# EE5904/ME5404 Part II

# Project 1
## SVM for Classification of Spam Email Messages

REPORT DUE ON 22 APRIL 2022

Thushara Sandakalum

sandakalum@u.nus.edu

# EE5904/ME5404 Part II

# Project 1
## SVM for Classification of Spam Email Messages

REPORT DUE ON 22 APRIL 2022

Thushara Sandakalum

sandakalum@u.nus.edu

# Outline

Project description

Recap

Task 1 : Train

Task 2 : Test

Task 3 : Evaluate

Important Notes

# Project Description

Project Goal

- Implement a SVM to classify spam or not a spam for the **Spam Email Data Set**

- Spam Email Data Set

  - **4601 samples** of email metadata taken from **UC Irvine Machine Learning Repository**

  - **57 features** per sample

  - **Label : +1 (spam), -1 (non-spam)**

  - http://archive.ics.uci.edu/ml/datasets/spambase

```
0.00000 0.01043 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.01043 0.01043 0.02105 0.00000 0.00000 0.00000
0.00000 0.03166 0.06332 0.00000 0.02105 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00196 0.00000 0.00000 0.02601 0.12811
0.98827
```

48 continuous real [0,100] attributes of type word_freq_WORD
= percentage of words in the e-mail that match WORD,
i.e. 100 * (number of times the WORD appears in the e-mail) /
total number of words in e-mail. A "word" in this case is any
string of alphanumeric characters bounded by non-alphanumeric
characters or end-of-string.

6 continuous real [0,100] attributes of type char_freq_CHAR
= percentage of characters in the e-mail that match CHAR,
i.e. 100 * (number of CHAR occurrences) / total characters in e-mail

# Project Description

Project Goal

- Implement a SVM to classify spam or not a spam for the **Spam Email Data Set**

- Spam Email Data Set
  - **4601 samples** of email metadata taken from **UC Irvine Machine Learning Repository**
  - **57 features** per sample
  - **Label : +1 (spam), -1 (non-spam)**
  - http://archive.ics.uci.edu/ml/datasets/spambase

- Dataset divided into **3 subsets**
  - Training set
  - Test set
  - Evaluation set (Not provided)

| Name ▲ | Value |
| --- | --- |
| eval_data | 57x600 double |
| eval_label | 600x1 double |
| test_data | 57x1536 double |
| test_label | 1536x1 double |
| train_data | 57x2000 double |
| train_label | 2000x1 double |

# Project Description

**Train** ➡

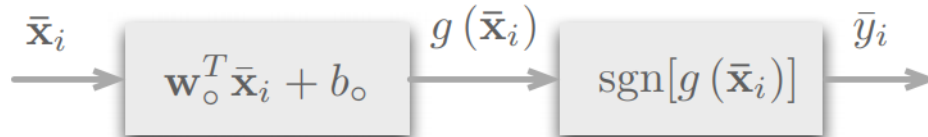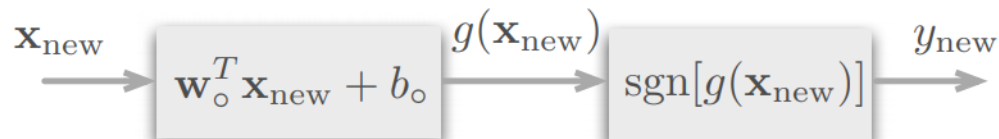Construction: For a given training set $S = \{(\mathbf{x}_1, d_1), \ldots, (\mathbf{x}_N, d_N)\}$, find optimal hyperplane $(\mathbf{w}_\circ, b_\circ)$ such that, for all $i \in \{1, 2, \ldots, N\}$,

$$\mathbf{x}_i \longrightarrow \boxed{\mathbf{w}_\circ^T \mathbf{x}_i + b_\circ} \xrightarrow{g(\mathbf{x}_i)} \boxed{\mathrm{sgn}[g(\mathbf{x}_i)]} \xrightarrow{y_i = d_i}$$
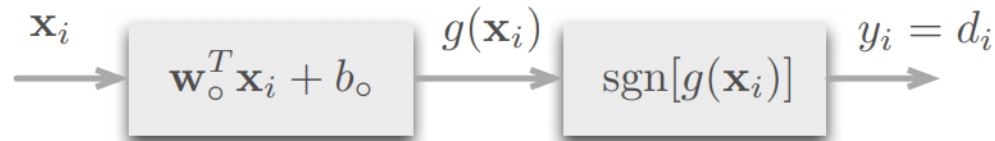
**Test** ➡

Testing: For a given test set $\bar{S} = \{(\bar{\mathbf{x}}_1, \bar{d}_1), \ldots, (\bar{\mathbf{x}}_{\bar{N}}, \bar{d}_{\bar{N}})\}$, compute output $\bar{y}_i$ of SVM (with $\mathbf{w}_\circ$ and $b_\circ$) for all $i \in \{1, 2, \ldots, \bar{N}\}$, and compare it against the known $\bar{d}_i$ to evaluate performance of SVM

$$\bar{\mathbf{x}}_i \longrightarrow \boxed{\mathbf{w}_\circ^T \bar{\mathbf{x}}_i + b_\circ} \xrightarrow{g(\bar{\mathbf{x}}_i)} \boxed{\mathrm{sgn}[g(\bar{\mathbf{x}}_i)]} \xrightarrow{\bar{y}_i}$$

**Evaluate** ➡

Application: Given a SVM with hyperplane $(\mathbf{w}_\circ, b_\circ)$, classify a data point $\mathbf{x}_{\mathrm{new}}$ that is not in $\Sigma = S \cup \bar{S}$:

$$\mathbf{x}_{\mathrm{new}} \longrightarrow \boxed{\mathbf{w}_\circ^T \mathbf{x}_{\mathrm{new}} + b_\circ} \xrightarrow{g(\mathbf{x}_{\mathrm{new}})} \boxed{\mathrm{sgn}[g(\mathbf{x}_{\mathrm{new}})]} \xrightarrow{y_{\mathrm{new}}}$$

# Recap – Hard Margin



Support vector

Optimal hyperplane

$x_2$

$x_1$

Margin of separation

## Discriminant function

$$g(\mathbf{x}) = \mathbf{w}_\circ^T \mathbf{x} + b_\circ$$
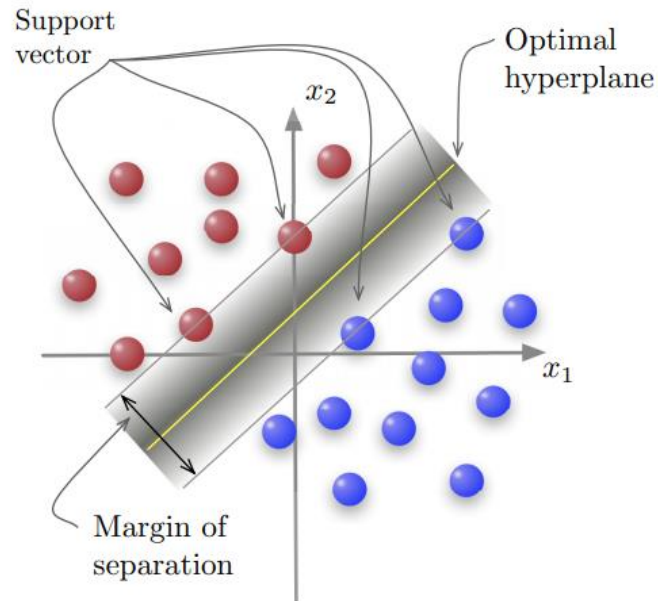
## Support vector: $\mathbf{x}_i$ that satisfies

$$g(\mathbf{x}_i) = \pm 1$$

Primal problem

Given data set : $S = \{(\mathbf{x}_i, d_i)\}, \quad i = 1, 2, \ldots, N$

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

Minimizing : $f(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$

Subject to : $d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1$

Known parameters: $\mathbf{x}_i, d_i$
Unknown variables: $\mathbf{w}, b$

Solve $\Rightarrow$ Optimum hyperplane

# Recap – Hard Margin

Finding optimal hyperplane (primal problem) Slide 73

$$\text{Given data set}: \quad S = \{(\mathbf{x}_i, d_i)\}$$
$$\text{Find}: \quad \mathbf{w} \text{ and } b$$
$$\text{Minimizing}: \quad f(\mathbf{w}) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w}$$
$$\text{Subject to}: \quad d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1$$

Alternative formulation using method of **Lagrange multipliers**

Finding optimal hyperplane (dual problem) Slide 79

$$\text{Given}: \quad S = \{(\mathbf{x}_i, d_i)\}$$
$$\text{Find}: \quad \text{Lagrange multipliers } \{\alpha_i\}$$

Linear kernel

$$\text{Maximizing}: \quad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\,\alpha_j\,d_i\,d_j\,\mathbf{x}_i^T\mathbf{x}_j$$

$$\text{Subject to}: \quad (1) \quad \sum_{i=1}^{N}\alpha_i\,d_i = 0$$
$$(2) \quad \alpha_i \geq 0$$

Karush-Kuhn-Tucker conditions

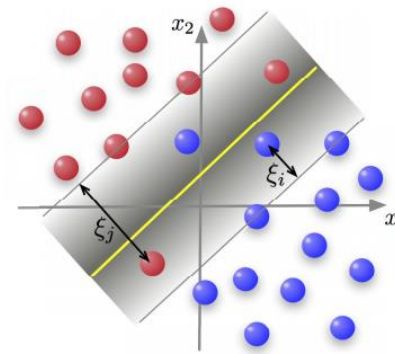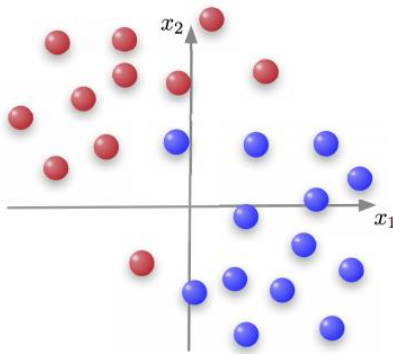For data point $x_i$ that is a support vector $\alpha_{o,i} \neq 0$

# Recap – Soft Margin

**Dealing with non-separable patterns:**

1. Find optimal hyperplane to minimize classification error     Slide 94

New function to be minimized

$$f(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i$$

**Dual problem** (with soft margin)     Slide 100                    Linear kernel          Slide 96

Find : $\alpha_i$

Maximize : $Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j$

- Value of $C > 0$ reflects cost of violating constraints
  - A large $C$ generally leads to smaller margin but also fewer misclassification of training data
  - A small $C$ generally leads to larger margin but more misclassification of training data

Subject to : $\sum_{i=1}^{N}\alpha_i d_i = 0$  and  $0 \le \alpha_i \le C$

- As a design parameter, value of $C$ is set by user

Soft Margin

# Recap – Soft Margin

Dealing with non-separable patterns:

2. Transform data into higher dimension space for separation Slide 94



Input space

Feature space

$\bullet = \varphi(\bullet)$
$\bullet = \varphi(\bullet)$
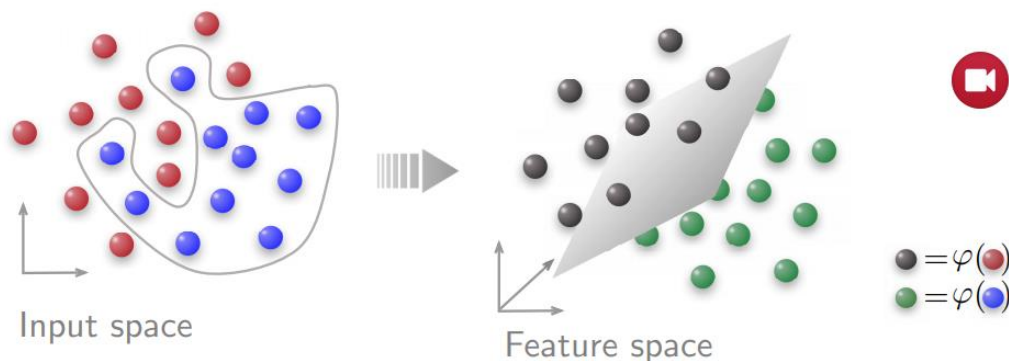
Dual problem with soft margin and transformation Slide 115

Find : $\alpha_i$

Maximize : $Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \boldsymbol{\varphi}^T(\mathbf{x}_i) \boldsymbol{\varphi}(\mathbf{x}_j)$

Subject to : $\sum_{i=1}^{N} \alpha_i d_i = 0, \ 0 \le \alpha_i \le C$

Nonlinear kernel

Soft Margin

# Task 1 - Data

Training set – 2000 samples

- Given – 'train.mat'
  - Features – (57 x 2000)
  - Label (2000 x 1)

- Features of a sample

```
0.00000 0.01043 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.01043 0.01043 0.02105 0.00000 0.00000 0.00000
0.00000 0.03166 0.06332 0.00000 0.02105 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00196 0.00000 0.00000 0.02601 0.12811
0.98827
```

- Label : +1 (spam), -1 (non-spam)

# Task 1 – Training set

Import the training set (i.e. train.mat)
- train_data (57 x 2000)
- train_label (2000 x 1)

Preprocess the 'data' (Various methods can be used including Sample scaling and Standardization [CHOOSE ONE METHOD])[a, b]

- Scale the data – Rescale the individual <u>sample x</u> such that $\|x\| = 1$

- Standardize the data – Transform each <u>feature</u> by removing the <u>mean</u> value of each feature and then dividing by each <u>feature's standard deviation</u>

Please ensure the 'label' is mapped into the set of {-1, +1}

[a] https://scikit-learn.org/stable/modules/preprocessing.html
[b] https://en.wikipedia.org/wiki/Feature_scaling

# Task 1 – Kernels

Hard-margin SVM with the linear kernel

$$K(x_1, x_2) = x_1^T x_2$$

Hard-margin SVM with a polynomial kernel

$$K(x_1, x_2) = (x_1^T x_2 + 1)^p$$

Soft-margin SVM with a polynomial kernel

$$K(x_1, x_2) = (x_1^T x_2 + 1)^p$$

# Task 1 – Hard and Soft Margins

Hard margin $0 \leq \alpha_i$

- C = $+ \infty$ (In theory)

  $0 \leq \alpha_i \leq C$

- C = <u>Large value (In practice e.g. $10^6$)</u>

Soft margin $0 \leq \alpha_i \leq C$

- C = 0.1, 0.6, 1.1, 2.1

# Task 1 – Calculate $\alpha_i$

How to calculate $\alpha_i$

$$\text{Maximize :} \quad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{Subject to :} \quad \sum_{i=1}^{N} \alpha_i d_i = 0, \; 0 \leq \alpha_i \leq C$$

Use quadprog function (Quadratic programming)

**Description**

Solver for quadratic objective functions with linear constraints.

quadprog finds a minimum for a problem specified by

$$\min_{x} \frac{1}{2} x^T H x + f^T x \text{ such that } \begin{cases} A \cdot x \leq b, \\ Aeq \cdot x = beq, \\ lb \leq x \leq ub. \end{cases}$$

$H$, $A$, and $Aeq$ are matrices, and $f$, $b$, $beq$, $lb$, $ub$, and $x$ are vectors.

You can pass $f$, $lb$, and $ub$ as vectors or matrices; see Matrix Arguments.

`x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options)` solves the preceding problem using the optimization options specified in `options`. Use `optimoptions` to create `options`. If you do not want to give an initial point, set `x0 = []`.

# Task 1 – Calculate $\alpha_i$ - quadprog

**Description**

Solver for quadratic objective functions with linear constraints.

quadprog finds a minimum for a problem specified by

Maximize:
$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j K(\mathbf{x}_i, \mathbf{x}_j)$$

Subject to : $\sum_{i=1}^{N} \alpha_i d_i = 0, \ 0 \le \alpha_i \le C$

$$\min_{x} \frac{1}{2} x^T H x + f^T x \text{ such that } \begin{cases} A \cdot x \le b, \\ Aeq \cdot x = beq, \\ lb \le x \le ub. \end{cases}$$

Convert the problem from 'Max' to 'Min'

◦ Max Q(α) → Min - Q(α)

If $f$ is to be maximized instead, such a maximization problem can be expressed as a minimization problem by the transformation

$$\max_{\mathbf{w}} f(\mathbf{w}) = -\min_{\mathbf{w}}\left[-f(\mathbf{w})\right]$$

Slide 62

# Task 1 – Calculate $\alpha_i$ - quadprog

Maximize: $Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \dfrac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j K(\mathbf{x}_i, \mathbf{x}_j)$

Subject to : $\sum_{i=1}^{N} \alpha_i d_i = 0, \ 0 \le \alpha_i \le C$

**Description**

Not used

Solver for quadratic objective functions with linear constraints.

quadprog finds a minimum for a problem specified by

$$\min_{x} \frac{1}{2} x^T H x + f^T x \text{ such that } \begin{cases} A \cdot x \le b, \\ Aeq \cdot x = beq, \\ lb \le x \le ub. \end{cases}$$

$\min_{x} \dfrac{1}{2} x^T H x + f^T x$
$\begin{cases} H_{ij} = d_i d_j K(x_i, x_j) \\ f = (-1, -1, \cdots\cdots, -1)^T \end{cases}$

$A \cdot x \le b,$
$\begin{cases} A = [\ ] \\ b = [\ ] \end{cases}$

$Aeq \cdot x = beq,$
$\begin{cases} Aeq = (d_1, d_2, \cdots\cdots, d_N) \\ beq = 0 \end{cases}$

$lb \le x \le ub.$
$\begin{cases} lb = (0, 0, \cdots\cdots, 0)^T \\ ub = (C, C, \cdots\cdots, C)^T \end{cases}$

```
x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options)
```

# Task 1 – Calculate $\alpha_i$ - quadprog

```
x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options)
```

Hard-margin SVM with the linear kernel
$$K(x_1, x_2) = x_1^T x_2$$

For illustration only

$\min\limits_{x} \frac{1}{2} x^T H x + f^T x$ 
$\left\{\begin{array}{l} H(i,j) = d_i d_j x_i^T x_j \\ f = -\text{ones}(2000,1) \end{array}\right.$

$A \cdot x \leq b,$
$\left\{\begin{array}{l} A = [\,] \\ b = [\,] \end{array}\right.$

$Aeq \cdot x = beq,$
$\left\{\begin{array}{l} Aeq = train\_label' \\ beq = 0 \end{array}\right.$

$lb \leq x \leq ub.$
$\left\{\begin{array}{l} lb = zeros(2000,1) \\ ub = \text{ones}(2000,1) * C \end{array}\right.$

Hard margin $0 \leq \alpha_i$
C = $+\infty$ (In theory)
C = Large value (In practice e.g. $10^6$)

$x0 = [\,]$     $options = optimset('LargeScale', 'off', 'MaxIter', 1000)$

# Task 1 – Calculate $\alpha_i$ - quadprog

```
x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options)
```

Hard-margin SVM with a polynomial kernel
$$K(x_1, x_2) = (x_1^T x_2 + 1)^p$$

For illustration only

$$\min_x \frac{1}{2} x^T H x + f^T x$$

$\left\{\begin{array}{l} H(i,j) = d_i d_j (x_1^T x_2 + 1)^p \\ f = -\text{ones}(2000,1) \end{array}\right.$

$$A \cdot x \leq b,$$

$\left\{\begin{array}{l} A = [\,] \\ b = [\,] \end{array}\right.$

$$Aeq \cdot x = beq,$$

$\left\{\begin{array}{l} Aeq = train\_label' \\ beq = 0 \end{array}\right.$

$$lb \leq x \leq ub.$$

$\left\{\begin{array}{l} lb = zeros(2000,1) \\ ub = \text{ones}(2000,1) * C \end{array}\right.$

Hard margin $0 \leq \alpha_i$
C = $+\infty$ (In theory)
C = <u>Large value (In practice e.g. $10^6$)</u>

$x0 = [\,]$     $options = optimset('LargeScale','off','MaxIter', 1000)$

# Task 1 – Calculate $\alpha_i$ - quadprog

```
x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options)
```

Soft-margin SVM with a polynomial kernel
$$K(x_1, x_2) = (x_1^T x_2 + 1)^p$$

For illustration only

$$\min_{x} \frac{1}{2} x^T H x + f^T x$$

$$H(i,j) = d_i d_j (x_1^T x_2 + 1)^p$$
$$f = -\text{ones}(2000,1)$$

$$A \cdot x \leq b,$$

$$A = [\,]$$
$$b = [\,]$$

$$Aeq \cdot x = beq,$$

$$Aeq = train\_label'$$
$$beq = 0$$

$$lb \leq x \leq ub.$$

$$lb = zeros(2000,1)$$
$$ub = \text{ones}(2000,1) * C$$

Soft margin $0 \leq \alpha_i \leq C$
C = 0.1, 0.6, 1.1, 2.1

$$x0 = [\,] \qquad options = optimset('LargeScale','off','MaxIter', 1000)$$

# Task 1 – Select support vectors

Based on KKT conditions

- For a support vector, $\alpha_i \neq 0$ (In theory, $\alpha_i > 0$ )

- However, in practice, $\alpha_i > threshold$

- How to decide?
  - Choose an appropriate threshold (e.g. 1e-4) to determine the corresponding $\alpha_i$ to the support vectors
  - Then make the other smaller $\alpha_i$ values which are less than the threshold zero

# Task 1 – Discriminant function g(x)

Hard Margin SVM with Linear Kernel

Maximizing : $Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \dfrac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, \alpha_j \, d_i \, d_j \, \mathbf{x}_i^T \mathbf{x}_j$

Subject to :  (1)  $\sum_{i=1}^{N} \alpha_i \, d_i = 0$

  (2)  $\alpha_i \geq 0$

Discriminant function

$g(\mathbf{x}) = \mathbf{w}_\circ^T \mathbf{x} + b_\circ$

After $\alpha_{\circ,i}$ is obtained, we can calculate $\mathbf{w}_\circ$ and $b_\circ$ as follows:

$$\mathbf{w}_\circ = \sum_{i=1}^{N} \alpha_{\circ,i} \, d_i \, \mathbf{x}_i \,, \quad b_\circ = \dfrac{1}{d^{(s)}} - \mathbf{w}_\circ^T \mathbf{x}^{(s)}$$

where $\mathbf{x}^{(s)}$ is a support vector with label $d^{(s)}$

# Task 1 – Discriminant function g(x)

Soft Margin SVM with Linear Kernel

Maximize : $Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j$

Subject to : $\sum_{i=1}^{N} \alpha_i d_i = 0$ and $0 \le \alpha_i \le C$

**Discriminant function**

$$g(\mathbf{x}) = \mathbf{w}_\circ^T \mathbf{x} + b_\circ$$

After $\alpha_{\circ,i}$ is obtained, we can calculate $\mathbf{w}_\circ$ as follows:     Slide 104,105

$$\mathbf{w}_\circ = \sum_{i=1}^{N} \alpha_{\circ,i} \, d_i \, \mathbf{x}_i$$

After $\mathbf{w}_\circ$ is obtained, we can calculate $b_\circ$ as follows:     ② Take $b_\circ$ as the average of all such $b_{\circ,i}$

① For each example $\mathbf{x}_i$ with $0 < \alpha_i \le C$,

$$b_\circ = \frac{\sum_{i=1}^{m} b_{\circ,i}}{m}$$

$$b_{\circ,i} = \frac{1}{d_i} - \mathbf{w}_\circ^T \mathbf{x}_i$$

where $m$ is the total number of $\mathbf{x}_i$ with $0 < \alpha_i \le C$.

# Task 1 – Discriminant function g(x)

**Soft Margin** SVM with <u>Nonlinear Kernel</u>

Maximize : $Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \boldsymbol{\varphi}^T(\mathbf{x}_i) \boldsymbol{\varphi}(\mathbf{x}_j)$

Subject to : $\sum_{i=1}^{N} \alpha_i d_i = 0, \ 0 \leq \alpha_i \leq C$

Discriminant function

$$g(\mathbf{x}) = \sum_{i=1}^{N} \alpha_{\circ,i} d_i K(\mathbf{x}, \mathbf{x}_i) + b_\circ$$

Determine $b_\circ$ in   Slide 123

$$g(\mathbf{x}) = \sum_{i=1}^{N} \alpha_{\circ,i} d_i K(\mathbf{x}, \mathbf{x}_i) + b_\circ$$

using the fact that for a support vector $\mathbf{x}^{(s)}$

$$g(\mathbf{x}^{(s)}) = \pm 1 = d^{(s)}$$

Take $b_\circ$ as the average of all such $b_{\circ,i}$

$$b_\circ = \frac{\sum_{i=1}^{m} b_{\circ,i}}{m}$$

where $m$ is the total number of $\mathbf{x}_i$ with $0 < \alpha_i \leq C$.

# Task 1 – Summary

Given a training set $S = \{(\mathbf{x}_i, d_i)\}$, $i = 1, \ldots, N$

**1** Find a suitable kernel

Choose expression then check Mercer's condition

Soft Margin

**2** Choose a value for $C$

Quadratic programming

**3** Solve for $\alpha_{\circ,i}$

**4** Determine $b_\circ$ in

Kernel

$$g(\mathbf{x}) = \sum_{i=1}^{N} \alpha_{\circ,i} d_i K(\mathbf{x}, \mathbf{x}_i) + b_\circ$$
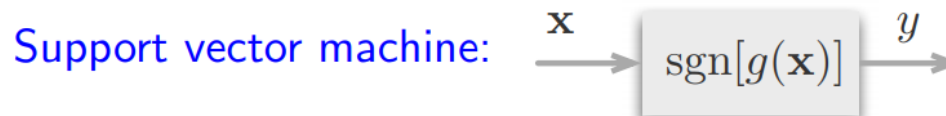
using the fact that for a support vector $\mathbf{x}^{(s)}$

$$g(\mathbf{x}^{(s)}) = \pm 1 = d^{(s)}$$

Maximize : $\quad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j K(\mathbf{x}_i, \mathbf{x}_j)$

Subject to : $\quad \sum_{i=1}^{N} \alpha_i d_i = 0, \ 0 \leq \alpha_i \leq C$

Support vector machine: $\quad \mathbf{x} \longrightarrow \boxed{\mathrm{sgn}[g(\mathbf{x})]} \longrightarrow y$

# Task 2 - Data

Test set – 1536 samples
- Given – 'test.mat'
  - Features – (57 x 1536)
  - Label (1536 x 1)

- Features of a sample

```
0.00000 0.01043 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.01043 0.01043 0.02105 0.00000 0.00000 0.00000
0.00000 0.03166 0.06332 0.00000 0.02105 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00196 0.00000 0.00000 0.02601 0.12811
0.98827
```

- Label : +1 (spam), -1 (non-spam)

# Task 2 – Test set

Import the test set (i.e. test.mat)
- test_data (57 x 1536)
- test_label (1536 x 1)

Preprocess the 'data' (Various methods can be used including Sample scaling and Standardization [CHOOSE ONE USE for TRAINING])[a, b]

- Scale the data – Rescale the individual <u>sample</u> x such that $\|x\| = 1$

- Standardize the data – Transform each <u>feature</u> in the same manner with the training data. Use the <u>mean and variance of each feature</u> from <u>your training set</u>.

Please ensure the 'label' is mapped into the set of {-1, +1}

[a] https://scikit-learn.org/stable/modules/preprocessing.html
[b] https://en.wikipedia.org/wiki/Feature_scaling

# Task 2 – Test set

Discriminant function

$$g(\mathbf{x}) = \mathbf{w}_\circ^T \boldsymbol{\varphi}(\mathbf{x}) + b_\circ = \sum_{i=1}^{N} \alpha_{\circ,i} d_i \underbrace{\boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x})}_{K(\mathbf{x}_i,\mathbf{x})} + b_\circ$$

To classify a new data point $\mathbf{x}_{\text{new}}$

$$d_{\text{new}} = \text{sgn}\left[g(\mathbf{x}_{\text{new}})\right]$$

For illustrations only

$$g(x_{test}) = \sum_{i=1}^{N} \alpha_{o,i} d_i K(x_i, x_{test}) + b_0$$

If g(x_test) > 0
    x_test_label = +1
Else
    x_test_label = -1

# Task 2 – Test set

**Discriminant function**

$$g(\mathbf{x}) = \mathbf{w}_{\circ}^{T}\boldsymbol{\varphi}(\mathbf{x}) + b_{\circ} = \sum_{i=1}^{N} \alpha_{\circ,i} d_i \underbrace{\boldsymbol{\varphi}^{T}(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x})}_{K(\mathbf{x}_i,\mathbf{x})} + b_{\circ}$$

To classify a new data point $\mathbf{x}_{\text{new}}$

$$d_{\text{new}} = \text{sgn}\left[g(\mathbf{x}_{\text{new}})\right]$$

| Type of SVM | Training accuracy | | | | Test accuracy | | | |
|---|---|---|---|---|---|---|---|---|
| Hard margin with Linear kernel | ? | | | | ? | | | |
| Hard margin with polynomial kernel | $p=2$ | $p=3$ | $p=4$ | $p=5$ | $p=2$ | $p=3$ | $p=4$ | $p=5$ |
| | ? | ? | ? | ? | ? | ? | ? | ? |
| Soft margin with polynomial kernel | $C=0.1$ | $C=0.6$ | $C=1.1$ | $C=2.1$ | $C=0.1$ | $C=0.6$ | $C=1.1$ | $C=2.1$ |
| $p=1$ | ? | ? | ? | ? | ? | ? | ? | ? |
| $p=2$ | ? | ? | ? | ? | ? | ? | ? | ? |
| $p=3$ | ? | ? | ? | ? | ? | ? | ? | ? |
| $p=4$ | ? | ? | ? | ? | ? | ? | ? | ? |
| $p=5$ | ? | ? | ? | ? | ? | ? | ? | ? |

If g(x_test) > 0

  x_test_label = +1

Else

  x_test_label = -1

# Task 3 - Data

Evaluation set – 600 samples
- **Not Given** – 'eval.mat'
  - eval_data (57 x 600)
  - eval_label (600 x 1)

# Task 3 - Evaluation

Design your own SVM
- ◦ Hard margin or Soft margin?
- ◦ Linear or Polynomial kernel?
- ◦ What are the values for p and C?

→ To produce the best performance

To classify the 600 samples in the evaluation set

Not Given – 'eval.mat'
eval_data (57 x 600)
eval_label (600 x 1)

Output : A column vector (600 x 1) named 'eval_predicted'

# Task 3 - Evaluation

Hardcode the discriminant function g(x) in the file for evaluation
- ◦ If necessary, store the required variables in a separate *.mat file and load at the beginning of the code
- ◦ Don't run 'quadprog'

Prepare the code so that it could handle the evaluation dataset and able to preprocess the evaluation dataset (use train dataset statistics)
- ◦ Note: the eval_data is a (57 x 600) matrix

Your code should generate a column vector (600 x 1) named 'eval_predicted'

# Task 3 - Evaluation

Please name your m-file for Task 3 as 'svm_main.m'

Do Not clear any variable in the 'svm_main' script
◦ Don't use 'clear' command

Don't use 'load eval.mat'

Before submitting your code, please ensure that the code runs without errors by testing it with a dummy data set (A Dummy dataset can be created using the training set and test set)

# Important Notes

Preprocess your data – Choose one method
- ◦ Sample scaling/ Mean normalization/ standardization/ Rescaling …

Use the training set statistics to preprocess the other data sets
- ◦ For training set, test set and eval set

Check Mercer Condition for Kernel suitability

Don't use inbuilt MATLAB functions like 'fitsvm', 'predict'

# Important Notes

Procedure to build SVM

- Preprocess data
- Choose a suitable kernel
  - Linear/ Nonlinear ?
- Choose C
  - Hard margin/ Soft margin
    - Hard margin $0 \leq \alpha_i$
      - C = $+\infty$ (In theory)
      - C = Large value (In practice e.g. $10^6$)
- Solve for $\alpha_i$
  - Quadratic programming
- Support vector selection
  - Choose an appropriate threshold (e.g. 1e-4) to determine the corresponding $\alpha_i$ to the support vectors and make smaller $\alpha_i$ values zero.
- Determine the discriminant function g(x)

# Important Notes : Submission

Submit all your codes that you have implemented for the entire project

Submit all required files – even if it is 'train.mat'

Make sure your codes run without error

All codes should be executable with the given datasets in the workspace without any additional inputs

# Important Notes : Submission

Report

◦ Details on Implementation

◦ Completed Table 1

◦ Discussion under following subheadings

   ◦ Data pre-processing

   ◦ Admissibility of the kernels

   ◦ Existence of optimal hyperplanes

   ◦ Table 1 - Comments on results (with supporting arguments)

   ◦ Task 3 – Discussion on design decisions

| Type of SVM | Training accuracy | | | | Test accuracy | | | |
|---|---|---|---|---|---|---|---|---|
| Hard margin with Linear kernel | ? | | | | ? | | | |
| Hard margin with polynomial kernel | $p=2$ | $p=3$ | $p=4$ | $p=5$ | $p=2$ | $p=3$ | $p=4$ | $p=5$ |
| | ? | ? | ? | ? | ? | ? | ? | ? |
| Soft margin with polynomial kernel | $C=0.1$ | $C=0.6$ | $C=1.1$ | $C=2.1$ | $C=0.1$ | $C=0.6$ | $C=1.1$ | $C=2.1$ |
| $p=1$ | ? | ? | ? | ? | ? | ? | ? | ? |
| $p=2$ | ? | ? | ? | ? | ? | ? | ? | ? |
| $p=3$ | ? | ? | ? | ? | ? | ? | ? | ? |
| $p=4$ | ? | ? | ? | ? | ? | ? | ? | ? |
| $p=5$ | ? | ? | ? | ? | ? | ? | ? | ? |

TABLE I: Results of SVM classification.

# Important Notes : Submission

svm_main.m          Other m-files, if needed *.mat files          A0123456B_SVM.pdf
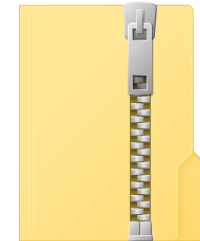
Example student No = A0123456B

Folder name = A0123456B          Non password protected

Due date 22 April 2022

[sandakalum@u.nus.edu](mailto:sandakalum@u.nus.edu)