyperplane.
  - Neural network tend to overfit more than SVM.
  - Neural network have many local minima, SVM has only one global minima.
- Multi-layer neural networks require specifying number of hidden layers and number of nodes at each layer, but SVM does not need them.
- What do the learned weights in a trained multi-layer neural network mean?

# Combination of Classifiers

Possible to combine classifiers:

- SVM with decision tree:
  - Instead of splitting based only on one attribute, split using a linear classifier trained by SVM.
- SVM with $K$-nearest neighbours:
  - Use $K$-nearest neighbours to classify test instances inside the margin.
- SVM with naïve Bayes:
  - Train Naïve Bayes classifier, produces conditional probabilities $P(x_i|c)$.
  - Modify every (training and testing) instance $\mathbf{x}$ by multiplying each attribute $x_i$ by $P(x_i|c)$.
  - Train and test on the modified instances.

# Contents

# Summary

- SVM is a maximum margin linear classifier.
- Often good for classifying test instances in high dimensional space.
- Lagrangian theory – primal and dual problems.
- When training instances are not linearly separable:
  - Use soft margin.
  - Use kernel.
- Usage tips:
  - Transforming non-numeric attributes.
  - Normalize attributes.
  - Tune parameters when training.
- Comparison with other classifiers.

# Bibliography


- Introduction to SVM, with applications: [Burges, 1998], [Osuna et al., 1997].
- Practical guide on using SVM effectively: [Hsu et al., 2003].
- LIBSVM manual, including weighted instances: [Chang and Lin, 2001].
- Tutorial on Lagrangian theory: [Burges, 2003], [Klien, 2004].
- SVM with posterior probabilities: [Platt, 2000].
- Attribute selection using SVM: [Guyon et al., 2002].
- Handling multiple classes: [Hsu and Lin, 2002], [Tibshirani and Hastie, 2007].
- SVM combined with...
  - Decision tree: [Bennett and Blue, 1998], [Tibshirani and Hastie, 2007].
  - $K$-nearest neighbours: [Chiu and Huang, 2007].
  - Naïve Bayes: [Li et al., 2007].

# References I

Bennett, K. P. and Blue, J. A. (1998).
A support vector machine approach to decision trees.
In *IEEE World Congress on Computational Intelligence*, pages 2396–2401.

Burges, C. J. C. (1998).
A tutorial on support vector machines for pattern recognition.
*Data Mining and Knowledge Discovery*, 2(2):121–167.

Burges, C. J. C. (2003).
Some notes on applied mathematics for machine learning.
In *Advanced Lectures on Machine Learning*, pages 21–40.

Chang, C.-C. and Lin, C.-J. (2001).
*LIBSVM: a library for support vector machines*.
Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`.

Chiu, C.-Y. and Huang, Y.-T. (2007).
Integration of support vector machine with naïve bayesian classifier for spam classification.
In *International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pages 618–622.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002).
Gene selection for cancer classification using support vector machines.
*Machine Learning*, 46(1-3):389–422.

Hsu, C.-W., Chang, C.-C., and Lin, C.-J. (2003).
A practical guide to support vector classification.
Technical report, Department of Computer Science, National Taiwan University.

Hsu, C.-W. and Lin, C.-J. (2002).
A comparison of methods for multiclass support vector machines.
*IEEE Transactions on Neural Networks*, 13(2):415–425.

# References II

Klien, D. (2004).

Lagrange multipliers without permanent scarring.
Available at `http://www.cs.berkeley.edu/~klein/papers/lagrange-multipliers.pdf`.

Li, R., Wang, H.-N., He, H., Cui, Y.-M., and Du, Z.-L. (2007).

Support vector machine combined with k-nearest neighbors for solar flare forecasting.
*Chinese Journal of Astronomy and Astrophysics*, 7(3):441–447.

Osuna, E. E., Freund, R., and Girosi, F. (1997).

Support vector machines: Training and applications.
Technical Report AIM-1602, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Platt, J. (2000).

Probabilistic outputs for support vector machines and comparison to regularized likelihood methods.
In *Advances in Large Margin Classifiers*, pages 61–74.

Tibshirani, R. and Hastie, T. (2007).

Margin trees for high-dimensional classification.
*Journal of Machine Learning Research (JMLR)*, 8:637–652.